4.3 FUNCTIONS AND POINTERS

LEVEL–1

Q.1 Which of the following statements is correct

(i) A pointer varriable can be assigned the value of another pointer variable not necessary of the same data type.

(ii) A pointer variable can be assinged a null value

(iii) one pointer variable can be subtracted from another provided both pointers point to element of the same array.

(a) (i),(ii)

(b) (i),(ii) (iii)

(c) (ii),(iii)

(d) (i),(iii)

Q.2 Which one of the following is a correct function category?

(i) Functions with no arguments and no return values

(ii) functions with arguments and no return values

(iii) functions with arguments and return values

(a) only (iii)

(b) (i), (iii)

(c) (ii) and (i)

(d) (i),(ii),(iii)

Q.3 Which of the following statement is correct?

(i) A static variable may be either an internal type or an external type, depending on the place of declaration.

(ii) Internal static, variable are those which are declared inside a function. The scope of internal static variables extend upto the end of the functions in which they are defined.

(iii) The internal static variables are similar to auto variables, except that they remain in existence throughout the remainder of the program and initilize with zero.

(a) (i) and (ii)

(b) (i) and (iii)

(c) (ii) and (iii)

(d) (i),(ii), (iii)

Q.4 Recursive function are executed in a

(a) Last in first out order

(b) First in first out order

(c) paralled fashion

(d) all of these

Q.5 The default parameter passing mechanism is

(a) call by reference

(b) call by value

(c) call by value result

(d) none of these

Q.6 Which of the following statement is incorrect about function

(i) The use of a function avoids the need for reduntant programming of the same instructions.

(ii) The use of functions enables a programmer to build a customized library of frequently used routines or of routines containing system dependent features.

(iii) Some functions, accept information but do not return anything whereas other functions return multiple values at different times

(a) None of (i), (ii), (iii)

(b) (i), (ii), (iii)

(c) (i), (ii)

(d) (ii), (iii)

Q.7 The difference between an access function and utility function is _____

(a) There is no difference between access function and utility function.

(b) An access function is a public class member function that returns the value of one of the class's data memberss and utility function is a private class member function that is used only within the class to perform "technical" tasks.

(c) An access function is a public class member function that returns the value of the one of the class's data members and utility function is a protected class member function.

(d) An access function is a private class member functions that returns the value of one of the class's data members and utility function is a public class member function.

Q.8 Consider the statements

```
putchar (getchar ( ) ) ;
putchar (getchar ( ) ) ;
```

If

a

b

is the input, the output will be

(a) ab

(b) an error message

(c) this can't be the input

(d) none of these

Q.9 Correct way of defining a pointer to the function int*display (int*) is

(a) int * (*func_ptr) (int x) ;
func_ptr = display;

(b) int(*func_ptr) (int) ;
(*func_ptr) (int) = display (int);

(c) int(*func_ptr) (int) ; func_ptr = display ( )

(d) int*((*func_ptr) ( ) ; func_ptr = display

Q.10 The most appropriate matching for the following pairs

Group I		Group II	
A.	Indirect addressing	1.	Loops
B.	Immediate addressing	2.	Pointers
C.	Auto decrement addressing	3.	Constants

Codes:

	A	B	C
(a)	3	1	2
(b)	2	3	1
(c)	1	3	2
(d)	3	2	1

Q.11 A function's purpose is to print customer data. Which of the following is the best name for this function?

(a) last Function ( ). It is the final function called in most programs, and this name identifies the function's timing

(b) print Customer ( ). It states the function's purpose and is easy to read

(c) print customer data ( ) It states everything the function will do

(d) pcd ( ) It's short for "print customer data" and takes few keystrokes

Q.12 Pointers are of

(a) integer datatype

(b) character datatype

(c) unsigned integer datatype

(d) none of these.

Q.13 What would be the equivalent pointer expression for referring the array element a [i][j][k][l]?

(a) *(*(*(*(a + i) + j) + k) + l)

(b) ((((a + i) + j)+ k) + l)

(c) a + i + j + k + l

(d) ((((****a + i) + j) + k) + l)

Q.14 Which of the following error will be reported on compiling the program given below?

```
#include <stdio.h>
int main()
{
    int a[] = {10, 20, 30, 40, 50};
    int j;
    for (j = 0; j < 5; j++)
    {
        printf ("%d\n",*a);
        a++;
    }
    return 0;
}
```

(a) Error : 'Declaration syntax'

(b) Error : 'Expression syntax'

(c) Error : 'Lvalue required'

(d) Error : 'Rvalue required'

Q.15

```
#include <stdio.h>
void display (int);
void show (int, int);
int main()
{
    int i = 10, j = 20, k;
    display (i);
    show (i, j);
    return 0;
}
void display (int c, int d)
{
    printf("%d %d\n", c,d);
}
void show (int c)
{
    printf("%d\n", c);
}
```

(a) Error

(b) 10 20
20

(c) 10 0
10

(d) None of the above

Q.16 Which of the following functions is more versatile for positioning the file pointer in a file?

(a) rewind()

(b) fseek()

(c) ftell()

Q.17 The following loop

```
for (putchar('c'); putchar('a'); putchar('r'))
putchar('t');
```

outputs

(a) a syntax error

(b) cartrt

(c) catrat

(d) catratratratrat...

Q.18
```
#include <stdio.h>
int main ()
{
    int show();
    int (*f)();
    f = show;
    printf ("address = %d\n", f);
    (*f)();
    return 0;
}
int show()
{
    printf ("Diamonds are forever\n");
    return 0;
}
```
(a) address = 528
Diamonds are forever
(b) Error
(c) No output
(d) Diamonds are forever

Q.19 A possible output of the following program fragment
```
for (i = getchar ( ); ; i = getchar())
if (i = = 'x') break;
else putchar (i) ;
```
is
(a) mi
(b) mix
(c) mixx
(d) none of the above

Q.20 The following program
```
main ( )
{
    printf("tim");
        main ( );
}
```
(a) is illegal
(b) keeps on printing tim
(c) prints tim once
(d) none of the above

Q.21 'max' is a function that returns the larger of the two integers, given as arguments. Which of the following statements finds the largest of three given number?
(a) max (max(a, b), max(a, c))
(b) max (a, max(a, c))
(c) max (max (a, b), max(b, c))
(d) max (b, max(a, c))

Q.22
```
main ( )
{ int a = 5, b = 2 ;
 print f("%d", a +++b);
}
```
(a) results in syntax error
(b) prints 7
(c) prints 8
(d) none of the above

Q.23 The following program
```
main ( )
{ int abc ( );
   abc( ) ;
   (*abc) ( );
}
int abc ( )
{ printf ("come") ; }
```
(a) results in a compilation error
(b) prints 'come' 'come'
(c) results in a run time
(d) prints 'come' 'come'

Q.24 Choose the best answer
Prior to using a pointer variable
(a) it should be declared
(b) it should be initialized
(c) it should be both declared and initialized
(d) None of the above

Q.25 The operators > and < are meaningful when used with pointers, if
(a) the pointers point to data of similar type
(b) the pointer point to structure of similar data type
(c) the pointer point to elements of the same array
(d) none of these

Q.26 While sorting a set of names, representing the names as an array of pointers is preferable to representing the names as a two dimensional array of characters, because

(a) storage needed will be proportional to the size of the data

(b) execution will be faster

(c) swapping process becomes easier and faster

(d) none of the above

Q.27 Consider the two declarations

void *voidPrt;

char *charPrt;

Which of the following assignments are syntactically correct?

(a) voidPtr = charPtr

(b) charPtr = voidPtr

(c) *voidPtr = *charPtr

(d) *charPtr = voidPtr

Q.28 Which of the following oprations can be applied to pointer variable(s)?

(a) Division

(b) Multiplication

(c) Casting

(d) None of these

Q.29 Consider the declaration

```
int a = 5, *b = &a;
```

The statement

```
printf ("%d", a * b);
```

prints

(a) 25

(b) garbage

(c) 5 × address of b

(d) an error message

Q.30 As soon as a pointer variable is freed, its value

(a) is set to null

(b) becomes unpredictable

(c) is set to 1

(d) remains the same

Q.31 Consider the program

```
main ()
{
    int y = 1 ;
    printf ("%d", (*(char *) &x));
}
```

If the machine in which this program is executed is little-endian (meaning, the lower significant digits occupy lower addresses), then the output will be

(a) 0

(b) 99999999

(c) 1

(d) unpredictable

Q.32 Calloc(m, n); is equivalent to

(a) malloc(m*n, 0);

(b) memset(0, m*n);

(c) ptr = malloc(m*n); memset(p, 0, m*n);

(d) ptr = malloc(m*n); strcpy(p, 0);

Q.33 Reference is not same as pointer because

(a) they are one and the same

(b) reference doesn't need an explicit dereferencing mechanism

(c) a reference once established cannot be changed

(d) a reference can never be null

Q.34 One of the disadvantages of pass-by reference is that the called function may inadvertently corrupt the caller's data. This can be avoided by

(a) declaring the actual parameters constant

(b) passing pointers

(c) declaring the formal parameters constant

(d) all of the above

LEVEL–2

Q.35 Trace the output

```
# include <stdio.h)>
int funct 1 (int count);
main ()
{
int a, count;
for (count = 1: count < = 5;++ count)
{
    a = func 1 (count);
    print ("%d",a);
    }
}
funct1 (int x)
{
int y = 0;
y + = x;
return (y);
}
```

(a) 1 2 3 4 6
(b) 1 2 3 4 5
(c) 1 2 4 6 8
(d) None of these

Q.36 Trace the output

```
int x [] ={0 1,2,4};
void S1 (int *P1,  int * P2)
{
    int temp;
    temp = *P1;
    *P1=*P2;
    *P2=temp;
}
main ( )
{
int i = 1;
S1 (&i, &x[i]);
print f ("%d%d\n",i,x [i]),
}
```

(a) 1 2
(b) 2 1
(c) 2 2
(d) None of the above

Q.37 What does the following declaration means:
(i) int * ptr [10]; (ii) int (*ptr) [10];
(a) (i) pointer to the array of 10 elements
    (ii) Same as above
(b) (i) Pointer to the array of 10 elements
    (ii) array of 10 integer pointers
(c) (i) Array of 10 integer pointers
    (ii) Pointer to the array of 10 elements
(d) (i) Array of 100 pointers
    (ii) Same as (i)

Q.38 Trace the output

```
main ( )
{
  extern int fun (float),
  int a;
  a = fun (3.14);
  printf("%d", a);
}
  int fun (float aa)
  {
  float aa;
  return ((int) aa);
  }
```

(a) 3
(b) 3.14
(c) 0
(d) 3.0

Q.39 Consider the following declaration

```
(A) float f (float x)
    {
      float y;
      y = 3 * x – 1;
      return (y);
    }
(B) void f (float x)
    {
      float y;
      y = 3 * x – 1;
      printf("%f", y),
    }
```

The appropriate function call for above function is:
(a) y = f(x), f(x)
(b) f(x), y = f(x)
(c) y = f(x), y = f(x)
(d) f(x), f(x)

Q.40 What would be the values of i, x, and c. If scanf("%3d%5f%c", &i, &x, &c) is executed with input data 10S256.875ST?

(a) i = 10, b = 256.8, c = T

(b) i = 010, b = 256.87, c = 5

(c) i = 100, b = 256.87, c = T

(d) i = 10, b = 56.875, c = T

Q.41 Match the following:

Group I		Group II	
1.	a pointer to an array of 8 floats	a.	float *(*f) ( )
2.	a pointer to an array of 8 pointer to float	b.	float (*f) ( )
3.	a pointer to a function that returns a float	c.	float (*a) [8]
4.	a pointer to a function that returns a pointer to a float	d.	float *(*a) [8]

(a) 1 – c, 2 – d, 3 – a, 4 – b

(b) 1 – d, 2 – c, 3 – a, 4 – b

(c) 1 – c, 2 – d, 3 – b, 4 – a

(d) 1 – d, 2 – c, 3 – b, 4 – a

Q.42 What is the output of the following 'C' program?

```
main ( )
{
    const int x = 5;
    int*ptrx;
    ptrx = &x;
    *ptrx = 10;
    printf("%d", x);
}
```

(a) Error

(b) 5

(c) 10

(d) Garbage value

Q.43 What will be the value returned by the following function, when it is called with 11?

```
recur (int num)
{
    if((num/2)!=0) return(recur (num/2)
    *10 + num%2);
    else return 1;
}
```

(a) 11

(b) 111

(c) Function does not return any value, because it goes into an infinite loop

(d) none of these

Q.44 Consider the program segment:

```
main ()
{
    int n =2,*ptr;
    ptr = & n;
    n*=3;
    printf("d\n",(*ptr)*(*ptr);
}
```

The above program segments prints.

(a) 6

(b) 36

(c) 4

(d) None of the above

Q.45 The output of the following program.

```
main ()
{static int x[]= {1, 2, 3, 4, 5, 6, 7, 8};
int i;
for (i=2, i <6;++i)
x [x[i]]=x[i];
for (i=0; i <8;++i)
printf ("%d",x[i]);
```

(a) is 1 2 3 3 5 5 7 8

(b) is 1 2 3 4 5 6 7 8

(c) is 8 7 6 5 4 3 2 1

(d) is 1 2 3 5 4 6 7 8

Q.46 The following program:

```
main ()
{
    int a=4;
    void change(int a);
    printf("%d",a);
}
    void change(int a)
{
    printf("%d",++a);
}
```

outputs

(a) 55

(b) 5

(c) 4

(d) 44

Q.47 Which of the following is the correct output for the program given below?

```
#include <stdio.h>
int main ()
{
    int a = 0, b = 1, c = 3;
    * ((a)? &b : &a) = a ? b : c ;
    printf("%d %d %d\n", a, b, c);
    return 0;
}
```

(a) 0 1 3

(b) 1 2 3

(c) 3 1 3

(d) 1 3 1

Q.48 Select any other way of writing the following expression such that 30 is used only once?

```
a <= 20? b = 30 : c = 30;
```

(a) *(a <= 20)? & b; & c = 30;

(b) a <= 20? (b:c) = 30;

(c) *((a <=20)? & b : & c) = 30;

(d) None of the above

Q.49 If the binary equivalent of 5.375 in normalised form is 0100 0000 1010 1100 0000 0000 0000 0000, what will be the output of the following program?

```
#include <stdio.h>
int main()
{
float a = 5.375;
char *p ;
int i ;
p = (char *) & a;
for (i = 0; i <= 3; i++)
    printf ("%02x\n", (unsigned char) p[i]);
return 0;
}
```

(a) 40 AC 00 00

(b) 04 CA 00 00

(c) 00 00 AC 40

(d) 00 00 CA 04

Q.50 Point out the error, if any, in the following function.

```
#include <stdio.h>
int main ()
{
    int f(int) ;
    int b;
    b = f (20);
    prinft ("%d\n", b);
    return 0;
}
int f (int a)
{
    a > 20? return (10) : return (20);
}
```

(a) 10

(b) 20

(c) error

(d) No result

Q.51 Point out the error, if any, in the following code.

```
#include <stdio.h>
#define SI(p, n, r) float si ; si = p * n* r /100;
int main()
{
    float p = 2500, r = 3.5;
    int n = 3;
    SI (p, n, r);
    SI (1500, 2, 2.5);
    return 0;
}
```

(a) multiple declaration of si

(b) error in si = p *n*r/100

(c) no error.

(d) Run-time correct

Q.52 What will be the output of the following program assuming that the array begins at location 1002?

```
#include <stdio.h>
int main()
{
    int a[2][3][4] = {
                        {
                          1, 2, 3, 4,
                          5, 6, 7, 8,
                          9, 1, 1, 2
                        }
                        {
                          2, 1, 4, 7,
                          6, 7, 8, 9,
                          0, 0, 0, 0
                        }
                    };
    printf ("%u%u%u%d\n", a, *a, **a, ***a);
    return 0 ;
}
```

(a) 1 1002 1002 1002

(b) 1002 1 5 9

(c) 1002 1002 1002 1

(d) None of the above

Q.53 Will the following program compile ; if yes provide the O/P:

```
#include <stdio.h>
int main()
{
int a = 10,*j;
void *k;
j = k = & a;
j ++ ;
k ++ ;
printf ("%u%u\n", j, k);
return 0;
}
```

(a) 11 11

(b) 10 11

(c) error

(d) 10 10

Q.54 Which of the following is the correct output for the program given below?

```
#include <stdio.h>
int main()
{
    cha* str;
    str = "%d\n";
    str++ ;
    str++;
    printf ( str -2, 300);
    return 0;
}
```

(a) No output

(b) 30

(c) 3

(d) 300

Q.55 Which statement will you add to the following program to ensure that the program outputs "Mumbadevi" on execution?

```
#include < stdio.h>
int main()
{
    char s []= "Mumbadevi";
    char t[25];
    char *ps,*pt;
    ps = s;
    pt = t ;
    while (*ps)
        *pt++ = *ps++;
/* add suitable statement here*/
printf ("%s\n", t);
return 0;
}
```

(a) *ps = '\0';
(b) *pt = '\0';
(c) ps = '\0';
(d) pt = '\0';

Q.56 What will be the output of the program (myprog) if it is executed from the command-line as shown below?

```
myprog 1 2 3
/* myprog. c*/
#include < stdio.h>
#include <stdlib.h>
int main(int argc, char*argv[])
{
    int i, j = 0 ;
    for (i = 0; i < argc ; i++)
        j = j + atoi (argv[i]);
    printf ("%d\n", j);
    return 0;
}
```

(a) 123
(b) 0
(c) Error
(d) "123"

Q.57 Which of the following statements are correct about the code given below :

```
int main (int ac, char*av[])
{
}
```

(A) 'ac' contains count of arguments supplied at command-line.
(B) av[] contains addresses of arguments supplied at command-line.
(C) In place of 'a'c and 'av', 'argc' and 'argv' should be used.
(D) 'ac' contains address of an integer whose value is equal to the count of arguments supplied at command-line.
(E) The variables 'ac' and 'av' are always local to main().
(F) The maximum combined length of the command-line arguments including the spaces between adjacent arguments should be 128 characters.

(a) A, B, C
(b) A, B, E
(c) C, E, F
(d) B, E, F

Q.58 To correctly add three floats supplied at command-line which statement will you add to the code given below?

```
#include <stdio.h>
int main (int argc, char*argv[])
{
    int i ;
    float j;
    for (i = 0; i < argc ; i++)
{
    /* add suitable statement here*/
}
printf ("%f\n", j);
return 0;
}
```

(a) j = j + atoi(argv[i]);
(b) j = j + argv[i](atoi);
(c) j = j + atof(argv[i]);
(d) Error in other statements.

Q.59 What will be the O/P of following program :

```
#include <stdio.h>
#include <stdarg.h>
void display (int num, ...);
int main()
{
    display (4, 'A', 'a', 'b', 'c');
    return 0;
}
void display (int num, ....)
{
    char c, int j;
    va_list ptr;
    va_start (ptr, num);
    for (j = 1; j <= num; j++)
    {
        c = va_arg (ptr, char);
        printf ("%c", c);
    }
    printf ("\n");
}
```

(a) 4 A a b c

(b) 4 A a b

(c) A a b c

(d) None of the above

Q.60 The statement

```
if (my Ptr ! = NULL)
    *myPtr = NULL ;
else
    *myPtr = NULL ;
```

has the same effect as the statement(s)

(a) if (myPtr) *myptr = NULL;
else *myptr = NULL;

(b) *myPtr = NULL;

(c) if (! myPtr) *myPtr = NULL ;
else *myPtr = NULL;

(d) if (myPtr = = NULL) *myPtr = NULL;
else *myPtr = NULL,

Q.61

```
#include <stdio.h>
int main ()
{
    int i, fun1(), fun2(), fun3();
    int(*f[3])();
    f[0] = fun1;
    f[1] = fun2;
    f[2] = fun3;
    for (i = 0; i <= 2; i++)
        (*f[i])();
    return 0;
}
int fun1()
{
    printf ("Hail\n");
    return 0;
}
int fun2()
{
    printf ("the\n");
    return 0;
}
int fun3()
{
    printf ("viruses!\n");
    return 0;
}
```

(a) Runtime error

(b) Compilation error

(c) Hail
the
viruses!

(d) 0 1 2

Q.62
```
#include <stdio.h>
int main ()
{
    static int a[] = {2, 4, 6, 8, 10};
    int i;
    for (i = 0; i <= 4; i++)
    {
        *(a + i) = a[i] + i[a];
        printf ("%d\n", *(i + a));
    }
    return 0;
}
```
(a) 4
8
12
16
20

(b) 2
4
6
8
10

(c) 1
2
3
4
5

(d) Error

Q.63 Consider the declarations

char first (int (*) (char, float));

int second (char, float);

Which of the following function invocation is valid?

(a) first (*second);

(b) first (&second);

(c) first(second);

(d) None of the above

Q.64 A function q that accepts a pointer to a character as argument and returns a pointer to an array of integer can be declared as

(a) int (* q(char*))[]

(b) int * q(char *)[]

(c) int (*q) (char *) []

(d) None of the above

Q.65 a → b is syntactically correct if

(a) a and b are structures

(b) a is a structure and b is a pointer to a structure

(c) a is a pointer to a structure and b is a structure

(d) a is a pointer to a structure in which b is a field

Q.66 ftell

(a) is a function

(b) gives the current file position indicator

(c) can be used to find the size of a file

(d) is meant for checking whether a given file exists or not

Q.67 The statement fseek (fp, 0L, 0) ; - if syntactically correct, means

(a) fp is a file pointer

(b) position the read-write-head at the start of the file

(c) position the read-write-head at the end of the file

(d) erase the contents of the file

Q.68 Consider the c code:

```
main ( )
{ char *p ;
  p = "Hello" ;
   print f ("%c\n", *&*p) ;
}
```

The Output will be:

(a) ello

(b) Hello

(c) H

(d) Error

Q.69 main ( )

```
{ Static int a[3][3] = {1,2,3,4,5,6,7,8,9};
    int i,j;
    Static *p[] = {a, a+1, a+2};
    for (i = 0 ; i < 3 ; i++)
{
    for (j = 0 ; j < 3 ; j++)
    print f("%d\t %d\t %d\t %d\n",
        *(*(p + i)+j), *(*(j + p)+i), *(*(i + p)+j),
        *(*(p + j)+i);
    }
{
```

(a)
1 1 1 1
2 2 2 2
3 3 3 3
4 4 4 4

(b)
1 1 1 1
2 4 2 4
3 7 3 7
4 2 4 2

(c)
1 2 3 4
3 5 7 5
2 4 2 4
4 2 4 2

(d)
1 2 3 4
5 6 7 8
9 0 1 2
3 4 5 6

Q.70 Assume that the random number generating function – rand( ), returns an integer between 0 and 10000 (both inclusive). To randomly generate a number between a and b (both inclusive), use the expression

(a) (rand( ) % (b – a + 1)) + a

(b) (rand( ) % (b – a)) + a

(c) rand( ) % (b – a)

(d) (rand( ) % a) + b

LEVEL–3

Q.71 Trace the output

```
# include <stdio.h>
int fun1 (int a);
int fun2 (int a);
main ( )
    {
      int a = 0, b = 1, count;
      for (count = 1; count <= 5; ++count)
          {
            b + = fun1 (a) + fun2 (a);
            printf("%d", b);
          }
    }
int  fun1 (int a)
{
  int b;
  b = fun2 (a);
  return (b);
}
int fun2 (int a)
{
  static int b = 1;
  b + = 1;
  return (b +a);
}
```

(a) 6 15 18 24 28

(b) 6 15 28 45 66

(c) 6 15 30 45 60

(d) 6 15 24 33 41

Q.72 Trace the output

```
main ()
{
void f1 (int a) ;
int i;
for (i=0;i<=5;i++)
f1(i);
}
void f1 (int a)
{
static int k;
int p = 1;
while (k<4)
{
p*=a
k++;
}
print ("%d",p);
}
```

(a) 0 1 1 1 1 1

(b) 0 1 2 3 4 5

(c) 0 1

(d) 0 1 4 9 16 25

Q.73 Consider the following code:

```
main ( )
{ char *p = "hai friends", *p1;
  p1 = p ;
  while (*p! = \0") ++ *p++ ;
   printf ("%s%s",p,p1),
}
```

(a) hai friends

(b) ibj ab hai friends

(c) p hai friends

(d) ibj ! gsjfoet

Q.74 Output of the following program is:

```
# include <stdio.h>
int a = 100, b = 200;
int func1 (int x);
main ( )
{
    int count, C ;
    for (count = 1; count <= 5; ++ count)
    {
        C = 4 * count * count ;
        printf("%d", func1(C));
    }
}
int funct1 (int x)
{
    int C ;
    C = (x < 50) ? (a + x) : (b – x);
    return (C) ;
}
```

(a) 104 106 116 120

(b) 104 116 136 136

(c) 104 110 116 126

(d) 105 116 136 156

Q.75 Programme given below does

```
main()
{
    int x,y;
    int*x pointer
    int temp=3;
    x=5* (temp+5)
    x pointer =& temp;
    y=5*(*x pointer+5);
    printf("x=%d\n",x);
    printf("y=%d\n",y);
}
```

(a) x = 40, y = 40

(b) x = 80, y = 80

(c) x = 20, y = 20

(d) none of the above

Q.76 What are the outputs of the following program?

```
main ( )
{
int u = 1;
int v = 3;
void funct1 (int u, int v);
void funct2 (int *pu, int *pv);
    printf("u = %d", v = %d", u, v);
    funct1(u, v);
    printf("u = %d", v = %d", u,v);
    funct2(&u, &v);
    printf("u = %d", v = %d", u,v);
}
funct1 (int u, int v)
{
u = u + v;
v = u – v;
u = u – v;
}
funct2 (int *pu, int *pv)
{ *pu = *pu + *pv;
*pv = *pu – *pv;
*pu = *pu – *pv; }
```

(a) u = 1 v = 3 u = 1 v = 3 u = 3 v = 1
(b) u = 1 v = 3 u = 3 v = 1 u = 1 v = 3
(c) u = 1 v = 3 u = 3 v = 1 u = 3 v = 1
(d) u = 1 v = 3 u = 1 v = 3 u = 1 v = 3

Q.77 Consider the following program

```
# include <stdio.h>
main ( )
{
    int x = 105, y = 0, z = 30;
char k[5];
strcpy (k, "FAIL");
if (x/20 = = 5 && (x*y > 100 | | y + z * 2 < 75))
   if (x << 2 <= 256)
                strcpy (k, "PASS");
          else strcpy (k, "pass"),
       else strcpy (k, "fail");
}
```

The value of k on execution of the above program will be

(a) pass
(b) FAIL
(c) PASS
(d) None of these

Q.78 The following program prints the elements of 2*2 matrix.

```
#include <stdio.h>
#include <conio.h>
main ()
    {
        int a [20][20],i,j;
        a[1][1]=2;a[1][2]=3;a[2][1]=4;a[2]
        [2]=6;
        for(i=1;i<=2;++i)
        for (j=1;j<=2;++j)
    {
        printf("%d,a[i][j]);
        }
        printf("\n");
    }
}
```

(a) $\begin{bmatrix} 1 & 3 \\ 2 & 4 \end{bmatrix}$

(b) $\begin{bmatrix} 2 & 3 \\ 1 & 2 \end{bmatrix}$

(c) $\begin{bmatrix} 2 & 3 \\ 4 & 6 \end{bmatrix}$

(d) none of these

Q.79

```
Main ()
    {
        int i = 5
        int a;
        a=fun(i)
        printf("%d",a);
    }
    fun (int x)
    {
    if (x>=4)
        x=x*x;
    else
        x=x*2
    return (x);
}
```

(a) 6
(b) 12
(c) 16
(d) none of these

Q.80
```
# include <stdio.h>
#include<conio.h>
void ext fun ();
int num;
void main ()
{
    extern int num;
        //External variable declaration
    clrscr ();
    num = 10
    printf("%d\n,num);
        ext fun ();
        getch ();
            }
        void ext fun ()
            {
            extern int num;// External variable
            declaration
            num=num+50;
            printf("%d,num);
            }
```
(a) 5,55
(b) 10,60
(c) 10,70
(d) none.

Q.81
```
#include <stdio.h>
int main ()
{
    static int a[] = {0, 1, 2, 3, 4};
    static int *p[] = {a, a+1, a+2, a+3, a+4};
    int **ptr;
    ptr = p;
    **ptr++;
    printf ("%d %d %d\n", ptr-p, *ptr-a, **ptr);
    *++*ptr;
    printf ("%d %d %d\n", ptr-p, *ptr-a, **ptr);
    ++**ptr;
    printf ("%d %d %d\n", ptr-p, *ptr-a, **ptr);
    return 0;
}
```
(a) 1 1 1
    2 2 2
    3 3 3
(b) 1 1 1
    1 1 2
    2 3 3
(c) 1 1 1
    1 2 2
    1 2 3
(d) Error message:
    L value required in function main.

Q.82 Trace the output
```
# include <stdio.h>
typedef union
{   int i;
    float f;
}   udef;
void funct (udef);
main ( )
{   udef u;
    u.i = 100;
    u.f = 0.5;
    funct(u);
    printf("%d%f\n", u.i, u.f);
}
void funct (udef u)
{   u.i = 200;
    printf("%d%f\n", u.i, u.f);
    return;
}
```
(a) garbage garbage
    garbage 0.500000
(b) 200 garbage
    garbage0.500000
(c) garbage garbage
    garbage garbage
(d) 200 garbage
    garbage garbage

GATE QUESTIONS

Q.83 In which of the following case (s) is it possible to obtain different results for callby reference and call by name parameter passing? **[GATE 1989]**

(a) Passing an expression as a parameter
(b) Passing an array as a paramter
(c) Passing a pointer as a parameter
(d) Passing an array element as a parameter

Q.84 The most appropriate matching for the following pairs

Group I		Group II	
X.	m = malloc (5);m = NULL;	1.	using dangling pointers
Y.	free (n);n–> value = 5;	2.	using initialized pointers
Z.	char * P;*P= 'a' ;	3.	lost memory

is **[GATE 2000]**

[1-Mark]

(a) X–1,Y–3,Z–2

(b) X–2,Y–1,Z–3

(c) X–3,Y–2,Z–1

(d) X–3,Y–1,Z–2

Common Data For Questions 85 & 86:

The following program fragment is written in a programming language that allows global variables and does not allow nested declarations of functions.

```
global int i = 100, j = 5;
void P (x) {
        int i = 10;
        print (x+10);
        i = 200;
        j = 20;
        print (x) ;
}
main () {P(i+j);}
```

Q.85 If the programming language uses static scoping and call by need parameter passing mechanism, the values printed by the above program are **[GATE 2003]**

[2-Marks]

(a) 115,220

(b) 25,15

(c) 25,220

(d) 115, 105

Q.86 If the programming language uses dynamic scoping and call by need parameter passing mechanism, the values printed by the above program are **[GATE 2003]**

[2-Marks]

(a) 115,220

(b) 25,220

(c) 115, 105

(d) 25,15

Q.87 Assume the following C variable declaration

int *A [10], B [10] [10];

Of the following expressions

(I) A [2] (II) A [2] [3] (III) B [1] (IV) B [2] [3]

which will not give compile time errors if used as left hand sides of assignment statements in a C program? **[GATE 2003]**

[1-Mark]

(a) I, II and IV only

(b) II and IV only

(c) II, III and IV only

(d) I, III, and IV only

Q.88 Consider the C program shown below

```
# include < stdio. h>
# define print (x) print f ("%d",x)
int x;
void Q (int z ) {
        z+=x;
        print (z);
}
 void p (int *y) {
        int x  = *y+2;
        Q(x);*y=x–1;
        print (x)
}
main (void) {
        x=5;
        p(&x);
        print (x);
}
```

The output of this program is [GATE 2003] [2-Marks]

(a) 12 7 6
(b) 14 6 6
(c) 22 12 11
(d) 7 6 6

Q.89 Consider the following C function.

```
float f (float x, int y)
{
    float p, s; int i ;
    for (s=1, p=1, i=1;i<y;i++)
    {
        p*=x/i;
        s+=p;
    }
return s;
}
```

For large values of y, the return value of the function f best approximates [GATE 2003] [1-Mark]

(a) X^y
(b) e^x
(c) In (1+x)
(d) X^x

Q.90 Consider the following C function

```
void swap (int a,int b)
    { int temp;
    temp    = a ;
    a       = b ;
    b       = temp ;
    }
```

In order to exchange the values of two variables x and y. [Gate 2004] [1-Mark]

(a) swap (x, y) cannot be used as the parameters are passed by value
(b) swap (x, y) cannot be used as it does not return any value
(c) call swap (x, y)
(d) call swap (&x, &y)

Q.91 Consider the following C program segement:

```
char p[20];
char * s = "string" ;
int length = strlen (s) ;
for     (i = 0 ; i < length; i ++)
        p[i] = s[length – i] ;
printf("%s", p);
```

The output of the program is [GATE 2004] [2-Marks]

(a) no output is printed
(b) gnirt
(c) gnirts
(d) string

Q.92 Consider the following C program which is supposed to compute the transpose of a given 4 × 4 matrix M. Note that, there is an X in the program which indicates some missing statements. Choose the correct option to replace X in the program.

```
#include <stdio.h>
#define ROW 4
#define COL 4
int M[ROW][COL] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10,
11, 12, 13, 14, 15, 16};
main ( )
{
    int i, j, t;
    for (i=0;i<4,++i)
        {
            X
        }
    for  (i=0;i<4;++i)
        for (j=0;j<4;++j)
            printf("%d", M[i][j]);
            }
```

[IT-GATE 2004] [2-Marks]

(a) for (j=0;j<4;++j){
t = M[i][j];
M[i][j] = M[j][i];
M[j][i] = t;
}

(b) for(j=0;j<4;++j){
M[i][j] = t;
t = m[j][i];
M[j][i] = M[i][j];
}

(c) for (j=i;j<4;++j){
t = M[j][i];
M[i][j] = M[j][i];
M[j][i] = t;
}

(d) for (j=i;j<4;++j){
M[i][j] = t;
t = M[j][i];
M[j][i] = M[i][j];
}

Q.93 Choose the correct option to fill ?1 and ?2 so that the program prints an input string in reverse order. Assume that the input string is terminated by a new line character.

```
#include <stdio.h>
void wrt_it(void);
int main (void)
{
    printf("Enter Text");
    printf("/n");
    wrt_it( );
    printf("/n");
    return 0;
}
void wrt_it(void)
{
    int c;
    if (?1)
        wrt_it( );
    ?2
}
```

[IT-GATE 2004]

[2-Marks]

(a) ?1 is getchar( ) !='/n'
?2 is getchar(c);

(b) ?1 is (c=getchar( ) !='/n'
?2 is getchar(c);

(c) ?1 is c !='/n'
?2 is putchar(c);

(d) ?1 is (c=getchar( )) !='/n'
?2 is putchar(c);

Q.94 Consider the following C-program

```
double foo(double);/*Line1*/
int main ( ) {
double da, db;
//input da
db = foo (da);
}
double foo(double a){
return a;
}
```

The above code complied without any error or warning. If Line1 is deleted, the above code will show **[GATE 2005]**

[2-Marks]

(a) Compiler errors

(b) Some complier-warning due to type-mismatch eventually leading to unitended results

(c) No compile warning or error

(d) Some complier-warning not leading to unintended results

Q.95 Consider these two functions and two statements S1 and S2 about them.

```
S1 :  int work1(int*a, int i, int j)
      {
            int x = a[i + 2];
            a[j] = x + 1;
            return a[i + 2] - 3;}
      }
S2:   int work2(int*a, int i, int j)
      {
            int t1 = i + 2;
            int t2 = a[t1];
            a[j] = t2 + 1;
            return t2 - 3;
      }
```

S1: The tranformation from work 1 to 2 is valid, i.e. for any program state and input arguments, work2 will compute the same output and have the same effect on program state as work1

S2 All the tranformations applied to work1 to get work2 will always improve the performance (i.e. reduce CPU time) of work2 compared to work1

[GATE 2006]

[2-Marks]

(a) S1 is true and S2 is true

(b) S1 is true and S2 is false

(c) S1 is false and S2 is false

(d) S1 is false and S2 is true

Q.96 Consider the C code to swap two integers and these five statements: the code

```
void swap (int*px, int*py) {
        *px = *px - *py;
        *py = *px + *py;
        *px = *py - *px;
}
```

S_1: will generate a compilation error

S_2: may generate a segmentation fault at runtime depending on the arguments passed

S_3: Correctly implements the swap procedure for all input pointers referring to integers stored in memory locations accessible to the process

S_4: implements the swap procedure correctly for some but not all valid input pointers

S_5: may add or subtract integers and pointers

[Gate 2006]

[2-Marks]

(a) S_2 and S_5

(b) S_2 and S_4

(c) S_1

(d) S_2 and S_3

Q.97 Consider the following C function:

```
int f(int n){
    static int r = 0;
if  (n <= 0) return 1;
if  (n > 3)
    {r = n;
    return f(n - 2) + 2;
}
return f(n - 1) + r;
}
```

What is the value of f(5)? **[Gate 2007]**

[2-Marks]

(a) 18

(b) 7

(c) 5

(d) 9

Q.98 What is printed by the following C program?

```
int f(intx, int *py, int **ppz)
{
    int y, z;
    **ppz += 1; z = **ppz;
    *py += 2; y = *py;
    x +=3;
    return x + y +z;
}
```

```
void main ( )
{
    int c, *b, **a;
    c = 4; b = &c; a = &b;
    printf("%d", f(c, b, a));
}
```

[Gate 2008]
[2-Marks]

(a) 18
(b) 22
(c) 19
(d) 21

Q.99 Choose the correct option to fill ?1 and ?2 so that the program below prints an input string in reverse order. Assume that the input string is terminated by a newline character.

```
void reverse (void) {
    int c;
    if (?1) reverse ( );
    ?2
}
main ( ) {
    printf("Enter Text") ; printf("\n");
    reverse ( ); printf("\n");
}
```

[Gate 2008]
[2-Marks]

(a) ?1 is ((c = getchar ( )) ! = '\n')
?2 is putchar (c) ;
(b) ?1 is (c ! = '\n')
?2 is putchar (c) ;
(c) ?1 is (getchar ( ) ! = '\n')
?2 is getchar (c);
(d) ?1 is (c = getchar ( )) ! = '\n')
?2 is getchar (c)

Q.100 Consider the program below:

```
#include<stdio.h>
int fun (int n, int*f_p) {
    int t, f;
    if (n <= 1) {
        *f_p = 1 ;
        return 1;
}
    t = fun (n - 1, f_p) ;
    f = t + *f_p ;
    *f_p = t ;
    return f;
}
int main ( ) {
    int x = 15 ;
    printf("%d\n", fun (5, &x));
    return 0;
}
```

The value printed is: [Gate 2009]
[1-Mark]

(a) 6
(b) 15
(c) 8
(d) 14

ANSWER KEY

1	c	2	d	3	d	4	a	5	c
6	a	7	b	8	d	9	a	10	b
11	b	12	d	13	a	14	c	15	c
16	b	17	d	18	a	19	a	20	b
21	a, c, d	22	b	23	b	24	c	25	c
26	a, b, c	27	a	28	c	29	d	30	d
31	c	32	c	33	b, c, d	34	c	35	b
36	d	37	c	38	a	39	a	40	a
41	c	42	a	43	b	44	b	45	a
46	c	47	c	48	c	49	c	50	c
51	a	52	c	53	c	54	d	55	b
56	b	57	b	58	c	59	c	60	all
61	c	62	a	63	c	64	a	65	d
66	a, b, c	67	a, b	68	c	69	b	70	a
71	b	72	a	73	d	74	b	75	a
76	a	77	a	78	c	79	d	80	b
81	c	82	b	83	b	84	d	85	a
86	b	87	b	88	a	89	b	90	a
91	a	92	a	93	d	94	b	95	a
96	d	97	a	98	a	99	a	100	c

SOLUTIONS

S.1 (c)

(i) is incorrect statement, corrected (i) statement is –A pointer variable can be assigned the value of another pointer variable provided both pointers points to objects of the same data type.

S.2 (d)

A funcions depending on whether arguments are present or not and whether a value is returned or not **(i),(ii) and (iii) are categories of functions.**

S.3 (d)

(i),(ii), (iii) are correct statements.

Due to (iii), internal static variables can be used to retrain values between function calls.

S.4 (a)

A recusive function uses the stack. Hence the LIFO order i.e. last in first out order is there.

S.5 (c)

Which means a function will be manipulating a copy of the local variable, passed as argument. So, any change will be local and hence will not be reflected in the calling routine.

S.6 (a)

(i), (ii) and (iii) are characteristics of function.

S.7 (b)

(b) ⇒ is the difference between an access function and utility function.

S.8 (d)

The input is actually a\nb. Since we are reading only two characters, **only a and \n will be read and printed.**

S.12 (d)

Pointers are actually addresses. Though the address will be an integer, it is not of integer data type. Both have different set of operations defined on them, e.g., integer addition is different from pointer addition.

S.15 (c)

Output

10 0

10

Explanation:

When **display( )** is called one argument is passed to it, whereas while defining **display( )** two arguments are used. Would it result into an error? No, since the compiler accepts a mismatch in the number of arguments being passed and collected. If this is so, which of the two variables would collect the value being passed to **display( )**? **c** collects the value being passed, hence the **printf( )** prints out the value of **c** as 10, whereas **d** is printed as 0.

What if we pass two arguments and collect them in one variable? This is what is being done in the call to function **show( )**. Here the first value gets collected in the variable **c,** whereas the second value gets ignored. The value collected in **c**, i.e., 10 then gets printed.

S.17 (d)

Execution process of for loop

i.e., for (Initialization; condition; increment/ decrement)

{body of for loop}

S.18 (a)

Output

address = 528

Diamonds are forever

Explanation:

This program demonstrates the use of pointers to functions, **f**, which is declared to be a pointer to the function **show()**, is assigned the address of this function. The address of a function can be obtained by simply mentioning the name of the function. The **printf()** in **main()** prints this address, which may be something other than 528 when you run the program.

The last statement in **main()** calls the function **show()**. Instead of calling the function by name, we make use of the pointer to the function. As a result control goes to **show()**, where the **printf()** prints "Diamonds are forever".

S.20 (b)

This involves recursion-main ( ) calling itself. So, it keeps on printing tim

S.21 (a, c, d)

All the three function calculate greater of 3 numbers assignment from right.

S.22 (b)

The compiler will tokenize a +++b as a, ++, +, b. So, **a+++b is equivalent to a+++ b, which evaluates to 7.**

S.23 (b)

The function 'abc' can be invoked as abc () or (*abc) (). **Both are two different ways of doing the same thing.**

S.24 (c)

Using a pointer variable, without initializing it, will be disastrous, as it will have a garbage value.

S.29 (d)

Since 'a' is an integer and 'b' is a pointer, they can't be multiplied.

S.34 (c)

If the formal parameters are declared constant, the compiler will not allow any changes to be made. So the question of - inadvertently corrupting the caller's data, does not arise at all.

S.35 (b)

Function in called inside the loop as

a = funct1 (count)

and then the value of a is printed

The value of count is passed 5 times from 1 to 5. and each time the same value is returned by the statement retrun(y); as int y+=x statements put the value of count into x and finally into they.

Hence the output produced will be 1 2 3 4 5.

S.36 (d)

By S1 (&i, & x[i]), we are passing address, and that will reflect the swapping values in the main (). But when we are printing the values we get i = 1 and due to i = 1, we get x[i] = x[1] =1. **Therefore we get the output as 11.**

S.37 (c)

There is difference between * ptr [10] and (*ptr)[10]

Since* has lower precedence than [ ] **int * ptr [10] declares ptr as an array of 10 pointers while int (*ptr) [10] declares ptr as a pointer to an array of 10 element.**

S.38 (a)

a= fun.(3.14); will call the function fun(aa) which is having the statement

retrun (int) aa;

⇒ 3.14 is copied into aa and (int) aa type casts the value aa into int. Hence int value will be returned.

⇒ int corresponding to 3.14 (which is 3) is returned.

S.39 (a)

The function call for given code is

(A) y = f(x) ⇒ Since we are returning a y from function f

(B) f(x) ⇒ Since we are not returning anything from function f.

S.40 (a)

Input ⇒ 10_256 875_T·

%3d ⇒ 10 ⇒ i; 3 spaces are reserved

%5f ⇒ 256.8 ⇒ x; 5 spaces are reserved

%c ⇒ T ⇒ c; the next value goes to c.

S.42 (a)

x is a constant. Hence its value can't be changed. Hence if we try to change the value of x by any means **(here by the use pointer), it will producer an error.**

S.43 (b)

rec (11) → rec (s)*10+1

= (rec(2)*10+1)*10+1

= 100*rec(2)+10+1

= 100*rec (2)+11

= 100*1+11 = **111**

S.44 (b)

∵ n* = 3

⇒ n = n * 3

= 2 * 3 = 6

So, *p + r = 6

(*p + r) * (*p + r)

= **36**

S.45 (a)

x[x[2]] = x[2]

x[3] = 3

So sequence 1, 2, 3, 3, 4, 5, 6, 7, 8

Similarly for 3, 4, 5,

We get the sequence is 1, 2, 3, 3, 5, 5, 7, 8.

S.46 (c)

A change (a), prints 5 but the value of 'a' main () is still 4. **So main () will print 4.**

S.47 (c)

*((a)? & b : & a) = a ? b : c;

or * ((0)? & b : & a) = a? b : c;

or * & a = 0 ? b : c;

or a = c;

or a = 3

so, a = 3, b = 1, c = 3

S.50 (c)

return statement cannot be used as shown with the conditional operators. Instead the following statement can be used :

return (a > 20? 10 : 20);

S.51 (a)

This code will generate the following error:

-Multiple declarations for si

To remove this error, modify the macro as shown below :

#define SI(p, n, r) p* n* r/100

S.53 (c)

An error would be reported in the statement k++ since arithmetic on void pointers is not permitted unless the void pointer is approximately typecasted.

S.56 (b)

When atoi() tries to convert argv[0] to a number it cannot do so (argv[0] being a file name) and hence returns a zero.

S.58 (c)

'atof' function converts a string to a floating point. Hence the statement

j = j + atof(argv[i]);

at i = 0, add first floating value to j and so on, results into the addition of three floating point values through commandline.

S.59 (c)

No error. It will output A a b c.

S.60 (a, b, c, d)

Pointer points to address of variable, and *myPtr represents value of variable.

S.61 (c)

Output

Hail

the

viruses!

Explanation:

Here we are dealing with an array of pointers to functions. To begin with, in the elements of the array **f[]** the addresses of functions **fun1(), fun2()** and **fun3()** are stored. Next, with the **for** loop, calles are made to each of these functions making use of their addresses. The first call results in "Hail" being outputted. Now, when **i** is 1, **fun2()** is called, and "the" is printed. Finally, for **i** equal to 2, **fun3()** gets called, and the **printf()** here displays "viruses!".

S.62 (a)

Output

4

8

12

16

20

Explanation:

Imbibe the following two facts and the program becomes very simple to understand:

- Mentioning the name of the array gives the base address of the array.
- Array elements are stored in contiguous memory locations. On adding 1 to the address of an integer, we get the address of the next integer.

With those facts clearly laid out, let us now try to understand the program. Remember that internally C always accesses array elements using pointers. Thus, when we say **a[i]**, internally C converts it to ***(a + i)**, which means value of i integer from the base address. Now, if the expression **a[i]** is same as ***(a + i)** then ***(i + a)** must be same as **i[a]**. But ***(a + i)** is same as ***(i + a)**. Therefore **a[i]** must be same as **i[a]**.

Thus **a[i]**, ***(a + i)**, ***(i + a)** and **i[a]** refer to the same element

- the i^{th} element from the base address.

Therefore the expression used in the **for** loop, ***(a + i) = a[i] + i[a]** is nothing but **a[i] = a[i] + a[i]**. Thus all that is done in the **for** loop is each array element is doubled and then printed out through **printf()**.

S.63 (c)

The first declaration means, first is a function (returning a character), whose only argument is, a pointer to a function that takes a character and float as arguments and returns an integer. The name of a function can be used as the starting address of the function (i.e., a pointer to it).

S.66 (a, b, c)

Feature of ftell()

S.67 (a, b)

fseek (fp, 0L, 0) takes file pointer to start of the file that is 0L and conditions of read and write.

S.68 (c)

* is a dereference operator & is a reference operator. They can be applied any number of times provided it is meaningful.

Here p points to the first character in the string "Hello". *p dereferences it and so it value is H. Again '&' references it to an address and '*' dereferences it to the value **H.**

S.70 (a)

An example will help you understand. Let a be 6 and b be 10. There are 5 numbers between 6 to 10 (both inclusive). This is b – a + 1 (10 – 6 + 1). rand( ) % (b – a + 1) returns an integer form 0 to (b – a). To make it a to b, add a, giving **rand( ) % (b – a + 1) + a.**

S.71 (b)

a = 0, b = 1

In for loop condition is true

b = b + fun1 (0) + fun2 (0)

In fun1 (0), b = fun2 (a) = fun2 (0)

In fun2 (0), b = 1

b = b + 1 = 1 + 1 = 2

return (2 + 0)

In fun2 (0), b = 2 (due to static)

b = b + 1 = 2 + 1 = 3

return (0 + 3) = 3

Therefore, in main ( )

b = 1 + 2 + 3 = 6

Similarly we will calculator other values as 15, 28, 45, 66.

S.72 (a)

Inside the loop, the function will be called 6 times for (0to5).

f(0)→0 ⇒ p=p*a=1*0=0

Now, k is static variable, so after while loop execution its value remain 4. So, while loop will not execute for any other value of variable a. So **sequence is 011111.**

S.73 (d)

++ *p++ will be palse in given order

→ *p that is value at the location currently pointed by p will be taken.

→ ++*p the retrived value will be incremented.

→ when ; is encountered the location will be incremented that is p++ will be executed.

Hence, in the while loop initial value pointed by p is 'n', which is changed to 'i' by executing ++*p

and pointer moves to point, 'a' which is similarly changed to b and so on. Similarly blank space is connected to ' ! '. Thus, **we obtain value in p becomes 'ibj ! gsjfoet' and since p reaches '\0' and p1 points to p thus p1 does not print anything.**

S.74 (b)

funct1 (intx) is called on the values of

c =4×1×1=4 ⇒ retrun a+x ⇒ 100+4 = **104**

c =4×2×2=16 ⇒ retrun a+x ⇒ 100+16 = **116**

c =4×3×3=36 ⇒ retrun a+x ⇒ 100+36 = **136**

c =4×4×4=64 ⇒ retrun b–x ⇒ 200–64 = **136**

c =4×5×5=100 ⇒ retrun b–x ⇒ 200–100 = **100**

S.75 (a)

Output of the above program;

x=40

y=40

This program consists of two expressions, first is an ordinary arithmetic expresion and the other is a pointer expression.

x=5* (temp+5)

where temp = 3

x=5* (3+5)=40

Using the pointer arithmetic

x pointer = & temp;

where temp = 3,

y=5*(3+5)

= 5*8 = 40.

S.76 (a)

First printf prints u =1 v =3

again funct1 (u,v) in called

⇒ Funct1 (1,3)

⇒ u = u+v = 1 + 3 = 4

v = u–v = 4 – 3 = 1

u = u–v = 4–1 = 3

Again printf will print u = 1 v = 3 because values of u and v actually have not changed.

When funct2 (&u, &v); is called

and it is called reference hence the values of u and v are actually interchanged.

⇒ when thind printf (c) is executed, the values of u and v are actually interchanged.

S.77 (a)

if (x/20==5 && (x*y>10011y+z*2<75)) is true.

Hence the inner if is checked

⇒ if (x<<2<=256) is checked

which is fasle

⇒ else part strcpy (x, "pass"); will execute.

S.78 (c)

There output of the program is

2 3

4 6

In this program, the elements of matrix are supplied through arithmetic statements and printed by using nested 'for' statement.

S.79 (d)

fun(5)

function will return x = 5*5

⇒ **x = 25**

S.80 (b)

Output of above program is

10

60

because extern variable remains throughout the program.

S.81 (c)

Output

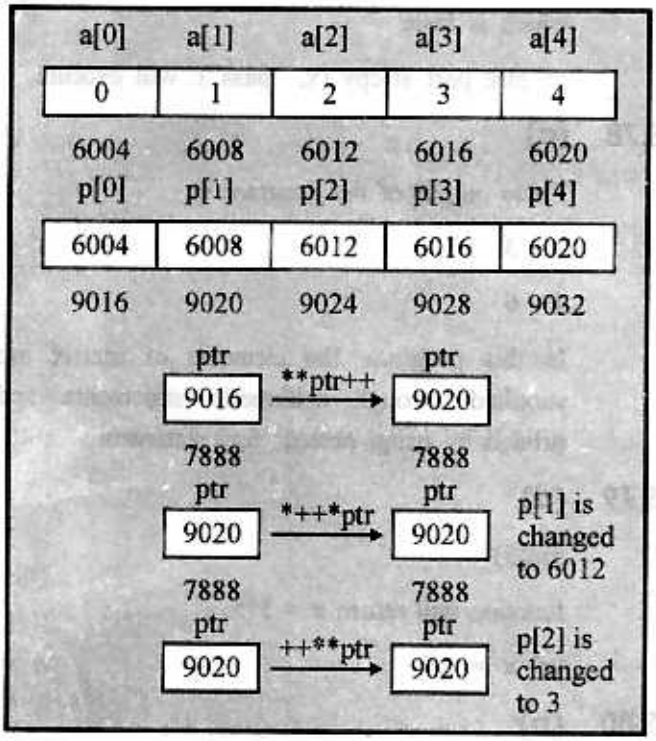
1 1 1

1 2 2

1 2 3

Explanation:

To begin with, the array **a[ ]** is initialized and the array **p[ ]** is set up such that it contains the address of elements of array **a[ ]**. Thus array **p[ ]** becomes an array of pointers. The base address of this array of pointers is then assigned to **ptr**, which is rightly called a pointer to a ponter. The possible arrangement of the array elements in memory is shown in the following figure.



Let us now analyze the expression ****ptr++**. Since the unary operators are associated from right to left, the above expression would evaluate in the order; ***(*(ptr++))**. **ptr** contains the address 9016, therefore incrementing **ptr** would store the address 9020 in **ptr**. ***ptr** would give the value at this address. This value turns out to be 6008, and ***(6008)** would give the value at 6008, i.e. 1. However, this value is not assigned to any variable while evaluating ****ptr++**. Once you are sure that **ptr** contains 9020, let us now proceed to find out what the output of **printf()** is. Let us take one expression at a time and analyze it step by careful step.

ptr – p

Since **ptr** is containing the address 9020, we can as well say that **ptr** is containing the address given by **p + 1**. Thus **ptr - p** is reduced to **(p + 1 – p)**, which yields 1.

*ptr – a

***ptr** means value at the address contained in **ptr**. Since **ptr** contains 9020, the value at this address would be 6006. Now 6008 can be imagined as **(a + 1)**. Thus the expression becomes **(a + 1 – a)**, which is nothing but 1.

**ptr

ptr contains 9020, so ***ptr** would yield 6008, and hence ****ptr** becomes ***(6006)**, which would yield 1.

Thus the output of the first **printf()** turns out to be 1 1 1.

The next statement needs a closer look. In ***++*ptr**, the order of evaluation would be ***(++ (*ptr))**. Since **ptr** contains 9020, ***(ptr)** would yield the value at 9020, i.e. 6008. Then ++ goes to work on **6008** and increments it such that it is now 6012. Thus **p[1]** would now contain 6012. And finally ***(6012)** would give 2, which is ignored since it is not assigned to any variable.

Now, with **ptr** containing 9020, let us once again analyze the expression **ptr – p, ptr – a** and ****ptr**.

ptr – p

Since **ptr** contains 9020, it can be visualized as **(p + 1)**. thus **ptr – p** would become **(p + 1 – p)**, which would be 1.

*ptr – a

***ptr** would give value at address 9020, i.e. 6012,

which is nothing but the address given by **a + 2**. Thus the expression becomes **(a + 2 – a)**, which gives 2.

**ptr

***ptr** gives the value at address 9020, i.e. 6012, and ***(6012)** gives the value at 6012, i.e. 2.

Thus the output of the second **printf()** would be 1 2 2.

Finally, we reach the third expression **++**ptr**. As the unary operators are evaluated from right to left, the order of evaluation of the above expression becomes: **(++(*(*ptr)))**. Since **ptr** contains 9020, ***ptr** yields 6012. ***(6012)** results into 2. This value at the address 6012 is then incremented from 2 to 3.

ptr – p

Since **ptr** contains 9020, it can be visualized as **(p + 1)**, thus **ptr – p** would become **(p + 1 – p)**, which would be 1.

*ptr – a

***ptr** would give value at address 9020, i.e. 6012, which is nothing but the address given by **a + 2**. Thus the expression becomes **(a + 2 – a)**, which gives 2.

**ptr

ptr** gives the value **a address 9020, i.e. 6012, and ***(6012)** gives the value at 6012, i.e. 3.

Thus the output of the third **printf()** is 1 2 3.

S.82 (b)

The union variable u is passed to funct by value. Hence the reassignment within funct is not recognized within main. Therefore **we get only 200 and 0.50000 as legal value.**

S.83 (b)

When we pass an array as a parameter it is possible to obtain different results for call by reference and call by name.

S.85 (a)

main () { (i+j)}

i and j are 100 and 5 (global values)

∴ printf (x+10)

printf (i+j+10) → 100+4→115

Again print(x) ⇒ local i and j are used

⇒ **print (200+20) →220**

S.86 (b)

print(x+10) uses value of i and j locally of they are there.

Now i = 10 local and j = 5 global

⇒ **print (x+10) ⇒ print (10+5+10)→25**

and **print (x) remains the same → 220.**

S.87 (b)

Here (II) A [2][3] will give the compile time error as A is the array of pointer and hence it can be used only A [0], A[1]..... A [9] only.

S.88 (a)

In main () global value of x in 5

we call p (ex) ⇒ p (int *y) will be called.

In this function int x = *y+2 Þ x = 5+2=7 (local value of x to function p)

Q (x); will all int function Q (int 2)

local value of x (=7) will be capied to z

again z≠=x ⇒ z will been 7+5 = 12

Hence first time 12 will be printed.

Next control goes at *y=x–1

Hence *y is conneted wiull the global x(=5) to function main (). And x–1 in reffered to as the local value of x (=7)

Hence *y=x–1 = 7–1 = 6

∴ print (x) ⇒ local value of x=ϕ will be printed second time

Now the function p finished.

Hence control return to main when print (x) is to be executed ⇒ in this print (x) the global value of x refered to *y (=6) will be printed.

S.89 (b)

For larger values of y

$$P = p*\frac{x}{i} = \frac{x}{i}$$

$$s = s+p = 1+\frac{x}{i}$$

when 1< y and iff

$$S = 1+\frac{x}{i}+\frac{x}{i+1}+\ldots = 1+x+\frac{x^2}{2}+\frac{x^3}{\underline{|3}}+\frac{x^4}{\underline{|4}}+\ldots$$

Hence, for larger value of y, $S = e^x$.

S.90 (a)

The main program for given code is

```
#include <stdio.h>
        void main ( )
        { int x, y;
              x = 1 ;
              y = 2 ;
               swap (x, y)
               printf("%d/n", a);
               printf("%d/n", y);
             }
```

If we call swap (x, y) then there is no interchange in the value of x and y because the parameter are passed by value. There is interchange in formal parameter a and b but not interchange in actual parameter x an y because the scope of a and b lies with in the fucntion but not in the main program so the value of x and y remain unchanged. If we will call it by reference then the value of x and y will be interchange look like below code.

```
Void swap (int * a, int * b)
          { int temp ;
    temp   = *a;
    *a     = *b;
    *b     = temp;
     }
```

S.91 (a)

Consider the C code

```
1.  #include<stdio.h>
2.  #include<string.h>
3.  main( )
4.  {   charp[20];
5.      char * s = "string";
6.      int length = strlen(s);
7.      for (int i = 0; i < lenght; i++)
8.          p[i] = s[length – i];
9.      printf("%s", p);
10. }
```

Upto 1 to line 7 there is no problem in the code but in line 8, s is a character pointer variable and p is a character variable. In C we can't assign the value of a pointer variable into normal character variable so **the output of the program produces garbage value or no output is printed.**

S.92 (a)

To swap the element in row with column i.e. M[i][j] with M[j][i] using third variable t (is defined as int type)

So, block at place of X will be replaced by

```
{   t = M[i] [j]
    M[i] [j] = M[j] [i]
    M[j] [i] = t
}
```

S.93 (d)

This is same as the producer consumer process.

```
(C = getchar ()) ! = '/n'
    putchar (C);
```

Here, 'getchar()' fetches the characters until it reaches to the end of string, now 'putchar (C)' puts the characters in inverse recursively.

S.94 (b)

When we don't declare a function (speicfy the prototype), the default rules that the function (foo is our case) would have been assumed to be int, even through it really return a double. This will generates compiler warning and our program will have undefined behaviour is disastrous.

S.95 (a)

Both $work_1$ and $work_2$ performs the same task so S_1 is true. But code in $work_2$ will improve because consider the code.

in work1

int x = a[i + 2];

here a[i + 2] is computed twice.

return a[i + 2] – 1

but in $work_2$

$t_1 = i + 2$

$t_2 = a[t_1]$;

return t_2 – 3 noneed for 2nd computation. So S_2 **is also correct.**

S.96 (d)

S_2 and S_3 are correct statements.

S.97 (a)

f(5)

r = 5

f(3) + 2 = **18**

↑

f(2) + 5 = 16

↑

f(1) + 5 = 11

↑

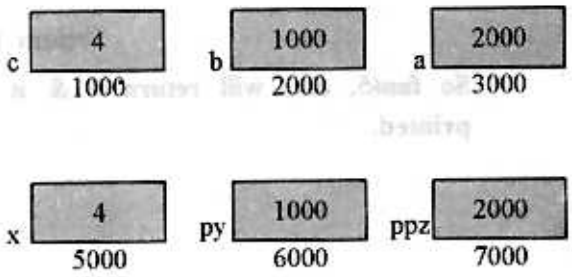
f(0) + 5 = 6

↑

1

S.98 (a)

f(c, b, a) is called by the main ( ) function the graphical execution of the program is given below.



int y, z;

ppz = **ppz + 1=4+1=5; z=ppz = 5

*py = *py + 2; y = *py = 6

x = 4 + 3 = 7

return x + y + z = 7 + 6 + 5 = 18

S.99 (a)

Here we are using the '=' operator which has priority than '!=' operator.

So (C = getchar ( )) has to be in brackets and after reversing the string we use function putchar (c) for printing the character.

S.100 (c)

x = 15

fun(5, &x) let &x = 100

↓

fun(5, 100)

↓ t=5 f=5+3 *f_p=5

return 8

fun(4, 100)

↓ t=3 f=3+2 *f_p=3

return 5

fun(3, 100)

↓ t=2 f=2+1 *f_p=2

return 3

fun(2, 100)

↓ t=1 f=1+-1 *f_p=1

return 2

fun(1, 100)

*f_p=1

return 1

So fun(5, &x) will return 8 & it will be printed.

4.4 ARRAYS

LEVEL–1

Q.1 Arrays can be initialized by default to 0 provided they are

(a) automatic

(b) static

(c) external

(d) (b) and (c)

Q.2 If an array is used as function argument the array is passed

(a) by value

(b) by reference

(c) none of the above, as array cannot be used a function argument.

(d) call by name

Q.3 A constructor that takes as argument an instance of the class in which the constructor is being declared. Such constructor is called as _____

(a) default constructor

(b) deferred constructor

(c) copy constructor

(d) destructor

Q.4 Which one of the following statement is invalid about (i) and (ii):

(i) Ratio y=x;

(ii) Ratio y;y=x;

(a) There is no difference between (i) and (ii), both are valid statements

(b) (ii) declaraction calls the default constructor and then the assignment operator

(c) (i) declaration calls the copy constructor

(d) (b) and (c) are correct statements

Q.5 Arrays can be initialized provided they are

(a) automatic

(b) static

(c) external

(d) both (b) and (c)

Q.6 If initialization is a part of delcaration of a structure then the storage class can be

(a) automatic

(b) static

(c) register

(d) anything

Q.7 Match the following:

Group I		Group II	
1.	two different classes derive a single class	a.	hybrid inheritance
2.	one class derives three different classes	b.	multilevel inheritance
3.	one class derives one class and new class derives one more class	c.	hierarchical inheritance
4.	one class derives two different class and two derived classes derives one new class	d.	multiple inheritance

(a) 1-d,2-b,3-a,4-c

(b) 1-d,2-c,3-b,4-a

(c) 1-b,2-a,3-c,4-d

(d) 1-b,2-c,3-d,4-a

Q.8 Which of the following array initialization is perfectly valid.

(i) int arr [2] [ ] = {12, 34, 23, 45, 56, 45};

(ii) int arr [ ] [ ] = {12, 34, 23, 45, 56, 45};

(iii) int arr [ ] [3] = {12, 34 23, 45, 56, 45};

(a) all of the above are valid

(b) both (i) and (ii)

(c) only (iii)

(d) only (i) and (iii)

Q.9
```
main ( )
{
 int arr [ ] = {10, 20, 30, 45, 67, 56, 74};
 int *i, *j;
 i = & arr [1];
 j = & arr [5];
 printf("%d%d", j–i, *j–*i);
}
```
What would be the output of above program.

(a) 4, 36

(b) 5, 36

(c) 4, 57

(d) Error. Pointer cannot be subtacted from one another.

Q.10 argv is a/an

(a) array of strings

(b) array of character pointers

(c) pointer to an array of character pointer

(d) None of these

Q.11 The rule for implicit type conversion in 'C' is

(a) unsigned < int < double < float

(b) int < unsigned < double < float

(c) unsigned < int < float < double

(d) int < unsinged < float < double

Q.12 In C programming language x – = y + 1; means

(a) x = x – y – 1

(b) x = – x + y + 1

(c) x = – x – y – 1

(d) x = x – y + 1

Q.13 If integer needs two bytes of storage, then maximum value of a signed integer is

(a) 2^{15}

(b) $2^{15} - 1$

(c) 2^{16}

(d) $2^{16} - 1$

Q.14 Choose the correct statements

(a) All the elements of the array should be of the same data type and storage class

(b) The number of subscripts determines the dimension of the array

(c) The array elements need not be of the same storage class

(d) In an array definition, the subscript can be any expression yielding a non zero integer value.

Q.15 If a is a unsigned integer variable whose value is hx: 6db7, what is the value of ~a?

(a) hx: 9248

(b) hxhh1

(c) h11hhhh1

(d) hx248

Q.16 Which of the following statements are correct about an array?

(a) The array int num[26]; can store twenty-six elements.

(b) The expression num[1] designates the very first element in the array.

(c) It is necessary to initialize the array at the time of declaration.

(d) The expression num[27] desiginates the twenty-eighth element in the array.

(e) The declaration num[SIZE] is allowed if SIZE is a macro.

Q.17 Consider the declaration

static char hello [ ] = "hello";

The output of print f("%s\n", hello);

will be the same as that of

(a) puts ("hello");

(b) puts (hello);

(c) print f("%s\n", "hello");

(d) puts ("hello\n");

Q.18 Consider the following type definition.

typedef char x[10];

x myArray[5];

What will "sizeof(myArray)" be? (Assume one character occupies 1 byte)

(a) 15 bytes

(b) 10 bytes

(c) 50 bytes

(d) 30 bytes

Q.19 C does no automatic array bound checking. This is

(a) true

(b) false

(c) C's asset

(d) C's shortcoming

Q.20 If 'arr' is a two dimensional array of 10 rows and 12 columns, then arr[5] logically points to the

(a) sixth row

(b) fifth row

(c) fifth column

(d) sixth column

Q.21 Pick the correct answer.

If x is an one dimensional array, then

(a) &x[i] is same as x + i – 1

(b) * (x + i) is same as * (&x[i])

(c) *(x + i) is same as x[i]

(d) *(x + i) is same as *x + i

Q.22 Consider the declaration

char x[] = "WHATIZIT";

char * y = "WHATIZIT";

Pick the correct answers.

(a) The output of puts(x) and puts(y) will be the same

(b) The output of puts (x) and puts(y) will be different

(c) The output of puts (y) is implementation dependent.

(d) None of the above comments are true.

LEVEL–2

Q.23 Trace the output

```
main ( )
{   unsigned int m [ ] = { 0×01, 0×02, 0×04,
                           0×08, 0×10, 0×20,
                           0×40, 0×80);
unsigned char n, i;
scanf("%d, &n);
for (i = 0; i <=7; i++)
        { if (n & m [i])
          printf("\n yes");
         }
}
```

(a) This program would give an error

(b) Testing whether the individual bits of n are on or off

(c) None of the above

(d) Putting off all bits which are on in the number n

Q.24 Following declaration can be used for

```
:
for (i = 0; i <= n–2; i++)
for (j = i + 1; j <= n–1; j++)
if (a[i] > a[j])
        {
          temp = a[i];
           a[i] = a[j];
           a[j] = temp;
        }
:
```

(a) sorting an array in ascending order

(b) sorting an array indescending order

(c) finding the unit matrix from given matrix

(d) (a), (b) & (c)

Q.25

```
main ( )
{
   int S[4] [2] = {
                      {1234, 56},
                      {1212, 33},
                      {1434, 80},
                      {1312, 78}
                   };
   int i, j ;
   for (i = 0; i <= 3; i ++)
   printf("%d", S[i]);
}
```

Let the base address is 1245024. What will be the output of above program.

(a) 1234, 1212, 1434, 1312

(b) 1245024, 1245028, 1245032, 1245036

(c) Gives Error

(d) 1234, 56, 1212, 33

Q.26 Consider the following program segment:

```
void main ( )
{
        int d [ ] = {20, 30, 40, 50, 60};
        int *u, k;
        u = d;
        k = *((++u)++);
        k+ = k;
        (++u)+ = k;
        printf("%d", *(++u));
}
```

(a) 24842

(b) 24942

(c) Error

(d) error

Q.27 What error are you likely to get when you run the following program?

```
main ( )
{
   struct emp
   {
   char name [20];
   float sal;
};
struct emp e[10];
int i;
   for (i = 0; i <= 9; i++)
   scanf("%s%f", e[i].name  &e[i].sal);
```

(a) Strings cannot be nested inside structures

(b) Suspicious pointer conversion

(c) Floating point formats not linked

(d) Cannot use scan f( ) for structures

Q.28 Trace the output:

```
# include <stdio.h>
main ( )
{
    int i = 123; float x = 13.0, y = –3.3;
    printf("%7.0f%#7.0f%7g%#7g", x, x, y, y);
}
```

(a) 123 13 –3.3e+00 –3.30

(b) 13 13 –3.3 –3.30000

(c) None of these

(d) +123 +13 –3.3e+00 –3.3

Q.29 What is the output of the following 'C' program?

```
main ( )
{
    char str1[ ] = "Hello";
    char str2[ ] = "Hello";
    if (str1 = = str2)
        printf("\nequal");
    else
        printf("\nUnequal");
}
```

(a) Equal

(b) Error

(c) Unequal

(d) None of these

Q.30 Which of the following statements is true after execution of the program

```
int a[10], i, *p;
a[0] = 1;
a[1] = 2;
p = a;
(*p) ++;
```

(a) a[1] = 3

(b) a[1] = 2

(c) a[0] = 2

(d) all of these

Q.31 Consider the following program fragment

```
main ( )
{
int a, b, c;
b = 2;
a = 2*(b++);
c = 2*(++b);
}
```

Which of the following is correct?

(a) a = 4, c = 8

(b) b = 3, c = 6

(c) a = 3, c = 8

(d) a = 4, c = 6

Q.32 Which of the following constants does not produce over flow when assigned to an unsigned long integer variable?

(i) OXFFFFFFFF

(ii) 42949672961

(iii) 07777777777

(a) (i) and (iii) above

(b) none of these

(c) all of the above

(d) (i) and (ii) above

Q.33 The expression 5 – 2 – 3 * 5 – 2 will evaluate to 18, if – is right associative and *

(a) – has precedence over *

(b) – has precedence over *

(c) * has precedence over –

(d) * has precedence over –

Q.34 The following program

```
main ( )
{
static int a [ ] = {7, 8, 9} ;
printf("%d", 2 [a] + a [2]) ;
}
```

(a) results in segmentation violation error

(b) results in bus error

(c) will not compile successfully

(d) none of these

Q.35 The following program

```
main ( )
{
static char a [3] [4] = {"abcd", "mnop", "fghi"};
putchar (**a) ;
}
```

(a) prints garbage

(b) results in bus error

(c) will not compile successfully

(d) none of these

Q.36 The following program

```
main ( )
{
inc ( ) ; inc ( ) ; inc ( ),
}
inc ( )
{
static int x; printf("%d", ++x) ;
}
```

(a) prints 123

(b) prints 3 consecutive, but unpredictable numbers

(c) prints 111

(d) print 012

Q.37 Consider the declaration

```
struct wer {unsigned a : 5;
    unsigned : 0 ;
    unsigned b : 3;
    unsigned : 0 ;
    unsigned c : 2;
    unsigned : 0; } v;
```

The storage needed for v is

(a) 4 words

(b) 2 words

(c) 3 words

(d) 1 word

Q.38 If an int is 2 bytes wide then which of the following is the correct output for the program given below?

```
# include <stdio.h>
void fun (char**);
int main()
{
    char * argv[] = {"ab", "cd", "ef", "gh"};
    fun (argv);
    return 0,
}
void fun (char**p)
{
    char*t;
    t = (p += sizeof (int)) [-1];
    printf ("%s\n", t);
}
```

(a) ab

(b) cd

(c) ef

(d) gh

Q.39 How will you define the function f() in the following program?

```
int arr[MAXROW][MAXCOL];
fun (arr);
```

(a)
```
void fun (int a[][MAXCOL])
{
}
```

(b)
```
void fun (int a[MAXROW][])
{
}
```

(c)
```
void fun (int (*ptr)[MAXROW])
{
}
```

(d)
```
void fun (int(*ptr)[MAXCOL])/* ptr is pointer to an array*/
{
}
```

Q.40 What will be the output of the following program (myprog) if it is executed from the command-line as shown below?

myprog one two three

```
/* myprog.c*/
#include <stdio.h>
int main (int argc, char *argv[])
{
int i ;
for (i = 1; i <= 3; i++)
    printf ("%c", *argv[i]);
printf ("\n");
return 0;
}
```

(a) myp
(b) one
(c) ott
(d) no output

Q.41

```
#include <stdio.h>
int main()
{
    static int a[5] = {5, 10, 15, 20, 25};
    int i, j, m, n;
    i = ++a[1];
    j = a[1]++;
    printf ("i=%d j=%d a[1]=%d\n", i, j, a[1]);
    i = 1;
    m = a[i++];
    printf ("i=%d m=%d\n", i, m);
    i = 2;
    n = a[++];
    printf ("i=%d n=%d\n", i, n);
    return 0;
}
```

(a) i = 11 j = 11 a[1] = 12
    i = 1 m = 13
    i = 2 n = 20
(b) i = 11 j = 11 a[1] = 11
    i = 2 m = 12
    i = 3 n = 20
(c) i = 11 j = 11 a[1] = 12
    i = 2 m = 12
    i = 3 n = 20
(d) None of the above

Q.42

```
#include <stdio.h>
int main ()
{
    static int a[3][3] = {
                          1,2,3,
                          4,5,6,
                          7,8,9
                         };
    int *ptr[3] = {a[0], a[1], a[2]};
    int **ptr1 = ptr;
    int i;
    for (i = 0; i <= 2; i++)
        printf ("%d", *ptr[i]);
    printf ("\n");
    for (i = 0; i <= 2; i++)
        printf ("%d", *a[i]);
    printf ("\n");
    for (i = 0; i <= 2; i++)
    {
        printf ("%d\n", **ptr1);
        ptr1++;
    }
    return 0;
}
```

(a) 1 5 9
    1 5 9
    1
    5
    9
(b) 1 4 7
    1 4 7
    1
    4
    7
(c) Error : L value required in function main.
(d) 1
    4
    7
    1 4 7
    1 4 7

Q.43 Consider the following declaration:

int a, *b = &a, **c = &b;

The following program fragment

a = 4;

**c = 5;

(a) assigns 5 to a

(b) assigns the value of b to a

(c) assigns address of c to a

(d) does not change the value of a

Q.44 The following program fragment

int x [5][5], i, j ;

for (i = 0; i < 5; ++ i)

for(j = 0; j < 5 ; j ++)

x[i][j] = x[j][i];

(a) transposes the given matrix x

(b) makes the given matrix x, symmetric

(c) doesn't alter the matrix x

(d) none of the above

Q.45 The following program fragment

int m, n, b = m = n = 8;

char wer[80];

printf(wer, "%d%d%d", m, n, b);

puts (wer);

(a) prints the string 8 8 8

(b) prints the null string

(c) prints the string 888

(d) none of the above

Q.46 int hh = 16;

static char wer [] = "NO SUBSTITUTE FOR HARD WORK";

printf (" %-10. *s", hh, wer);

outputs

(a) NO SU

(b) NO SUBSTITUTE FO

(c) NO SU

(d) error message

Q.47 Choose the statement that best defines an array

(a) It is a collection of items that share a common name.

(b) It is a collection of items that share a common name and occupy consecutive memory locations.

(c) It is a collection of items of the same type and storage class that share a common name and occupy consecutive memory locations.

(d) None of the above

Q.48 A possible output of the following program fragment

static char wer [ ][5] = { "harmot", "merli", "axari"};

printf ({"%d %d %d", wer, wer [0], & wer [0] [0]); is

(a) 262164 262164 262164

(b) 262164 262165 262166

(c) 262164 262165 262165

(d) 262164 262164 262165

Common Data For Questions 49 & 52:

static char wer [3][4] = {"bag", "let", "bud"};

char (*ptr)[4] = wer;

Q.49 The possible output of printf ("% d %d", ptr, ptr + 1); is

(a) 262 262

(b) 262 266

(c) 262 263

(d) 262 265

Q.50 The possible output of printf ("%d %d", wer[1], wer[1] + 1); is

(a) 162 163

(b) 162 162

(c) 162 166

(d) 162 165

Q.51 putchar (* (wer[1] + 1));

(a) prints e

(b) prints a

(c) prints l

(d) prints b

Q.52 In which of the following cases will the character 't' be printed?

(a) putchar (*(* (ptr + 1) + 2));

(b) putchar (* (wer[1] + 2));

(c) putchar (* (ptr +1) + 2) ;

(d) None of the above

Q.53 Consider the following program

```
main ()
{
    char x[10], *ptr = x;
    scan f("%s", x);
    change (&x[4]);
}
change (char a[])
{puts (a);}
```

If 'abcdefg' is the input, the output will be

(a) abcd

(b) abc

(c) efg

(d) garbage

Q.54
```
void main ( )
{ int i ;
  char a[] = "\0" ;
  if (print f("%s\n", a))
    print f ("OK here \n") ;
else
    print f ("Forget it \n") ;
}
```

(a) Forget it

(b) OK here

(c) OK here
Forget it

(d) None of these

LEVEL–3

Q.55 Trace the output

```
# include <stdio h>
# define R    3
# define CO   4
int Z[R] [CO] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12};
main ( )
{   int a, b, c = 999;
    for (a = 0; a < R; ++a)
    for (b = 0; b < CO; ++b)
    if (z[a] [b] < c)
    c = z[a] [b];
  printf("%d", c);
}
```

(a) 1

(b) 2

(c) 12

(d) 11

Q.56 Array X is 2–D array that stores elements of n × n square matrix following a program segment is used to check whether matrix stored by A is symmetric or not. The logical expression to be written in first & second if loop is _____

```
flag = 1;
for (i = 0; i < n – 1; ++i)
for (j = 0, j < n – 1; ++j)
if ( ——— ) flag = 0;
if ( ——— ) printf("Matrix is symmetric");
else printf ("Matrix is not symmetric");
```

(a) X [i] [j]! = X [j] [i]
flag = = y

(b) X [i] [j]! = X [j] [i]
! flag

(c) X [i] [j]! = X [j] [i]
flag

(d) X [i] [j]! = X [j] [i]
flag = y

Q.57 Consider an array a[n] in which all the elements are stored in ascending order. After doing the same one integer x is read. Then by considering following code which statement is correct.

```
    :  :  :
    :  :  :
i = n – 1; b = 1;
while ((i >= 0) & & (b))
{
  if (a[i] > x)
        {
          a [i + 1] = a[i]
                i – –;
        }
        else b = 0;
}
a [i + 1] = x;
for (i = 0, i <= n, i++)
printf("%d", a[i]);
}
```

(a) The above code place x in proper ordered array properly

(b) The above code compares x with all elements and place it at the beginning of an array.

(c) The above code compares x with all elements and place at the end of an array.

(d) The above code compares x with all elements and place at the end of an array.

Q.58 What does the following program segment print?

```
main()
    {
    int i;
    int iarray [4]= {1, 2, 3, 4};
    #define SIZE (size of (iarray)/sizeof (int))
    for (i=0;i<SIZE;++i)
    iarray [i]+ =2;
    printf (value is %d\n",iarray[3]);
}
```

(a) 3

(b) 6

(c) Compilation error because of #define within main module

(d) None of the above

Q.59 If the following program fragment (assume negative numbers are stored in 2's complement form)

```
unsigned i = 1;
int j = – 4 ;
printf("%d", 8*size of (int) ;
```

output is (log in the answers are to the base two)

(a) log 8 (x + 3)

(b) log (x + 3)

(c) an unpredictable value

(d) 8 * log (x + 3)

Q.60 In a certain machine, the sum of an integer and its 1's complement is $2^{20} - 1$. Then size of (int), in bits will be

(a) unpredictable

(b) 32

(c) 16

(d) none of these

GATE QUESTIONS

Q.61 Let A be a two dimensional array declared as follows:

A: array [1...10][1...15] of integer;

Assuming that each integer takes one memory locations the array is stored into row major order and the first element of the array is stored at location 100, what is the address of the element A [i] [j] ? **[GATE 1998]**

(a) 15i + j+84

(b) 15j+i+84

(c) 10j+i+89

(d) 10i+j+89

Q.62 An n x n array v is defined as follows:

v[i,j] =i–j for all i, j, $1 \le i \le n$, $1 \le j \le n$

The sum of the elements of the array v is **[GATE 2000]**

[1-Mark]

(a) 0

(b) $n^2(n+1)/2$

(c) n^2-3n+2

(d) n–1

Q.63 Consider the following C declaration

```
struct {
    short  s [5]
    union {
    float y;
    long z; } u;
    } t;
```

Assume that objects of the type short, float and long occupy 2 bytes, 4 bytes and 8 bytes, respectively. The memory requirement for variable t, ignoring alignment considerations, is

[GATE 2000]

[1-Mark]

(a) 22 bytes

(b) 14 bytes

(c) 18 bytes

(d) 10 bytes

Q.64 A program P reads in 500 integers in the range [0, 100] representing the scores of 500 students. It then prints the frequency of each score above 50. What would be the best way for P to store the frequencies? **[GATE 2005]**

[1-Mark]

(a) A dynamically allocated array of 550 numbers

(b) An array of 100 numbers

(c) An array of 500 numbers

(d) An array of 50 members

Q.65 The following C function takes two ASCII strings and determines whether one is an anagram of the other. An anagram of a strings s is a string obtained by permuting the letters in s.

```
int anagram (char*a, char*b){
int count [128], j;
for (j = 0; j < 128; j++) count[j] = 0;
j = 0;
while (a[j] && b[j]){
    A;
    B;
}
for (j = 0; j < 128; j++) if (count[j]) return 0;
return 1;
}
```

Choose the correct alternative for statements A and B. **[IT-GATE 2005]**

[2-Marks]

(a) A: count [a[j]]++ and B: count[b[j]]- -

(b) A: count [a[j]]++ and B: count[b[j]]++

(c) A: count[a[j++]]++ and B: count[b[j]]- -

(d) A: count[a[j]]++ and B: count[b[j++]]- -

Q.66 The following C function takes a singly-linked list of integers as a parameter and rearranges the elements of the list. The list is represented as pointer to a structure. The function is called with the list containing the integers 1, 2, 3, 4, 5, 6, 7 in the given order. What will be the contents of the list after the function completes execution?

```
struct node (int value; struct node * next;};
void rearrange (struct node * list) {
    struct node *p, *q;
    int temp;
    if (!list || !list → next) return;
    p = list; q = list → next;
    while (q) {
    temp = p → value;
    p → value = q → value;
    q → value = temp;
    p = q → next;
    q = p ? p → next:0;
        }
    }
```

[IT-GATE 2005]

[2-Marks]

(a) 1, 2, 3, 4, 5, 6, 7

(b) 2, 1, 4, 3, 6, 5, 7

(c) 1, 3, 2, 5, 4, 7, 6

(d) 2, 3, 4, 5, 6, 7, 1

Q.67 Let a be an array containing n integers in increasing order. The following algorithm determines whether there are two distinct numbers in the array whose difference is a specified number S > 0

```
i = 0; j = 1;
while (j < n) {
    if (E) j++;
    else if (a[j] – a[i] == S) break;
    else i++;
}
if (j < n) printf ("yes") else printf ("no");
```

Choose the correct expression for E.

[IT-GATE 2005]

[2-Marks]

(a) a[j] – a[i] > S

(b) a[j] – a[i] < S

(c) a[i] – a[j] < S

(d) a[i] – a[j] > S

ANSWER KEY

1	d	2	b	3	c	4	a	5	a
6	d	7	b	8	c	9	a	10	b
11	d	12	a	13	b	14	a, b	15	a
16	a, d, e	17	a, b, c	18	c	19	a, d	20	a
21	b, c	22	a	23	b	24	a	25	b
26	b	27	c	28	b	29	c	30	c
31	a	32	d	33	b	34	d	35	d
36	a	37	c	38	b	39	a, d	40	c
41	c	42	b	43	a	44	b	45	c
46	b	47	c	48	a	49	b	50	a
51	a	52	a, b	53	c	54	b	55	a
56	c	57	a	58	c	59	b	60	b
61	a	62	a	63	c	64	d	65	a
66	b	67	b						

SOLUTIONS

S.1 (d)

Array can be initialized provided their storage calls is external or static if we initialized array without specifying static or external, by default it will take auto storage class and initialize array to garbage values. Automatic variable contains default value.

S.2 (b)

If array is used as functions argument the array is passed by reference. That is we are pasing pointer to 1^{st} element i.e. 0^{th} element of array. Array name can be treat like pointer to the 0^{th} element of array i.e.a [0].

S.3 (c)

Definition of copy constructor.

S.4 (a)

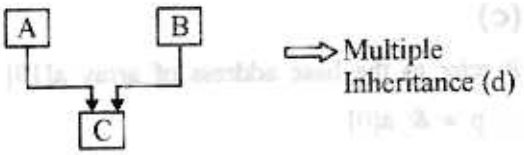
There is no difference between (i) and (ii).

(i) ⇒ it calls the copy constructor

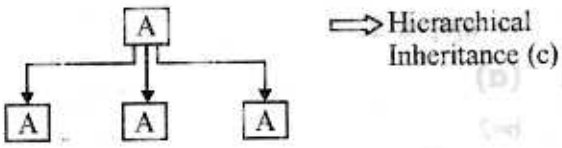
(ii) ⇒ it calls the default constructor and then the assignment operator

S.7 (b)

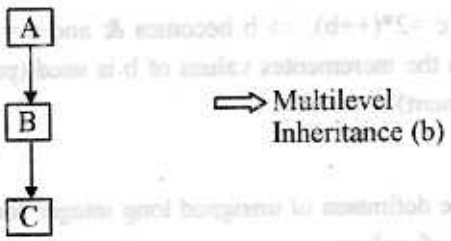
1.



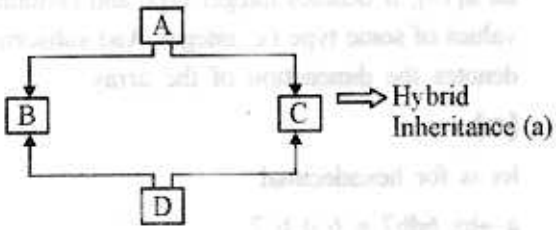
2.



3.



4.



S.8 (c)

It is important to remember that while initializing an array it is necessary to mention the second dimension, whereas the first dimension (row) is optional.

int arr [ ] [3] = {12, 34, 23, 45, 56, 45} is perfectly valid,

while (i) and (ii) are invalid.

S.9 (a)

i and j are pointer to the same array.

j – i gives number of elements between i and j, i.e., **4.**

$*j - *i = 56 - 20 = \mathbf{36}$

S.10 (b)

The declaration of argv is has * argv[];

S.11 (d)

Implicit type conversion rule depicts the increasing order of range of values.

S.12 (a)

x– = y+1 ⇒ x=x–(y+1)= x–y–1

S.13 (b)

The int signed int

range in – 32768 to 32767

⇒ Max range = $(2^{n-1} - 1)$

$= (2^{16-1}-1)$

$= \mathbf{2^{15} - 1}$

max value.

In signed magnitude form, one bit is dedicated to store the sign (e.g., 1 for negative and otherwise). Only the remaining 15 bits are available to store the magnitude. Hence the answer.

S.14 (a, b)

int a[15], if denotes integer type and contains 15 values of some type i.e. integer. And subscript 15 denotes the dimenstion of the array.

S.15 (a)

hx is for hexadecimal

a→hx 6db7 ≡ 6 d b 7

= 0110 1101 1011 0111

∴ ~ a = 1001 0010 0100 1000

= 9 2 4 8

S.17 (a, b, c)

All condition will print "hello"

S.18 (c)

myArray[5] points to every single field of array x[10],

Hence, **Total bytes = Total bytes of myArray[5] × Total bytes of x[10].**

S.19 (a, d)

C does no array bound checking. Because of this, one can access fifth element of an array that is declared to be of lesser size.

S.20 (a)

Arr[5] represents Arr[5][0] which points to sixth row

S.23 (b)

ox mean hexadecinal values

Hexadecimal	Decimal	Equivalent values
01	01	0000 0001
02	02	0000 0010
04	04	0000 0100
08	08	0000 1000
10	16	0001 0000
20	32	0010 0000
40	64	0100 0000
80	128	1000 0000

n&m[i] ⇒

S.24 (a)

The given declaration only be use for sorting an array in ascending order.

S.25 (b)

In the 2–D arrary s[i][j] the term s[i] refer to the address of the conesproding row.

Hence s[i] when printed using loop it will print the s[0], s[1], s[2] and s[4] the base address of all the 4 rows.

⇒ base address of 0^{th} row = **1245024**

⇒ base address of 1^{st} row = 1245024 + 4 = **1245028**

⇒ base address of 2^{nd} row = 1245028 + 4 = **1245032**

⇒ base address of 3^{rd} row = 1245032 + 4 = **125036**

S.28 (b)

with floating point representation causes a decimal point to be present in all floating point number, even if the data item is a whole number

S.29 (c)

When we are comparing the array names, the compiler actually compares the pointer as the array name refer to the base address of array.

S.30 (c)

P refer to the base address of array a[10]

∴ p ≡ & a[0]

∴ p ≡ &a [0]

∴ (*p)++ (a[0])++⇒1+1=2

⇒ **a[0] is 2.**

S.31 (a)

b=2

a=2*(b++) ⇒ a becomes 3 and a =2*2 because orignal value of b is used (postfix)

again c =2*(++b); ⇒ b becomes & and c = 2*4 = 8 as the incrementes values of b is used (prefix increment).

S.32 (d)

See the definition of unsigned long integer for the range of values

S.33 (b)

5 – 2 – 3 * 5 – 2 will yield 18, if it is treated as (5 – (2 – 3))*((5 – 2). i.e., **if–has precedence over * and if it associates from the right.**

S.34 (d)

a [2] will be converted to * (a + 2).

* (a + 2) can as well be written as * (2 + a).

* (2 + a) is nothing but 2 [a]. So, a [2] is same as 2 [a], which is same as * (2 + a). **So, it prints 9 + 9 = 18.**

S.35 (d)

* a points to the string "abcd". **a is the first character of "abcd", which is the character "a".

S.36 (a)

By default x will be initialized to 0. Since its storage class is static, it preserves its exit value (and forbids reinitialization on re-entry). **So, 123 will be printed.**

S.37 (c)

Total bits required for any variable v.

= 5 + 0 + 3 + 0 + 2 = 10 bits

⇒ 3 words

one word needs 4 bits

two words needs another 4 bits

⇒ **bits needs 3 words.**

S.40 (c)

* argv [] is a pointer array, which through the command line can only hold the starting address of every individual string.

Hence,

*argv[0], contains starting address of 'myprog', *argv[1]

contains starting address of 'one' and so on. Hence, **result is 'ott'.**

S.41 (c)

Output

i = 11 j = 11 a[1] = 12

i = 2 m = 12

i = 3 n = 20

Explanation:

In **i = ++t[1]**, since ++ precedes **a[1]**, firstly the value of **a[1]**, i.e. 10 would be incremented to 11, and then asgined to the variable **i**. as against this, in the next statement, firstly the element **a[1]** (i.e. 11) would be assigned to **j** and then the value of **a[1]** would be incremented to 12. The values of **i, j** and **a[1]** are then printed out.

Next, **i** is reset to 1. In **m = a[i++]**, since ++ succeeds **i**, firstly **a[i]**, i.e. 12 assigned to variable **m**, and then the value of **i**, i.e. 1, is incremented to 2. The values of **i** and **m** are then printed out.

After that **i** is once again reset to 2. In **n = a[–n–i]** since ++ precedes **i**, firstly it is incremented to 3 and then the value of **a[3]**, i.e. 20 is assigned to the variable **n**. Finally, the values of **i** and **n** are printed out.

S.42 (b)

Output

1 4 7

1 4 7

1

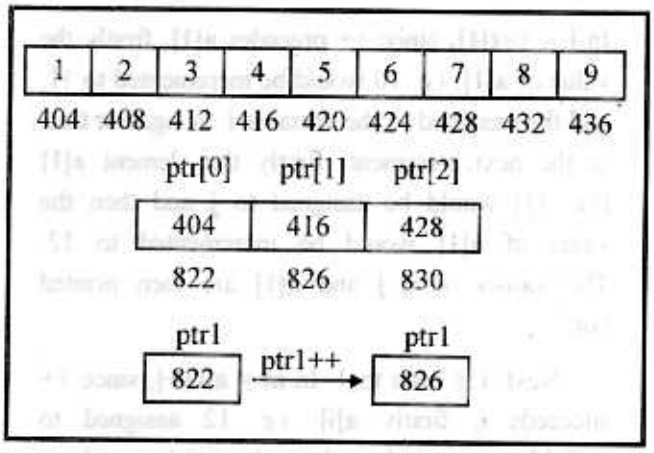
4

7

Explanation:

ptr[ ] has been declared as an array of pointers containing the base address of the three 1-D arrays. Once past the declarations, the control reaches the first **for** loop. In this loop the **printf()** prints the values at, addresses stored in **ptr[0], ptr[1]** and **ptr[2]**, which turn out to be 1, 4 and 7.

In the next **for** loop, the value at base addresses stored in the array **a[ ]** are printed, which once again turn out to be 1,4 and 7. The third **for** loop is also simple. Since **ptr1** has been initialised to the base address of the array **ptr[]**, it contains the address 822.



Therefore ***ptr1** would give the value at address 822, i.e. 404, and ****ptr1** would give the value at address given by ***ptr1**, i.e. value at 404, which is 1. On decrementing **ptr1** it points to the next location after 822, i.e. 826. Therefore next time through the **for** loop, ****ptr1** gives value at 416 (which is obtained through ***ptr1**), i.e. 4. Similarly, last time through the loop, the value 7 gets printed.

S.43 (a)

**c = 5, essentially a = 5, as can be seen with the following pictorial representation of the given declarations.

Address	Value	Name
100	4	a
120	100	b
135	120	c

S.44 (b)

condition x[i][j] = x[j][i] equal the matrix to **symmetric matrix.**

S.45 (c)

printf(wer, "%d%d%d", m, n, b);

print value of m, n, b only because wer array don't have values.

S.46 (b)

"%-10." takes only 16 characters through hh from array wer [], with 0 space and including space.

S.47 (c)

Definition of array.

S.48 (a)

All the three input to printf points to address of array wer, which is equal but a garbage value.

S.49 (b)

The declaration means, 'ptr' is a pointer, pointing to a one dimensional character array of size 4. It is assigned the address wer. So, **ptr and ptr + 1 will differ by 4 bytes.**

S.50 (a)

wer [1] represent to address of wer [1][0],

and wer[1] + 1 represents next address.

S.51 (a)

wer[1][0] + 1 represents second row and second column, i.e., **'e'.**

S.52 (a, b)

(a) *(ptr + 1) represents pointer to value 'g' and ***(* (ptr + 1) + 2)) represents value of t.**

(b) (* (wer[1] + 2)) represents to value of **wer[1][0] + 2.**

S.53 (c)

change (&x[4]) provides fifth elements of char x[10], after which elements are printed.

S.54 (b)

Print f will return how many characters does it print. Hence **printing a null character returns 1 which makes the if statement true, thus "OK here" is printed.**

S.55 (a)

if condition is true for every combination of R and co.

Hence c=z [a][b] is executed evey time.

will be executed when the loop is exited.

The last value is stored in c. Hence **1 will be printed because if condition is false, no value will assign in C.**

S.56 (c)

For checking symmetric condition of the matrix we have to check **X [i] [j]! = X [j] [i] and in second loop we have flag true then we are printing that matrix is symmetric.**

S.57 (a)

It is similar to insertion sort. So, x will be placed at proper order.

S.59 (b)

Let size of (int) = 1, So, – 4 will be stored as 11111100. Since we are adding unsigned and signed integers, the signed gets coverted to unsigned. So, i + j will become 11111101. We are trying to print this as an unsigned integer.

So, what is printed will be $2^8 - 1 - 2$. So log (x + 3) = 8 (i.e., *size of (int)).

S.61 (a)

staring address = 100

$\Rightarrow$ Ist element of 1st now =100

Ist element of 2nd row =100+15

Ist element of 3rd row =100+15+2

: :

Ist element of ith row =100+15(i–1)+1

IIst element of ith row = 100+15 (i–1)+i–1.

:

jth element of ith row = 100+15(i–1)+j–1

100+15i–15+j–1

= 84+15i+j.

S.62 (a)

Sum of elements of array

$$= \sum^{n}(i-j)$$

$$= \sum_{i=1}^{n} i - \sum_{j=1}^{n} j$$

$$= \frac{n(n+1)}{2} - \frac{n(n+1)}{2}$$

$$= 0$$

S.63 (c)

In struct, S is an array of type short, and it containts 5 element.

$\therefore$ memory required = 10 bytes

In union, two variables are declared out of which z is of type long, it requires 8 bytes

$\therefore$ memory required = 10+8 = 18 bytes.

S.64 (d)

There are 500 students the score range is 0 to 100. Print the frequency of those student whose score above 50. So frequency range contains score from 50 to 100, so **an array of 50 numbers is suitable for representing the frequency.**

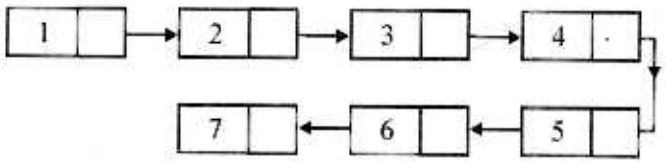
S.65 (a)

In permutation we can place a letter any where but number of letters and number of counts of each letter should be same.

Here a[j] or b[j] gives us the ASCII value of character, so we increase index of that character a[j] by 1 and then decrease the index of corresponding index of b's by one. At final, we must have all values of count zero.

S.66 (b)

Initial linked list is



After function execution pointer values will be as

p	q	temp
1	2	1
2	1	
3	4	
4	3	3
5	6	
6	5	5
7		

So final link will be 2, 1, 4, 3, 6, 5, 7.

S.67 (b)

Notice that numbers are stored in increasing order and S > 0. If we use a[i] – a[j] while i < j then result is always negative. So we have to choose option from first two options.

If **a[j] – a[i]<S**

This mean that we have to increase [j] so next time we compare with higher number.

If a[j] = a[i] we have done it else.

If a[j] – a[i] > S,

then we have to increase index of [i], which is not the condition.

4.5 STRUCTURES AND UNION

LEVEL–1

Q.1 Most appropriate sentence to describe unions is

(a) Unions are used for set operations

(b) Unions contain members of differnt data types which share the same storage area in memory

(c) Unions are less frequently used in program

(d) Unions are like structures

Q.2 Consider the following declarations

```
type def struct
{
   char name [20];
   char mid_name [5];
   char sur_name [20];

} NAME;
NAME class [20];
```

which of the following statement is correct about class?

(a) Class is a new type

(b) Class is an array of 20 characters only

(c) None of these

(d) Class is any array of 20 names where each name consists of a name, mid_name and sur_name

Q.3 Consider the declaration

```
static struct { unsigned a:5;
                unsigned b:5;
                unsigned c:5;
                unsigned d:5;} v=(1,2,3,4);
```

v occupies

(a) 4 words

(b) 2 words

(c) 1 words

(d) none of the above

Q.4 A short integer occupies 2 bytes, an ordinary integer 4 bytes and a long integer occupies 8 bytes of memory. A structure is defined as

```
struct ABC
{  short a;
   int b;
   long c;
} ABC [10];
```

Then the total memory requirement for ABC is

(a) 24

(b) 40

(c) 140

(d) 15

Q.5 Which of the following structure declaration is correct?

(a)
```
struct book
{
    char name[10];
    float price;
    int pages;
};
```

(b)
```
struct
{
    char name[10];
    float price;
    int pages;
}
```

(c)
```
struct book
{
    char name[10]
    float price
    int pages
},
```

(d) All of the above

Q.6 Will the following code work? If 'yes', Select answer.

```
#include <stdio.h>
#include <alloc.h>
struct emp
{
    int len;
    char name[1];
};
int main()
{
    char newname[] = "Rahul";
    struct emp * p = (struct emp*) malloc
(sizeof (struct emp) -1 + strlen (newname) +
1);
p -> len = strlen (newname);
strcpy (p -> name, newname);
printf ("%d%s\n", p-> len, p-> name);
return 0;
}
```

(a) 5 Rahul

(b) 0 Rahul

(c) error

(d) none of the above

Q.7 Point out the error, if any, in the following code.

```
#include <stdio.h>
int main()
{
    struct emp
    {
        char n[20];
        int age ;
    };
    struct emp e1 = {"Dravid", 23};
    struct emp e2 = e1;
    if (e1 == e2)
    printf (" The structures are equal\n");
return 0;
}
```

(a) The structures are equal

(b) Dravid

(c) compiler error

(d) None of the above

Q.8 Which of the following is the correct output for the program given below?

```
#include <stdio.h>
int main()
{
    union var
    {
        int a, b;
    };
    union var v;
    v.a = 10;
    v.b = 20;
    printf ("%d\n", v.a);
    return 0;
}
```

(a) 10

(b) 20

(c) 30

(d) 0

Q.9 Bit-fields will be accommodated in a word
(a) from left to right
(b) from right to left
(c) in a way that depends on the implementation
(d) none of the above

Q.10 Which of the following is not a low-level feature of C?
(a) Register storage class
(b) Bit-fields
(c) Bit-wise operations
(d) None of the above

Q.11 The declaration
enum cities{bethlehem, jericho, nazareth = 1, jerusalem}
assigns the value 1 to
(a) bethlehem
(b) nazareth
(c) bethlehem and nazareth
(d) jericho and nazareth

Q.12 Choose the correct statements
(a) enum variables can be assigned new values.
(b) enum variables can be compared
(c) Enumeration feature does not increase the power of C.
(d) Use of enumeration enhances the logical clarity of a program.

Q.13 It is not advisable to use macros instead of functions because
(a) It increases the code size
(b) no type checking will be done
(c) recursion is not possible
(d) pointers cannot be used with macro identifiers.

Q.14 Use of macro instead of function is recommended
(a) when one wants to reduce the execution time
(b) when there is a loop with a function call inside
(c) when a function is called in many places in a program
(d) in all the above cases

LEVEL–2

Q.15 Consider the following declarations
```
union abc
{   char color;
    int size;
}
struct xyz
{   char cout;
    int date;
    union abc,id;
}   flag;
```
To assign a color to a flag, the correct statement would be
(a) flag.id.color ='white'
(b) flag. abc color = 'w'
(c) flag. color = 'white'
(d) flag.color = 'w'

Q.16 Consider the following declaration
```
struct list {
    int x;
struct list*next;
    }*head;
```
The statement head x = 100
(a) is an erroneous statement
(b) creates a node of type list and assigns a value of x
(c) creates a head of the list
(d) assigns 100 to one element of the structure list

Q.17 The declaration
```
union XYZ
{   char color [12];
    int size;
}   S,P;
```
denotes S and P are variable of type XYZ and
(a) each can have a value of color and size
(b) each can represent either a 12 character color or an integer size at a time
(c) S, P are same as struct variables
(d) variables S and p cannot be used simultaneously in a statement.

Q.18 Trace the output

```
main ()
{   int v = 3; int * pv;
    pv =& v;
    printf("pv=%d    v=%d", *pv,v);
    *pv = 0;
    printf("*pv=%d   v=%d", *pv,v);
}
```

(a) 3 3 3 0

(b) 3 0 3 0

(c) 3 3 0 0

(d) None of these

Q.19 Trace the output:

```
# include <stdio.h>
main()
{ union
        { int i;
        float f;
        double d;
    }   u;
printf("%d\n",sizeof(u));
u.i = 100;
printf("%d%f\n",u.i,u.f,u.d);
u.f = 0.5;
printf("%d%f%f\n",u.i,u.f,u.d);
u.d =0.0166667;
printf("%d%f\n",u.i,u.f,u.d);
}
```

Which of the following statemtn is correct

(i) The output of first printf is 8

(ii) The output of second printf is , 100 and Gabage value

(iii) The output of thirds and fourth only contains 0.5000000 and 0.0166667 as legal values other may be garbage values.

(a) (i)

(b) (i),(ii)

(c) (i),(iii)

(d) (i),(ii),(iii)

Q.20 How will you free the memory allocated in programme()?

Consider the program :

```
#include <stdio.h>
#include <stdlib.h>
struct emp
{
    int len;
    char *name;
};
int main()
{
    char newname[] = "Rahul";
    struct emp *p = (struct emp*) malloc (sizeof
    (struct emp));
    p -> len = strlen (newname);
    p -> name = malloc(p -> len + 1);
    strcpy (p ->name, newname);
    printf ("%d %s\n", p ->len, p -> name);
    return 0;
}
```

(a) free (p -> name);
    free (p);

(b) free (p →len);

(c) free (p → name);

(d) free (p → newname);
    free (p);

Common Data For Questions 21 to 24:

```
struct addr
{
char city[10];
char street [20]
int pincode;
};
struct
{
char name [20];
int sex;
struct addr locate ;
} criminal, *kd = & criminal;
```

Q.21 Sex can be accessed by
(a) criminal.sex
(b) kd → sex
(c) (*kd). sex
(d) either (a) or (c), but not by (b)

Q.22 pincode can be accessed by
(a) criminal. locate.pincode
(b) criminal.pincode
(c) kd → locate.pincode
(d) kd.locate → pincode

Q.23 The third character in the criminal name can be accessed by
(a) criminal. name[2]
(b) kd → name[2]
(c) ((*kd). name)[2]
(d) either (b) or (c), but not by (a)

Q.24 *(kd → name + 2) can be used instead of
(a) *(criminal. name + 2)
(b) kd → (name + 2)
(c) *((*kd). name +2)
(d) either (a) or (b), but not (c)

Q.25 Which of the following statements are correct about the program given below?

```
#include <stdio.h>
int main()
{
    union a
    {
        int i ;
        char ch[2];
    };
    union a u1 = {512};
    union a u2 = {0, 2};
    return 0;
}
```

(a) u2 CANNOT be initialized as shown.
(b) u1 can be initialized as shown.
(c) To initialize char ch [ ] of u2 '.' operator should be used.
(d) The code causes an error 'Declaration syntax error'.

Q.26

```
#include <stdio.h>
int main ()
{
    struct s1
    {
        char *str;
        int i;
        struct s1 *ptr;
    };
    static structs s1 a[] = {
                            {"Nagpur", 1, a+1},
                            {"Raipur", 2, a+2},
                            {"Kanpur", 3, a}
                            };
    struct s1 *p = a;
    int j;
    for (j = 0; j <= 2; j++)
    {
        printf ("%d", --a[j].i);
        printf ("%s\n", ++a[j].str);
    }
    return 0;
}
```

(a) 0 agpur
1 aipur
2 anpur
(b) Nagpur
Raipur
Kanpur
(c) 2 agpur
1 aipur
0 anpur
(d) 2 Kanpur
1 Jaipur
0 Nagpur

Q.27
```
#include <stdio.h>
int main ()
{
    struct a
    {
        char ch[7];
        char*str;
    };
    struct b
    {
        char*c;
        struct a ss1;
    };
    static struct b s2 = {"Raipur", "Kanpur", "Jaipur"};
    printf ("%s %s\n", s2.c, s2.ss1.str);
    printf ("%s %s\n", ++s2.c, ++s2.ss1.str);
    return 0;
}
```
(a) Raipur Jaipur
aipur aipur

(b) Raipur Jaipur
Raipur Jaipur

(c) Raipur Jaipur
R J

(d) None of the above

GATE QUESTIONS

Q.28 Consider the following C declaration
```
struct node {
        int i;
        float j;
};
struct node *s [10];
```
defines s to be **[GATE 2000] [2-Marks]**

(a) An array, each element of which is a structure of type node

(b) A structure of 2 fields, each field being a pointer to an array of 10 elements

(c) A structure of 3 fields: an integer, a float , and an array of 10 elements

(d) An array, each element of which is a pointer to a structure of type node

Q.29 Consider the function f defined below.
```
struct item {
        int data;
        struct item * next;
};
int f(struct item *p) {
        return((p==NULL)||(p->next==NULL)||((p->data<=p->next->data)&&f(p->next)));
}
```
For a given linked list p, the function f returns 1 if and only if **[GATE 2003] [2-Marks]**

(a) the elements in the list are sorted in non-increasing order of data value

(b) not all elements in the list have the same data value

(c) the elements in the list are sorted in non-decreasing order of data value

(d) the list is empty or has exactly one element

Q.30 Consider the following C program segment
```
structCellNode {
        structCellNode    *leftChild ;
        int element;
        structCellNode    *rightChild ;
        };
int DoSomething (structCellNode*ptr)
{
    int value = 0;
    if(ptr!=NULL)
        { if (ptr->leftChild! = NULL)
            value = 1 + DoSomething (ptr->leftChild);
          if (ptr->rightChild! = NULL)
            value = max(value, 1 + DoSomething(ptr->rightChild));
    }
return(value),
}
```
The value returned by the function DoSomething when a pointer to the root of a non-empty tree is passed as argument is **[GATE 2004] [2-Marks]**

(a) The height of the tree

(b) The number of nodes in the tree

(c) The number of internal nodes in the tree

(d) The number of leaf nodes in the tree

Q.31 Consider the following C program:

```
#include <stdio.h>
typedef struct {
                    char *a;
                    char *b:
                    } t;
void f1 (t s);
void f2 (t *p);
main ( )
{
                    static t s = {"A","B"};
                    printf("%s %s \n", s.a,
                    s.b);
                    f1 (s);
                    printf("%s %s \n", s.a,
                    s.b);
                    f2 (&s);
}
void f1 (t s)
{
                    s.a = "U";
                    s.b = "V",
                    printf("%s %s \n", s.a,
                    s.b);
                    return;
}
void f2 (t *p)
                    p → a = "V";
                    p → b = "W";
                    printf("%s %s \n", p →
                    a, p → b);
                    return;
                    }
```

What is the output generated by the program?

[IT-GATE 2004]

[2-Marks]

(a) AB
UV
VW
VW

(b) AB
UV
AB
VW

(c) AB
UV
UV
VW

(d) AB
UV
VW
UV

ANSWER KEY

1	b	2	d	3	b	4	c	5	a
6	a	7	c	8	b	9	c	10	d
11	d	12	all	13	all	14	a, b	15	a
16	a	17	b	18	c	19	b	20	a
21	a, b, c	22	a, c	23	a, b, c	24	a, c	25	a, b, c
26	a	27	a	28	d	29	b	30	a
31	b								

SOLUTIONS

S.2 (d)

Definition of array of structures.

S.3 (b)

If there is no space to accommodate the entire bit field, it will be completely shifted to the next word.

S.4 (c)

ABC [10] is an array of structures

⇒ It is a collection of 10 structure variables.

Each structure variable needs 2+4+8 = 14 bytes of memory.

Hence total bytes needed = 14×10=140 for 10 variables.

S.5 (a)

In B semicolon is missing after the declaration.
In C semicolon is missing after each structure element.

S.6 (a)

The program allocates space for the structure with the size adjusted so that the name field can hold the requested name.

S.7 (c)

Structures can't be compared using the built-in == and ! = operations.

S.8 (b)

Union share a common memory area, which is over written by v.b. = 20, Hence 20.

S.9 (c)

Bit-field is a member of structure whose size is specified in terms of bits & is implementation dependent.

S.11 (d)

The listed places will be assigned the values 0, 1, 1, 2 respectively.

S.15 (a)

In given declaration we are using union as well as struct, and to assign a color to a flag the correct statement is **flag.id.color = 'White'.**

S.16 (a)

Head is a pointer variable of type struct list.

Hence assigning a value 100 (which is integer) is illegal.

S.17 (b)

S and P are variable so type XYZ and each can represent either a 12 character color or an integer size at a time.

S.18 (c)

The program begins by assgining an initial value of 3 to the integer variable v and then assigns the address of v to the pointer variable pv. Thus, pv becomes a pointer to v.

∴ *pv = 3, v=3

and

***pv=0, v=0**

S.19 (b)

(i) ⇒ size of union ⇒ maximum datatype size ⇒8

(ii) ⇒ due to u.i = 100⇒100 is printed and other values are garbage.

(iii) ⇒ due to u.f and u.d the 0 500000 and 0.016667 is printed and other garbage values are printed.

S.21 (a, b, c)

structure variable could be accessed by using '.' or → operater

S.22 (a, c)

Inheritance property of structures.

S.25 (a, b, c)

The C standard allows an initializer for the first member of a union. There is no standard way of initializing any other member, hence the error in initializing u2

If you still want to initialize different members of the union then you can define several variant copies of a union, with the members in different orders, so that you can declare and initialize the one, having the appropriate first member as shown below.

```
#include <stdio.h>
union a
{
    int i ;
    char ch[2];
};
union b
{
    char ch[2];
    int i ;
};
int main()
{
    union a u1 = {512};
    union b u2 = {0, 2};
    return 0;
}
```

S.26 (a)

Output

0 agpur

1 aipur

2 anpur

Explanation:

The example deals with a structure similar to the one we just encountered. Picking up from the **for** loop, it is executed for 3 values of **j**: 0, 1 and 2. The first time through the **for** loop, **j** is equal to zero, so the first **printf()** prints **--a[0].i**. Since the dot operator has a higher priority, first **a[0].i** is evaluated, which is 1. As - precedes the value to be printed, 1 is first decremented to 0, and then printed out.

The second **printf()** prints the string at address **++a[0].str.a[0].str** gives the starting address of "Nagpur". On incrementing, it points to the next character, 'a' of "Nagpur", so starting from 'a', the remaining sting "agpur" is outputted.

A similar procedure is repeated for **j = 1**, and then once again for **j = 2**, following which the execution is terminated.

S.27 (a)

Output

Raipur Jaipur

aipur aipur

Explanation:

At the outset, **struct a** is declared to comprise of a character array **ch[]** and a character pointer **str**. Next, **s2** is declared as a variable of type **struct b**, which is made up of a **char** pointer **c** and another variable **ss1** of type **struct a**. While initializing **s2**, the base address of the string "Raipur" is assigned to **s2.c**, "Kanpur" is assigned to **s2.ss1.ch[ ]**, and the base address of "Jaipur" is assigned to **s2.ss1.str**.

Combing to the **printf()**s now, the first one is supplied **s2.c** and **s2.ss1.str**. **s2.c** gives the base address of "Raipur", and **s1.ss1.str** gives the base address of "Jaipur". Since these base address are passed to **printf()**, it promptly prints out the two strings.

The second **printf()** uses incremented values of these addresses. On incrementing **s2.c** using the ++ operator, it now points to the next element 'a' of "Raipur". Similarly, on incrementing **s2.ss1.str**, it points to the 'a' of "Jaipur". With these as starting addresses, the remaining strings are printed out.

S.28 (d)

struct node *s[10];

is array of pointer. **Each pointer is of type struct node i.e. each pointer is for structure node.**

S.29 (b)

For a given linked list p, **the function f returns 1 if and only if not all elements in the list have the same data value** because if p==NULL or p->next ==NULL or p->data<=next->data and p->next.

S.30 (a)

The function DoSomething is a recursive function. DoSomething (ptr → left child) computes total number of node (edges) in left subtree recursively and the count stored in the variable value when we will add 1 then it includes the root nodes. Same thing exist for right subtree. Finally **max (value, t + DoSomething (ptr → rightchile)) return the number of edges in either left subtree or right subtree which is the height of a tree.**

S.31 (b)

AB
UV
AB
VW

Here, function 'f1' & 'f2' perform the same function, but f1 without pointer while f2 with pointer.

4.6

OOPs

LEVEL–1

Q.1 Consider the following declarations

```
class shape
{   public:
    ------
virtual void draw ( )=0;
    ------
};
```

Select the incorrect statement

(a) The compiler will not permit the user to instantiate an instance of a class that contains a pure virtual method

(b) The redefinition of a pure virtual method must be handled by an immediate subclass

(c) The deferred method must be declared explicitily by use of the virtual keyword

(d) (a) (b) (c) are incorrect

Q.2 Which of the following are good reasons to use an object orinted language?

(a) you can define you own data type

(b) an object oriented program can be taught of correct its own errors

(c) it is easier to conceptualize an object

(d) both (a) and (c)

Q.3 In object orient programming classes are useful because they

(a) bring together all aspects of an entity in one place

(b) permit data to be hidden from other classes

(c) can closely model objects in the real word

(d) all of these

Q.4 Runtime polymorphism can be achived by

(a) accessing virtual function through the pointer of base class

(b) by accessing virtual function through the object

(c) none of these

(d) both (a) and (b)

Q.5 Which one of the following is correct

(a) There can be any number of constructors and destructors

(b) Constructor can't be overloaded

(c) Destructors do not have retrun values

(d) Constructors return values

Q.6 Virtual function should be defined in ..... section of a class.

(a) protected

(b) public

(c) private

(d) anywhere

Q.7 Which one of the following is pure virtual function?

(a) virtual void funct () =0;

(b) virtual void funct (int n);

(c) virtual funct (int n) ;

(d) virtual void funct (int n)= 0;

Q.8 Method Overloading is

(a) Overloading different member function of a class

(b) Overloading without argument passing

(c) A feature in which multiple functions with same name and different signature.

(d) none of these

Q.9 Static member functions

(a) can access only static members in the same class

(b) has only one copy of it exists for all instances of a class

(c) can access constant members

(d) both (b) and (c)

Q.10 Static class member's definitions and intializations occurs

(a) inside the class

(b) inside the objects of the class

(c) outside the class and function bodies

(d) anywhere

Q.11 A pointer to a base class type can refer to an object of a derived class type. This is called as

(a) encapsulation

(b) memory addressing

(c) polymorphism

(d) inheritance.

Q.12 Constructor should be defined in ..... section of a class

(a) public

(b) private

(c) protected

(d) Abstract data type

Q.13 The argument of a copy constructor is passed by

(a) value

(b) reference

(c) pointer

(d) both (a) and (b)

Q.14 A function that can access private members of a class, even though it is not a member of the class itself, is

(a) a friends

(b) a private function

(c) an inline function

(d) never allowed in object oriented programming

Q.15 To perform multiple arithmetic operations in a single statement, overloaded operator functions should return

(a) a copy of the values passed to them

(b) void

(c) an object of the class type

(d) address of the function

Q.16 Inheritance is the priniciple that

(a) classes with the same name must be derived from one another

(b) knowledge of a general catagory can be applied

(c) one functions name may invoke different methods

(d) C++ functions may be used only if they have logical predecessors

Q.17 A default constructor

(a) has default values for all its arguments

(b) takes no arguments

(c) either (a) or (b)

(d) none of these

Q.18 Advantages of inheritance include (more than once)

(a) facilitating class libraries

(b) avoiding the rewriting of code

(c) providing a useful conceptual framework

(d) providing class growth through nataul selection

Q.19 A function in a derived class that has the same name as a function in the parent class will

(a) override the base class function

(b) be overridden by the base class function

(c) cause an error message to display

(d) execude immediately often the base class function executes.

Q.20 Which one of the following is incorrect?

(a) The data-centred design approach enables us to capture more details of a model in implementation form.

(b) It is possible to have multiple instances of an object to co-exist without any interference.

(c) We can build programs from the standard working modules that communicate with one another, rather than having to start writing the code from scratch.

(d) Message passing techniques for communication between objects makes the interface descriptions with external system is not much simpler.

Q.21 Select the correct statement about object oriented programming (OOP)?

(a) OOP allows us to decompose a problem into a number of entities object and then builds data and functions around these entities.

(b) OOP treats data as a critical element in the program development and does not allow it to flow freely around the system.

(c) OOP ties data more closely to the functions that operate on it and protect it from accidental modification from outside functions.

(d) All of the above

Q.22 Match the following

Group I		Group II	
(a)	The entire set of data and code of an object can be made a user-defined data type	(i)	Polymorphism
(b)	The data is not accessible to the outside world and only those functions which are wrapped in the class can access it	(ii)	Inheritance
(c)	The process by which objects of one class acquire the properties of objects of another class	(iii)	Class
(d)	Allowing objects having different internal to share the same external interface	(iv)	Encapsulation

(a) a – ii, b – iii, c – iv, d – i

(b) a – iii, b – iv, c – ii, d – i

(c) a – iii, b – iv, c – i, d – ii

(d) a – ii, b – iii, c – i, d – iv

Q.23 Assume a class C with obj1, obj2 and obj3. For the statement obj3 = 1 – obj2 to work correctly, the overloaded operator must

(a) use the object of which it is member as an operand

(b) return a value

(c) create a named temporary object

(d) both (a) and (b)

Q.24 Inheritance is a way to

(a) improve data hiding and encapsulation

(b) pass arguments and improve data hiding

(c) pass arguments and add features to existing classes without rewriting them

(d) make general classes into more specific classes and add features to existing classes without rewriting them

Q.25 Derivation of a class from other derived class is called ..... inheritance.

(a) Hybrid

(b) Multilevel

(c) Multiplath

(d) None of these

Q.26 The programming language feature that allows the same operation to be carried out differently depending on the object is

(a) information hiding

(b) allocation

(c) inheritance

(d) polymorphism

Q.27 Object-oriented programmers primarily focus on

(a) the physical orientation of objects within a program

(b) objects and the tasks that must be performed with those objects

(c) the step-by-step statements needed to solve a problem

(d) procedures to be performed

Q.28 A major advantage of inheritance is

(a) enabling people who have not studied programming to create useful applications.

(b) reducing the amount of memory required to execute a program

(c) not having to think about how objects will be used

(d) reducing the time it takes to create new objects

Q.29 Which of the following statement is false?

(a) A class is considered an object

(b) An example of a behaviour is the set function in a time class

(c) An example of an attribute is the minutes variable in a time class

(d) A class encapsulates all of an object's attributes and behaviours

Q.30 Variable names known only to the procedure in which they are declared are

(a) internal

(b) recent

(c) global

(d) local

Q.31

```
main ( )
{
        int a = 6, b = 10, x ;
        x = a | b ;
        cout << x < "\n";
}
```

Find the value of x.

(a) 15

(b) 10

(c) 12

(d) 14

Q.32 If a global variable is of storage class static, then

(a) the static declaration is unnecessary if the entire source code is in a single file

(b) the variable is recognized only in the file in which it is defined

(c) it results in a syntax error

(d) none of the above

Q.33 Choose the correct statement.

(a) The scope of a macro definition need not be the entire program.

(b) The scope of a macro definition extends from the point of definition to the end of the file

(c) New line is a macro definition delimiter.

(d) A macro definition may go beyond a line.

Q.34 If many functions have the same name, which of the following information, if present, will be used by the compiler to invoke the correct function to be used?

(a) The return value of the function

(b) The operator : :

(c) Function signature

(d) None of the above

Q.35 A constructor is called whenever

(a) a class is used

(b) a class is declared

(c) an object is used

(d) an object is declared

Q.36 `class Dog : public X, public Y` is an example of

(a) linear inheritance

(b) multiple inheritance

(c) repeated inheritance

(d) none of the above

Q.37 Overloading is otherwise called as

(a) ad-hoc polymorphism

(b) pseudo polymorphism

(c) virtual polymorphism

(d) transient polymorphism

LEVEL–2

Q.38 Which one of the following is not valid?

(i) int mul (int i, int j = 5, int k = 10)

(ii) int mul (int i = 5, int j)

(iii) int mul (int i = 0, int j, int k = 10)

(iv) int mul (int i = 2, int j = 5, int k = 10)

(a) (i), (ii), (iii)

(b) (ii), (iii)

(c) (i), (ii)

(d) (i), (iii)

Q.39
```
main ( )
{
        int k = 16, x, f;
        f = k % x ;
        cout << f << endl ;
}
```

(a) 0

(b) undefined

(c) 2

(d) 1

Q.40 Consider the following statements:

I. For each object different copy of static data members, available like normal data members.

II. Static data members must be defined and initialized like global variables, otherwise linking error will result.

Of these statements

(a) II is true and I is false

(b) I is true and II is false

(c) both I and II are false

(d) both I and II are true

Q.41 Consider the following statements

Statement 1: Constructor of a base class executed first and then constructor of derived class.

Statement 2: If the base class has constructors with arguments then it's not necessary for the derived class to have constructor.

Statement 3: If the base class do not have default constructor, they must be explicitly invoked.

Statement 4: Virtual base class constructor should be invoked first and then orderly invocation of constructors.

Statement 5: The derived class need not have a constructor as long as base class has a no-argument constructor.

Which of these statements are correct:

(a) All statements are correct.

(b) 1, 2, 3

(c) 1, 2, 4, 5

(d) 1, 3, 4, 5

Q.42 Which of the following is the best prototype to overload the << operator for a Number class?

(a) friend ostream operator << (const Number & num)

(b) Number & operator << (ostream & out, const Number & num);

(c) friend ostream & operator << (ostream & out, const Number & num);

(d) ostream & << (ostream & out, const Number & num)

Q.43 If class P is a parent to child class C, and C inherits with public access, P's function F( ) can be made private with which statements?

(a) private:
C::F;

(b) private:
C::F( );

(c) private:
P::F;

(d) Private:
P::F( );

Q.44 If you declare two objects as
Customer first Cust, second Cust
which of the following must be true?

(a) Each object will store a separate copy of any nonstatic data members.

(b) You cannot declare two objects of the same class.

(c) Each object will store a separate copy of any static member data.

(d) Each object will store a separate copy of any member fuction.

Q.45 Student senior ( ); is *a(n)*

(a) constructor call with all default arguments

(b) prototype for a function that returns a student objects.

(c) constructor call with no arguments

(d) object instantiation

Q.46 The prototype for the constructor for a Student class is
Student (const int *id* Num = 0, const double gpa = 4.0); The definition Student Lynette (1, 2);

(a) is illegal

(b) defines a student with *id*Num 1 and gpa 2.0

(c) defines a student with *id*Num 0 and gpa 1.0

(d) defines a student with *id*Num 0 and gpa 4.0

Q.47 Consider the following declaractionn in C++

enum color {red, blue, grren , yellow};

(i) color background = blue;

(ii) color background = 1;

(iii) color background = (color);

Which one of the following is a correct statement?

(a) only (i)

(b) (i),(ii)

(c) (i),(ii)

(d) (i),(ii),(iii)

Q.48 Consider the following declarations

---

p=new int;

if (!p)

{ cout<<"_____",

} ---

If 'if condition is valid then the output statements is _____

(a) Allocation successfull

(b) Allocation failed

(c) Error

(d) None of thhhhese

Q.49 A friends function

(a) receivers a this pointer for the class that makes it a friend

(b) receives a this pointer for any class object pases to it

(c) receives a this pointer for any class object passess to it

(d) does not have a this pointer for the clas that makes it a friend.

Q.50 Which one is the correct statement about procedure oriented programming language

(i) Emphasis is on doing things

(ii) Most of the function from program share global data

(iii) Data move openly around the system from functions to function

(iv) Employees top-down approach in program design

(a) (i), (ii) & (iv)

(b) only (iI, (ii)

(c) (i), (ii), (iii) & (iv)

(d) (ii), (iv)

Q.51 Match the following

Group I		Group II	
(i)	::*	(a)	To access a member using a pointer to the object and a pointer to that member
(ii)	*	(b)	To declare a pointer to a member of a class
(iii)	→*	(c)	To access a member using object name and a pointer to that member

(a) (i) – c, (ii) – a, (iii) – b

(b) (i) – b, (ii) – a, (iii) – c

(c) (i) – b, (ii) – c, (iii) – a

(d) (i) – c, (ii) – b, (iii) – a

Q.52 Which of the following comments about EOF are true?

(a) Its value is defined within stdio.h

(b) Its value is implementation dependent.

(c) Its value can be negative

(d) Its value should not equal the integer equivalent of any character.

Q.53 Let class APE be a friend of class SAPIEN. Let class HUMAN be a child class of SAPIEN and let MONKEY be a child class of APE. Then

(a) APE is not a friend of HUMAN

(b) MONKEY is not a friend of SAPIEN

(c) SAPIEN is not a friend of APE

(d) none of the above

GATE QUESTIONS

Q.54 Consider the following class definitions in a hypothetical Object Oriented language that supports inheritance and uses dynamic binding. The language should not be assumed to be either Java or C++, though the syntax is similar.

```
Class P {                    Class Q subclass of P {
    void f (int i) {             void f (int i ) {
    print (i) ;                  print (2*i);
    }                                }
}                                }
```

Now consider the following program fragment:

Px = new Q (); Qy = new Q () ;

Pz = new Q (); x.f(1);((P)y).f(1);z.f(1);

Here ((P)y) denotes a typecast of y to P. The output produced by executing the above program fragment will be **[GATE 2003]**

[2-Marks]

(a) 2 2 2

(b) 2 1 1

(c) 2 1 2

(d) 1 2 1

Q.55 Which of the following are essential features of an object-oriented programming languages? **[GATE 2005]** **[1-Mark]**

1. Abstraction and encapsulation
2. Strictly-typedness
3. Type-safe property coupled with sub-type rule
4. Polymorphism in the presence of inheritance

(a) 1, 3 and 4 only
(b) 1, 2 and 4 only
(c) 1 and 4 only
(d) 1 and 2 only

Q.56 An abstract Data Type (ADT) is **[GATE 2005]** **[1-Mark]**

(a) a data type that cannot be instantiated
(b) a data type for which only the operations defined on it can be used, but none else
(c) same as an abstract class
(d) all of the above

ANSWER KEY

1	d	2	c	3	d	4	a	5	c
6	d	7	a	8	c	9	a	10	c
11	c	12	a	13	b	14	a	15	c
16	b	17	a	18	a, b, c	19	a	20	d
21	d	22	b	23	d	24	d	25	b
26	d	27	b	28	d	29	a	30	d
31	d	32	a, b	33	all	34	b, c	35	d
36	b	37	a	38	b	39	b	40	c
41	c	42	c	43	b	44	a	45	b
46	d	47	c	48	b	49	d	50	c
51	c	52	all	53	a, b, c	54	c	55	a
56	b								

SOLUTIONS

S.1 (d)

(a) (b) (c) are incorrect statements about deferred methods or pure virtual functions in C.

S.20 (d)

(a), (b) & (c) are correct statement.

But in message passing techniques, **the interface descriptions with external system is much simpler.**

S.21 (d)

(a), (b), (c) are properties of OOP.

S.22 (b)

Definitions of (i), (ii), (iii), (iv)

S.31 (d)

$a \equiv d \Rightarrow \quad 0110$
$b=10 \Rightarrow \quad 1010$
$\mathbf{x=a|b \Rightarrow \quad 1110 \Rightarrow 14}$

S.38 (b)

(ii), (iii) are invalid because we cannot provide a default value to a particular argument in the middle of an argument list. It must be from right to left in an argument list.

S.39 (b)

f= x nod x

Here x in $\frac{\text{undefined}}{\text{uninitialized}} \Rightarrow$ **value of is undefined.**

S.40 (c)

All the objects are having the some copies of the static data member. ⇒ given statement I is false.

Again

Static data members should be defined inside the class by using static keyword by data type and variable name, as well as outside the class using scope resolution operator.

⇒ not like the global variables

⇒ **both are false.**

S.47 (c)

In C++, each enumerated datatype retains its own separate type. This means that **C++ does not permit an int value to be automatically converted to an enum value.**

But an enumerated value can be used in place of an int value.

S.48 (b)

The given code check for memory available. If sufficient memory is not available then malloc (), new returns a null pointer.

S.50 (c)

(i), (ii), (iii) & (iv) are characteristics of procedure oriented programming language.

S.51 (c)

(i), (ii), (iii) are member dereferencing operators and (b), (c), (a) are there respective functions

S.52 (a, b, c, d)

EOF represents end of file, which is define in stdio.h, depends on user but cann't be equal to character value.

S.55 (a)

Object oriented programming language is

Object based PL + Abstraction + Inheritance

The last wo are must for any L to be OOPL.

UNIT–5

DATA STRUCTURES AND ALGORITHMS

UNIT-5

DATA STRUCTURES AND ALGORITHMS

5.1

ARRAY AND QUEUE

LEVEL–1

Q.1 Let P be a singly linked list. Let Q be the pointer to an intermediate node x in the list. What is the worst case time complexity of the best known algorithm to delete the node x from the list?

(a) O(n)

(b) $O(\log^2 n)$

(c) O(logn)

(d) O(1)

Q.2 Suppose each data structure is stored in a circular array with N memory cells, then find the number NUMB of elements in a dequeue in terms of LEFT and RIGHT.

(a) RIGHT– LEFT + 1 (mod n)

(b) RIGHT + LEFT – 1 (mod n)

(c) RIGHT + LEFT + 1 (mod n)

(d) RIGHT – LEFT – 1 (mod n)

Q.3 If a queue is implemented using an array then what are the worst case time complexities of both queue and dequeue operations.

(a) O(1), O (n)

(b) O (n), (1)

(c) O (1), O (1)

(d) O (n), O (n)

Q.4 Consider the following unsorted array.

25 57 48 37 12 92 86 33

The fourth pass of bubble sort will give

(a) 12 25 37 33 48 57 86 92

(b) 25 12 37 33 48 57 86 92

(c) 12 25 33 37 48 57 86 92

(d) 12 25 33 37 48 57 92 86

Q.5 Which is the prototype of a priority queue:

(a) Timesharing system

(b) Batch processing system

(c) Multitasking system

(d) None of these

Q.6 Consider a dequeue maintained by a circular array with N memory cells. When an element is added to the dequeue, then what is the process of right and if element is deleted, then what is the process of right?

(a) decreased by (mod N), increased by 1

(b) decreased by 1, not affected

(c) increased by 1 (mod N), not affected

(d) increased by 1 (mod N), decreased by 1 mod (N)

Q.7 The binary search algorithm can only be used if the table is stored as an _____

(a) array

(b) stack

(c) list

(d) none of these

Q.8 When the queue is initialized, the value of rear will be equal to

(a) one is greater and equal to the value of front

(b) one greater than value of front

(c) one less than the value of front

(d) value of front

Q.9 The element being searched for is not in an array of 100 elements. What is the average number of comparisons needed in a sequential search to determine that the element is not there if the elements are ordered from largest to smallest?

(a) 75

(b) 50

(c) 100

(d) 30

Q.10 Which of the following can be described as a array?

(a) List

(b) Stack

(c) List ^ Info

(d) Stack ^ Info

Q.11 Let A be a dimensional array declared as follows;

A array [1....10] [1.....15] of integer ;

Assuming that each integer takes one memory locations the array is stored in row major order and the first element of the array is stored at location 100, what is the address of the element A [i] [j]?

(a) 15 i + j+84

(b) 15 j + i+84

(c) 10 i + j+89

(d) None of these

Q.12 Faster access to non local variables is achieved using an array of pointers to activation records, called a

(a) heap

(b) activation tree

(c) display

(d) stack

Q.13 If int s [s] is one dimensional array of integers, which of the following refers to the third element in the array?

(a) *(s+2)

(b) s+3

(c) *(s+3)+5

(d) *(s+3)+2

LEVEL–2

Q.14 Suppose you are given an implementation of a queue of integers. The operations that can be performed on the queue are:

(i) is Empty (Q)- returns true if the queue is empty, false otherwise,

(ii) delete (Q)- deletes the element at the front of the queue and returns its value,

(iii) insert (Q, i)- inserts the integer i at the rear of the queue.

Consider the following function:

```
void f (queue Q)      { int  i;
if  (!is Empty  (Q))   { i = delete (Q);
f(Q);
insert (Q, i);
} }
```

What operation is performed by the above function f ?

(a) Leaves the queue Q unchanged

(b) Reverses the order of the elements in the queue Q

(c) Deletes the elements at the front of the queue Q and inserts it at the rear keeping the other elements in the same order.

(d) Empties the queue.

Common Data For Questions 15 & 16:

Consider a binary max-heap implemented using an array.

Q.15 Which one of the following array represents a binary max-heap ?

(a) {25,12,16,13,10,8,14}

(b) {25,14,13,16,10,8,12}

(c) {25,14,16,13,10,8,12}

(d) {25,14,12,13,10,8,16}

Q.16 Consider the implementation of the stack using partially filled array. What goes wrong if we try to store the top of the stack at location [0] and the bottom of the stack at the last position of the array.

(a) Both push and pop would require non-linear time.

(b) Both push and pop would require linear time.

(c) The stack could not be used to check balanced parentheses.

(d) The stack could not be used to evaluate postfix expressions.

Q.17 Suppose we are sorting an array of eight integers using quick sort, and we have just finished the first partitioning, with the array looking like this.

2 5 1 7 9 12 11 10

which statement is correct?

(a) The pivot could be either 7 or 9

(b) The pivot could be 7, but it is not 9

(c) The pivot is not 7, but is could be 9

(d) Neither 7 nor 9 is the pivot

Q.18 An unordered list which is used as a priority queue, requires examining how many nodes for insertion and how many for deletion (with n nodes).

(a) one node, n/2 nodes

(b) n nodes, n nodes

(c) one node, n nodes

(d) n/2 nodes, n/2 nodes.

Q.19 How many values can be held by an array defined as

A(–1..n,1..n)?

(a) n (n–1)

(b) n^2 (n+1)

(c) n(n+1)

(d) n (n+2)

Q.20 Suppose a dequeue is stored in a circular array with N memory cells. At which of the following conditions is the dequeue full?

(i) LEFT = N and RIGHT =1

(ii) LEFT = RIGHT + 1

(iii) LEFT =1 and RIGHT = N

(iv) LEFT = RIGHT – 1 + N

(a) (i), (iii)

(b) (iii), (iv)

(c) (ii), (iii)

(d) (i), (iv)

Q.21 Consider array X with the following elements:

26 58 49 38 13 93 87 34

If we apply selection sort on the above data, which of the following statements is correct?

(a) The above list can be sorted in 6^{th} pass

(b) The above list can be sorted in 5^{th} pass

(c) The above list can be sorted in 7^{th} pass

(d) The above list can be sorted in 4^{th} pass

Q.22 Consider the following unsorted array:

22 55 44 66 11 99 88 33

If the value of d is 4, then the second pass of quick sort would give.

(a) 22 55 44 66 11 99 88 33

(b) 11 55 44 33 22 99 88 66

(c) 11 33 22 55 44 66 88 99

(d) 11 22 33 44 55 99 88 66

GATE QUESTIONS

Q.23 Suppose we want to arrange the n numbers stored in any array such that all negative values occur before all positive ones. Minimum number of exchanges required in the worst case is **[GATE 1999]**

[1-Mark]

(a) n–1

(b) n

(c) n+1

(d) None of the above

Q.24 Let s be a sorted array of n integers. Let t(n) denote the time taken for the most efficient algorithm to determine if there are two elements with sum less than 1000 in s. Which of the following statements is true? **[GATE 2000]**

[1-Mark]

(a) t(n) is 0 (1)

(b) $n \le t(n) \le n \log_2 n$

(c) $n \log_2 n \le t(n) < \left(\frac{n}{2}\right)$

(d) $t(n) = \left(\frac{n}{2}\right)$

Q.25 Consider an array representation of an n element binary heap where the elements are stored from index 1 to index n of the array. For the element stored at index i of the array ($i \le n$), the index of the parent is **[GATE 2001]**

[1-Mark]

(a) i – 1

(b) $\lfloor i/2 \rfloor$

(c) $\lceil i/2 \rceil$

(d) (i + 1)/2

Q.26 The best data structure to check whether an arithmetic expression has balanced parentheses is a **[GATE 2004]**

[1-Mark]

(a) queue

(b) stack

(c) tree

(d) list

Q.27 Suppose each set is represented as a linked list with elements in arbitrary order. Which of the operation among union, intersection, membership, cardinality will be the slowest?

[GATE 2004]

[2-Marks]

(a) union only

(b) intersection, membership

(c) membership, cardinality

(d) union, intersection

Q.28 A single array A[1..MAXSIZE] is used to implement two stacks. The two stacks grow from opposite ends of the array. Variables top 1 and top 2 (top 1< top 2) point to the location of the topmost element in each of the stacks. If the space is to be used efficiently, the condition for "stack full" is **[GATE 2004]**

[1-Mark]

(a) (top1 = MAXSIZE/2) and (top2 = MAXSIZE/2 + 1)

(b) top1 + top2 = MAXSIZE

(c) (top1 = MAXSIZE/2) or (top2 = MAXSIZE)

(d) top1 = top2 – 1

Q.29 Let A [1,...,n] be an array storing a bit (1 or 0) at each location, and f(m) is a function whose time complexity is θ(m). Consider the following program fragment written in a C like language;

```
counter = 0;
    for (i = 1; i ≤ n; i++)
    { if(A[i] = = 1) counter ++;
    else {f(counter); counter = 0;}
    }
```

The complexity of this program fragment is

[GATE 2004]

[2-Marks]

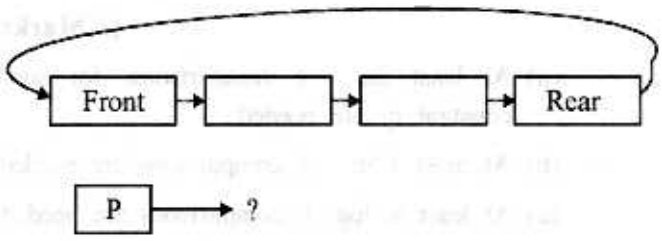
(a) $\Omega(n^2)$

(b) $\Omega(n \log n)$ and $O(n^2)$

(c) $\theta(n)$

(d) $O(n)$

Q.30 A circularly linked list is used to represent a Queue. A single variable p is used to access the Queue. To which node should p point such that both the operations on Queue and dequeue can be performed in constant time?



[GATE 2004]

[2-Marks]

(a) rear node

(b) front node

(c) not possible with a single pointer

(d) node next to front

Q.31 Two matrices M_1 and M_2 are to be stored in arrays A and B respectively. Each array can be stored either in row major or column major order in contiguous memory locations. The time complexity of an algorithm to compute $M_1 \times M_2$ will be: **[Gate 2004]**

[2-Marks]

(a) best if A is in row major and B is in column major order.

(b) best if both are in row major order

(c) best if both are in column major order.

(d) independent of the storage scheme.

Q.32 What is the number of vertices in an undirected connected graph with 27 edges, 6 vertices of degree 2, 3 vertices of degree 4 and remaining of degree 3? **[IT-GATE 2004]**

[2-Marks]

(a) 10

(b) 11

(c) 18

(d) 19

Q.33 Consider the following program module:

```
int module1 (int x, int y)
while (x!=y) {
    if (x>y)
        x=x-y;
    else y=y-x;
    }
return x;
}
```

What is Cyclomatic complexity of the above module? **[IT-GATE 2004]**

[2-Marks]

(a) 1

(b) 2

(c) 3

(d) 4

Q.34 Let a and b be two sorted arrays containing n integers each, in non-decreasing order. Let c be a sorted array containing 2n integers obtained by merging the two arrays a and b. Assuming the arrays are indexed starting from 0, consider the following four statements.

I. $a[i] \geq b[i] \Rightarrow c[2i] \geq a[i]$

II. $a[i] \geq b[i] \Rightarrow c[2i] \geq b[i]$

III. $a[i] \geq b[i] \Rightarrow c[2i] \leq a[i]$

IV. $a[i] \geq b[i] \Rightarrow c[2i] \leq b[i]$

Which of the following is TRUE?

[IT-GATE 2005]

[2-Marks]

(a) only I and II

(b) only I and IV

(c) only II and III

(d) only III and IV

Q.35 An element in an array X is called a leader if it is greater than all elements to the right of it in X. The best algorithm to find all leaders in an array.

[GATE 2006]

[1-Mark]

(a) Solves it in linear time using a left to right pass of the array

(b) Solves it in linear time using a right to left pass of the array

(c) Solves it using divide and conquer in time $\theta(n \log n)$

(d) Solves it in time $\theta(n^2)$

Q.36 A set X can be represented by an array x [n] as follows **[GATE 2006]**

[2-Marks]

$$x[i] = \begin{cases} 1 & \text{if } i \in X \\ 0 & \text{otherwise} \end{cases}$$

Consider the following algorithm in which x, y, and z are boolean arrays of size n:

```
algorithm zzz (x[], y[], z[]){
int i;
for (i = 0 ; i < n ; ++ i)
z[i] = (x [i] ∧ ~ y[i]) ∨ (~x [i] ∧y[i])
}
```

The set Z computed by the algorithm is

(a) $(X \cup Y)$

(b) $(X \cap Y)$

(c) $(X - Y) \cap (Y - X)$

(d) $(X - Y) \cup (Y - X)$

Q.37 Given two arrays of numbers $a_1 \ldots a_n$ and $b_1 \ldots b_n$ where each number is 0 or 1, the fastest algorithm to find the largest span (i, j) such that $a_i + a_{i+1} + \ldots + a_j = b_i + b_{i+1} + \ldots\ldots\ldots + b_j$, or report that there is no such span,

[GATE 2006]

[2-Marks]

(a) Takes $O(3^n)$ and $\Omega(2^n)$ time if hashing is permitted

(b) Take $O(n^3)$ and $\Omega(n^{2.5})$ time in the key comparison model

(c) Takes θ (n) time and space

(d) Takes O $(\sqrt{n})$ time only if the sum of the 2n elements is an even number

Q.38 An array of n numbers is given , where n is an even number. The maximum as well as the minimum of these n number needs to be determined. Which of the following is TRUE about the number of comparisons needed?

[GATE 2007]

[2-Marks]

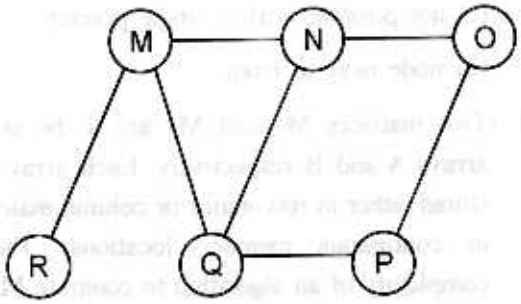
(a) At least 2n – c comparisons, for some constant c, are needed.

(b) At most 1.5n – 2 comparisons are needed.

(c) At least n, $\log_2$ n comparisons are needed.

(d) None of the above

Q.39 The Breadth First Search algorithm has been implemented using the queue data structure. One possible order of visiting the nodes of the following graph is



[GATE 2008]

[1-Mark]

(a) MNOPQR

(b) NQMPOR

(c) QMNPRO

(d) QMNPOR

Q.40 The minimum number of comparison required to determine if an integer appears more that n/2 times in a sorted array of n integers is

[GATE 2008]

[2-Marks]

(a) $\Theta(n)$

(b) $\Theta(\log n)$

(c) $\Theta(\log n^*n)$

(d) $\Theta(1)$

Q.41 The following C function takes a singly-linked list of integers as a parameter and rearranges the elements of the list. The list is represented as pointer to a structure. The function is called with the list containing the integers 1, 2, 3, 4, 5, 6, 7 in the given order. What will be the contents of the list after the function completes execution ?

```
struct node {
    int value;
    struct node *next;
};
Void rearrange(struct node *list) {
    struct node *p, *q;
    int temp;
    if (!list || !list -> next)return;
    p = list; q = list -> next;
    while(q){
        temp = p -> value; p -> value = q ->
        value;
        q -> value = temp; p = q -> next;
        q = p ? p -> next: 0;
    }
}
```

[GATE 2008]

[2-Marks]

(a) 1,2,3,4,5,6,7

(b) 2,1,4,3,6,5,7

(c) 1,3,2,5,4,7,6

(d) 2,3,4,5,6,7,1

Q.42 We have a binary heap on n elements and wish to insert n more elements (not necessarily one after another) into this heap. The total time required for this is **[GATE 2008]**

[2-Marks]

(a) $\theta(\log n)$

(b) $\theta(n)$

(c) $\theta(n \log n)$

(d) $\theta(n^2)$

Common Data For Questions 43 & 44:

A subsequence of a given sequence is just the given sequence with some elements (possibly none or all) left out. We are given two sequences X[m] and Y[n] of lengths m and n, respectively, with indexes of X and Y starting from 0.

Q.43 We wish to find the length of the longest common sub sequence (LCS) of X [m] and Y[n] as $\ell(m,n)$, where an incomplete recursive definition for the function $\ell(i,j)$ to compute the length of the LCS of X [m] and Y[n] is given below:

$\ell(i,j)$ = 0 if either i = 0 or j = 0

= expr 1, if i, j>0 and X [i–1] = Y [j–1]

= expr 2, if i, j > 0 and X [i–1] ≠Y[j–1]

Which one of the following options is correct?

[GATE 2009]

[2-Marks]

(a) expr 1 ≡ $\ell(i-1, j)+1$

(b) expr 1 ≡ $\ell(i, j-1)$

(c) expr 2 ≡ max ($\ell(i-1, j)$, $\ell(i, j-1)$)

(d) expr 2 ≡ max ($\ell(i-1, j-1)$, $\ell(i, j)$)

Q.44 The values of $\ell(i,j)$ could be obtained by dynamic programming based on the correct recursive definition of $\ell(i,j)$ of the form given above, using an arrayL [M,N], where M = m+1 and N = n+1, such that L [i,j] = $\ell(i,j)$.

Which one of the following statements would be TRUE regarding the dynamic programming solution for the recursive definition of $\ell(i,j)$.

[GATE 2009]

[2-Marks]

(a) All elements of L should be initialized to 0 for the values of $\ell(i,j)$ to be properly computed

(b) the values of $\ell(i,j)$ may be computed in a row major order or column major order of L [M,N]

(c) The values of $\ell(i,j)$ cannot be computed in either row major order or column major order of L [M,N]

(d) L [p,q] needs to be computed before L[r,s] if either p<r or q<s

Linked Questions 45 & 46:

Q.45 Consider a binary max-heap implemented using an array. Which one of the following array represents a binary max-heap? **[GATE 2009]**

[2-Marks]

(a) 25, 12, 16, 13, 10, 8, 14

(b) 25, 14, 13, 16, 10, 8, 12

(c) 25, 14, 16, 13, 10, 8, 12

(d) 25, 14, 12, 13, 10, 8, 16

Q.46 What is the content of the array after two delete operations on the correct answer to the previous question ?

[GATE 2009]

[2-Marks]

(a) { 14, 13, 12, 10, 8 }

(b) { 14, 12, 13, 8, 10 }

(c) { 14, 13, 8, 12, 10 }

(d) { 14, 13, 12, 8, 10 }

ANSWER KEY

1	a	2	a	3	c	4	a	5	a
6	d	7	a	8	d	9	c	10	c
11	a	12	c	13	a	14	b	15	c
16	b	17	a	18	c	19	d	20	c
21	c	22	d	23	a	24	b	25	b
26	b	27	d	28	d	29	d	30	a
31	c	32	a	33	c	34	c	35	c
36	d	37	c	38	b	39	c	40	b
41	b	42	d	43	c	44	b	45	c
46	d								

SOLUTIONS

S.1 (a)

Worst case time complexity to delete node x from the list.

= (time to traverse x–1 nodes) + delete (x)

= (n – 1) + 1

= n

∴ **time complexity = O (n).**

S.2 (a)

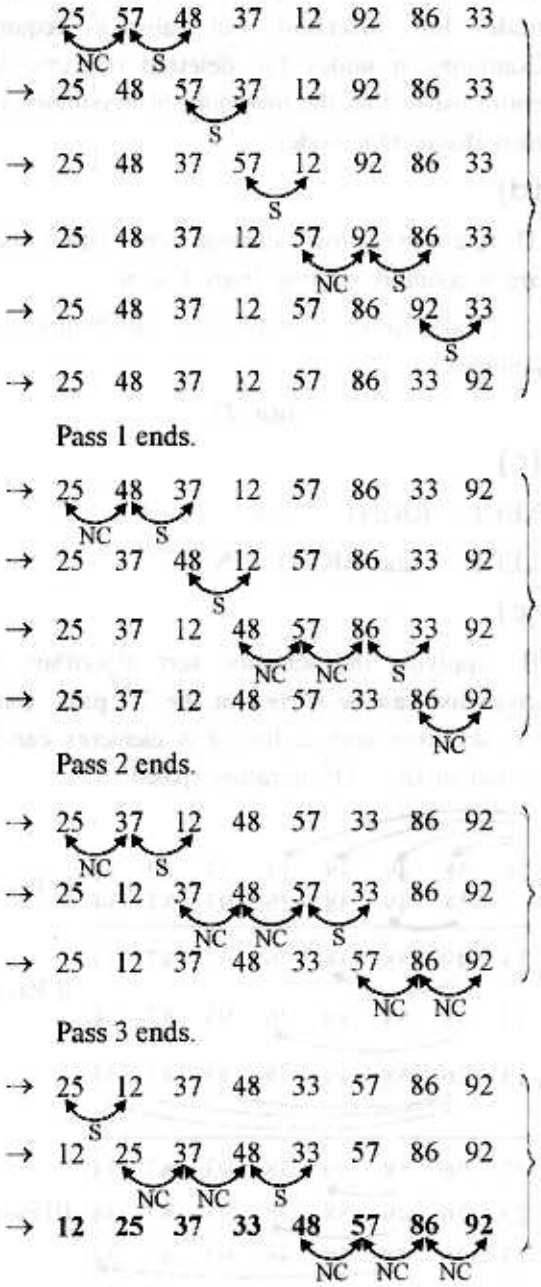
A queue is maintained by a circular array with N memory cells. If an element is added to it then RIGHT is increased by 1(mod N) whereas if an element is deleted then RIGHT is decreased by 1 mod (N).

S.3 (c)

Both the operations can be done in **O(1) time.** Which is a constant.

S.4 (a)

By Bubble sort method,

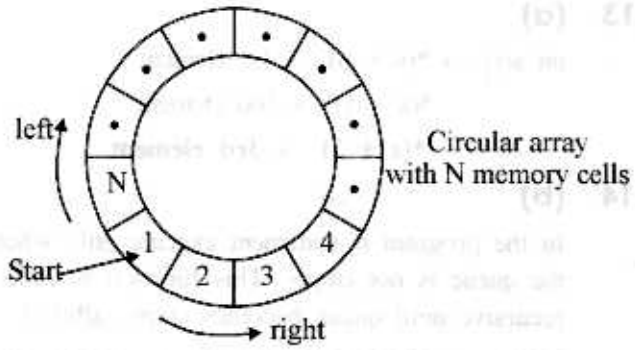


S.5 (a)

A prototype of a priority queue is **timesharing system.** Program of high priority are processed first and programs with the same priority form a standard queue.

S.6 (d)

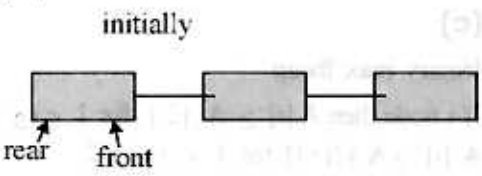
Circular dequeue



When an element is added to the dequeue, memory cell get increased by 1 (mod N) to the right.

When an element is deleted from the dequeue, memory cell get decreased by 1 (mod N) to the left.

S.8 (d)



S.9 (c)

Numbers → 100, 99, ..., 3, 2, 1

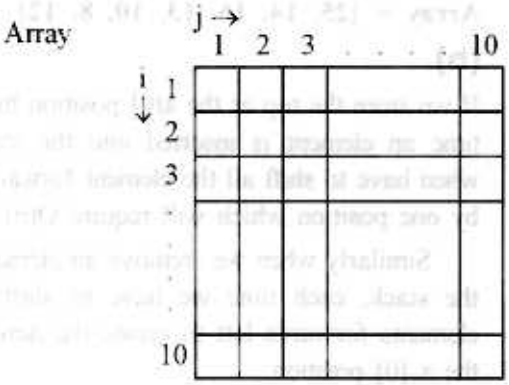
Search → 101

Because 101 is not present in the list, so we have to compare with all numbers in the list.

So, total comparisons will be **100.**

S.11 (a)

Row Major Order



If location of first array element starts from zero then address of A[i] [j] is

$$\text{Add}(A[i][j]) = 0 + 15(i - 1) + j - 1$$

If array start from location 100 then

Add (A[i] [j]) = 100 + 15(i – 1) + j – 1

= **15i + j + 84**

S.13 (a)

int s[s] → *(s + 0) → 1st element

*(s + 1) → 2nd element

***(s + 2) → 3rd element**

S.14 (b)

In the program if statement execute only when the queue is not empty. This function is called recursive until queue becomes empty after it.

Flow of control is transferred to next statement and insert the element in reverse order of element in the queue.

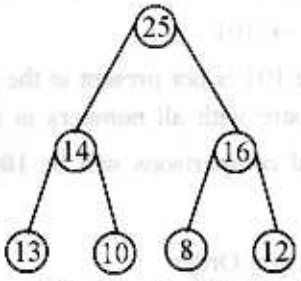
The condition if (! is empty (Q)) is true when the Q is not empty again i = delete (Q) delete the element from the front again and insert (Q, i) inserts the element at the rear of the queue. Hence **the reverse of the queue is obtained.**

S.15 (c)

Binary max lheap:

If i node then A [i] ≥ A [2i] for $1 \leq i \leq n/2$ and A [i] ≥ A [2i+1] for $1 \leq i \leq n/2$

where 2i is the left child and 2i +1 is right child of node i of max heap.



Array = {25, 14, 16, 13, 10, 8, 12}

S.16 (b)

If we store the top at the a[0] position then each time an element is inserted into the stack, we when have to shift all the element forwards right by one position which will require O(n) time.

Similarly when we remove an element from the stack, each time we have to shift all the elements forwards left to create the new top on the a [0] position.

This operation also takes the O(n) time.

Hence **push and pop will take the linear time.**

S.18 (c)

An unordered list may also be used as a priority queue. Such a list requires examining only **one node for insertion** but always requires examining **n nodes for deletion** (traverse the entire list to find the minimum or maximum and then delete that node).

S.19 (d)

There are (n+2) rows starting from –1 to n, there are n columns starting from 1 to n.

∴ Total number = number of rows*number of columns

= **n(n+2)**

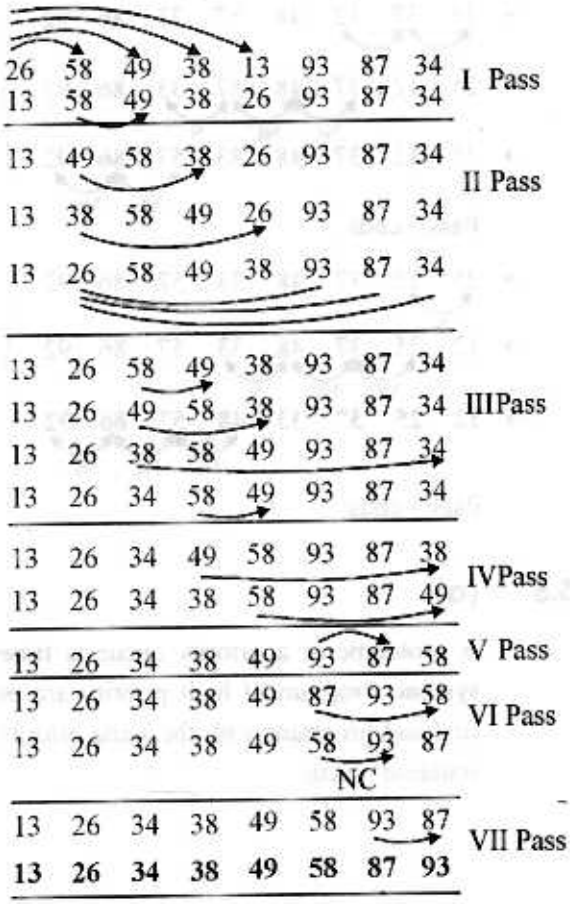
S.20 (c)

LEFT = RIGHT + 1 or

LEFT =1 and RIGHT = N.

S.21 (c)

By applying the selection sort algorithm, the given **list can be sorted in the 7th pass.** Using the selection sort, a list of n elements can be sorted in $(n - 1)^{th}$ iteration (pass).



S.22 (d)

Wheel sort also known as quick sort.

First pass:

d =4

11, 22, 44, 66, 55, 99, 88, 33

Second pass:

d = 3

11, 22, 33, 44, 55, 99, 88, 66

Third pass:

d = 2

11, 22, 33, 44, 55, 99, 88, 66

Fourth pass:

d = –1

11, 22, 33, 44, 55, 66, 88, 99

S.23 (a)

Worst case would be when all negative numbers will be after all positive number. Thus

(i) If number of positive numbers is equal to number of negative numbers. Then Number of exchanges required = n/2.

(ii) If number of positive numbers is not equal to number of negative numbers. Then number of exchange in worst case

Let one e.g., 2, –1, –2, –5, –6

For arranging in required position we need 4 exchanges per 5 numbers.

–1, –2, –5, –6, 2

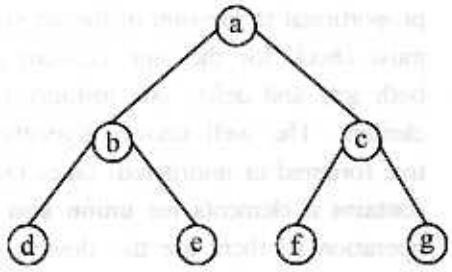
So, for n, **(n–1) exchange required.**

S.24 (b)

Let array be sorted in ascending order, if sum of first two elements is less than 1000 then number of operations are fixed and not dependent on n, i.e., complexity is O(1). But, if elements are not present as first two elements, then their complexity will be:

$$n \le t(n) \le n \log_2 n$$

S.25 (b)



The above tree can be represented in list form as

a	b	c	d	e	f	g
1	2	3	4	5	6	7

Let i = 5

∴ element [i] = e

From the heap tree, one can see that parent of e is b whose index is 2.

Analysing each of the given options for i = 5

(i) i–1 = 5 – 1=4

∴ This is false

(ii) $\lfloor i/2 \rfloor = \lfloor 5/2 \rfloor = 2$**(flooring)**

∴This is true.

(iii) $\lceil i/2 \rceil = \lceil 5/2 \rceil = 3$ (ceiling)

∴This is false.

(iv) (i+1)/2 = (5+1)/2 = 3

∴ This is false.

S.26 (b)

Stack is the data structure in which push and pop operations takes constant time. Hence it is the best data structure to check whether an arithmetic expression has a balanced parenthesis or not.

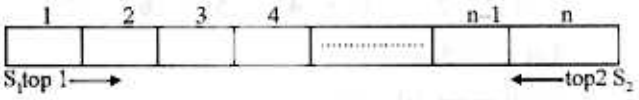
S.27 (d)

If each set is represented as a linked list with elements in arbitrary order. Membership ,cardinality takes O(1) time for an element in the set so the operation is very fast in the case of intersection, Let A and B are two sets. An operation such as A ∪ B requires time at least proportional to the sum of the sizes of the two sets. Since the list representing A and the list

representing B must be scanned at least once. Similarly operation A ∩ B requires time at least proportional to the sum of the set sizes, since we must check for the sum element appearing in both sets and delete one instance of each such element. The well known algorithm using 2-3 tree (ordered or unordered) takes O(nlogn) if set contains n elements for **union and intersection** operation so there are the slowest operation.

S.28 (d)

The implementation of 2 stacks in a single array A is called colliding strategy.



top 1 < top 2

S_1 grows from initial location of A and S_2 from final position of A growing in opposite direction. Here decrement of 1 in each step of top 2 results in increment of 1 in each step of top 1.

Hence condition is top 2–1–top 1 = 0

top 1 = top 2 – 1

S.29 (d)

The given code is

1. Counter = 0.;
2. for(i = 1; i < = n; i ++)
3. { if (A[i] = = 1) counter ++;
4. else { f(counter); counter = 0;}

The time complexity of the program fragment depends on the frequency (Number of steps) of line 3 and 4. In line 4 the frequency depends on the variable counter and there is no increment in the counter variable which is intialized to 0, so f(0) then counter = 0 means there is no cell in an array which is having a bit 0, so all cells in the array contains 1. Consider the line 3 if (A[i] = = 1) counter ++; the value of i will be increased upto n so the value of counter will be n. Since n is the frequency of the line 3 and the frequency of line 4 is 0. So the time complexity of line 3 is **O(n)** on average n and f(0) = O(1) is the time complexity of line 4. So the time complexity of the program fragement is maximum of line 3 and 4 which is **O(n)** on average.

S.30 (a)

Answer cannot be front node as we cannot get rear from front in O(1), but if **p is rear** we can implement both queue and dequeue in O(1) because from rear we can get front in O(1).

S.31 (c)

The multiplication process requires the elements of the rows of first matrix will be multiplied with the corresponding elements of the columns of the second matrix. The elements are always stored in row major order by default. But if one store the second matrix elements in column major order then they will be available with a time.

Ex. - $M_1 = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$

$M_2 = \begin{bmatrix} x & y \\ z & w \end{bmatrix}$

$M_1 \rightarrow$ A | a | b | c | d |

row major order

$M_2 \rightarrow$ B | x | z | y | w |

column major order

B | x | z | y | w |

row major order

If we store M_2 matrix in column major, we can directly multiply the respective number of array A and B.

But if we store my matrix in row major order then we have to skip the whole now to multiply next element. Thus this will be time taking.

S.32 (a)

Given,

Number of edge = 27

6 vertices with degree 2,

3 vertices with degree 4,

remaining with degree 3.

Taking options,

Let, Number of possible vertices be 10.

So, (12 + 12 + 3) = 27 edges.

Hence, number of vertices = 10

S.33 (c)

Cyclomatic Complexity of above module = 3

Here, CC becomes 1 for while construct, it becomes 2 for if..else construct.

Now, CC = Number of decisions +1 = 2 + 1 = 3

S.34 (c)

Let a[5] = {2, 4, 6, 8, 10}

b[5] = {1, 3, 5, 7, 9}

c[10] = {1,2,3,4,5,6,7,8,9,10}

(I) $a[i] \geq b[i] \Rightarrow c[2i] \geq a[i]$

put i = 2,

$a[2] \rightarrow 6,\ b[2] \rightarrow 5,\ c[4] \rightarrow 5$

So $a[2] \geq b[2] \Rightarrow c[4] \geq a[2]$ False

(II) $\mathbf{a[i] \geq b[i] \Rightarrow c[2i] \geq b[i]}$

put i = 2,

$a[2] \rightarrow 6,\ b[2] \rightarrow 5,\ c[4] \rightarrow 5$

So $[2] \geq b[2] \geq c[4] \geq b[2]$ **True**

Similarly

(III) $\mathbf{a[i] \geq b[i] \Rightarrow c[2i] \leq a[i]}$

put i = 2,

$c[4] \leq a[2]$ **True**

(IV) $a[i] \geq b[i] \Rightarrow c[2i] \leq b[i]$

put i = 2,

$c[4] \leq b[i]$ False

S.35 (c)

Let x contain n elements, divide it by n/2 repeadly so the recurrence become

$T(1) = 1$

$T(n) = 2T(n/2) + 1$

$T(n) = O(n \log n)$

$= \theta(n \log n)$

S.36 (d)

In given algorithm the for loop contains a logical expression

$z[i] = (x[i] \wedge \sim y[i]) \vee (\sim x[i] \wedge y[i])$

The equivalent set representation of given logical expression if we assume z[i] = z, x[i] = x and y[i] = y then

$z = (x \cap y') \cup (x' \cap y)$

$\mathbf{z = (x - y) \cup (y - x)}$

$[\because\ A \cap B' = A - B]$

S.37 (c)

$a_1, a_2 \ldots a_n$

$b_1, b_2 \ldots b_n$

In order to find out the largest span.

$\therefore$ Check the sums of $a_i + \ldots a_j$ and $b_i + \ldots b_j$ at each step.

If $a_1 + a_2 = b_1 + b_2$ go on to check $a_1+a_2+a_3$ and $b_1+b_2+b_3$.

If not, then check a_2+a_3 and b_2+b_3

Similarly a check is done at each of the n places during traversal. A separate variable has to be kept that contains the maximum span observed hitherto.

$\therefore$ The fastest algorithm computes with **$\theta(n)$ time and space.**

S.38 (b)

When we apply the divide and conquer method such that divide n into part of n/2 then recurrence equation is

$T(n) = 1$ for $n = 2$

$T(n) = 2T(n/2) + 2$ for $n > 2$

The solution of T(n) is

$$T(n) = \frac{3}{2}n - 2$$

$$= \mathbf{1.5\ n - 2\ comparison}$$

S.39 (c)

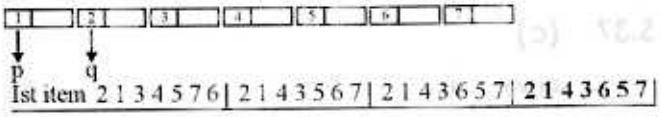
The BFS using Queue data structure is **QMNPRO**.

S.40 (b)

Consider an array of n elements which stores integers in sorted order, A[1], A[2],...,A[n]. If we want to search an element x with the help of binary search then in expected case comparison is not greater than $\lceil \log(n + 1) \rceil$. But if integer appears n/2 times in an array and array is already sorted then it take $\theta(1)$ times for consecutive locations. So total comparison is not greater then **$\theta(\log n)$** time.

S.41 (b)

Execution of code will be like this:



The function rearrange ( ) exchanges data of every node with its next node. It starts exchanging data from the first node itself.

S.42 (d)

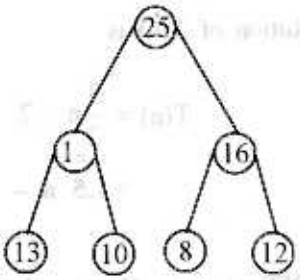
A binary heap of n elements is there and we have to insert n more elements, for inserting each element into the heap we have to perform n comparison.

Hence for inserting the n elements we, have to perform n*n comparison.

$\Rightarrow$ **Time required** = θ (n^2).

S.45 (c)

A tree is max-heap if data at every node in the tree is greater than or equal to it's children's data. In array representation of heap tree, a node at index i has its left child at index 2i + 1 and right child at index 2i + 2.



S.46 (d)

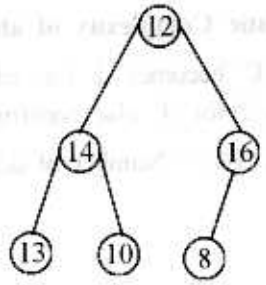
For Heap trees, deletion of a node includes following two operations:

(1) Replace the root with last element on the last level.

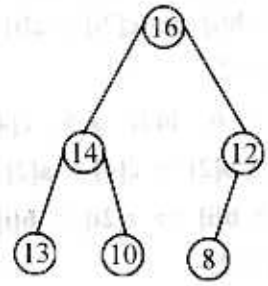
(2) Starting from root, heapify the complete tree from top to bottom.

Let us delete the two nodes one by one

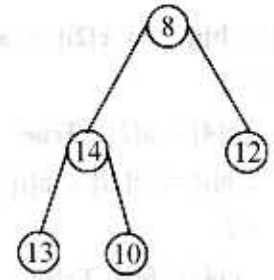
Delete 25 : Replace by 12



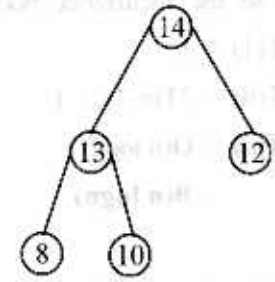
After heapify:



Delete 16: Replace it by 8



After heapify



= **{14, 13, 12, 8, 10}**

5.2

TREE AND BINARY SEARCH TREE

LEVEL–1

Q.1 Level order traversal of a rooted tree can be done by starting from the root and performing

(a) preorder traversal

(b) in-order traversal

(c) depth first search

(d) breadth first search

Q.2 In a depth-first traversal of a graph G with n vertices , k edges are marked as tree edges. The number of connected components in G is

(a) k

(b) k+1

(c) n-k-1

(d) n-k

Q.3 In a binary tree, the number of internal nodes of degree 1 is 5, and the number of internal nodes of degree 2 is 10. The number of leaf nodes in the binary tree is

(a) 10

(b) 11

(c) 12

(d) 15

Q.4 Which of the following sequences of array elements forms a heap?

(a) {23,17,14,6,13,10,1,12,7,5}

(b) {23,17,14,6,13,10,1,5,7,12}

(c) {23, 17, 14, 7, 13, 10, 1, 5, 6, 12}

(d) {23, 17, 14,7, 13, 10, 1, 12, 5, 7}

Q.5 Suppose that we have numbers between 1 and 100 in a binary search tree and want to search for the number 55. Which of the following sequences CANNOT be the sequence of nodes examined?

(a) {10,75,64,43,60,57,55}

(b) {90,12,68,34,62,45,55}

(c) {9, 85, 47, 68, 43, 57, 55}

(d) {79,14, 72, 56, 16, 53, 55}

Q.6 The number 1,2.....,n are inserted in a binary search tree in some order. In the resulting tree, the right subtree of the root contains p nodes. The first number to the inserted in the tree must be

(a) p

(b) p+1

(c) n–p

(d) n–p+1

Q.7 When searching for the key value 60 in a binary search tree, nodes containing the key values 10, 20, 40, 50, 70, 80, 90 are traversed, not necessarily in the order given. How many different orders are possible in which these key values can occur on the search path from the root to the node containing the value 60?

(a) 35

(b) 64

(c) 128

(d) 5040

Q.8 Which of the following is TRUE ?

(a) The cost of searching an AVL tree is θ(log n) but that of a binary search tree is θ(n)

(b) The cost of searching an AVL tree is θ(log n) but that of a complete binary tree is θ(n log n)

(c) The cost of searching a binary search tree is θ(log n) but that of an AVL tree is θ(n)

(d) The cost of searching an AVL tree is θ(n log n) but that of a binary search tree is θ(n)

Q.9 The order of an internal node in a B+ tree index is the maximum number of children it can have. Suppose that a child pointer takes 6 bytes, the search filed value takes 14 bytes, and the block size is 512 bytes. What is the order of the internal node.

(a) 24

(b) 25

(c) 26

(d) 24

Q.10 Select the true statement.

(a) Every binary tree is either complete or full.

(b) Every complete binary tree is also a full binary tree.

(c) Every full binary tree is also a complete binary tree

(d) No binary tree is both complete and full.

Q.11 Which data structure has the fastest insertion procedure?

(a) Binary search tree

(b) Ordered array

(c) Heap

(d) Ordered linked list

Q.12 Suppose we convert a binary tree T into an extended binary tree, then nodes in the original tree T become nodes in the extended tree.

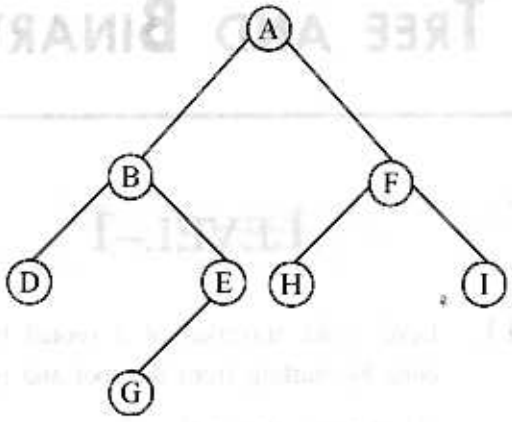
(a) Dead

(b) Internal

(c) External

(d) Leaf

Q.13 Which option is true?



(i) strictly binary tree

(ii) not a strictly binary tree

(iii) almost complete binary tree

(iv) complete binary tree

(a) (i) and (iii)

(b) Only (iii)

(c) Only (iv)

(d) (ii) and (iii)

Q.14 Let T be a binary tree, then which of the following statements is correct?

(i) A terminal node is called a leaf and a path ending in a leaf is called a branch

(ii) The nodes with same level and depth are said to belong to the same generation

(iii) The height of a tree T is the maximum number of nodes in a branch of T and it is one more than the largest level number of T.

(a) (i), (iii)

(b) (ii), (iii)

(c) (i), (ii)

(d) (i), (ii), (iii)

Q.15 Consider the complete binary tree T_{26}, with all the nodes in order, sequentially from left to right, generation by generation. For such a complete binary tree, the parent node of nodes 24 and 25 would be

(a) 12 13
(b) 10 11
(c) 12 12
(d) 11 11

Q.16 Which out of following works on "Divide and conquer" policy?

(a) Selection sort
(b) Heap sort
(c) Quick sort
(d) Shell sort

Linked Questions 17 & 18:

Q.17 What is the maximum number of comparisons required to search an element in an unsorted list of n element using binary search :

(a) $\log_2 n$
(b) $\log_2 n + 1$
(c) $\log_2 n - 1$
(d) Cannot be determined

Q.18 There are two sorted lists each of length n. An element is to be searched in both the lists. The lists are mutually exclusive. The maximum number of comparisons required using binary search is:

(a) $\log_2 2n$
(b) $\log_2 n^2$
(c) $1 + \log_2 n$
(d) cannot be determined

Q.19 Binary search requires average no of comparisons

(a) $\log_2 n$
(b) $\log_2(\log_2 n)$
(c) n^2
(d) $\log_2 (n \log^2 n)$

Q.20 The maximum height of a balanced binary search tree with n nodes is

(a) $1.44 \log_2 n$
(b) $\log_2 n$
(c) $1.44 \log_2 (2n)$
(d) $\log_2 (2n)$

Q.21 In a top-down multiways tree

(i) Multiways search trees
(ii) a semileaf must be a leaf
(iii) a semileaf must be a non leaf

(a) (i ,ii)
(b) (i,iii)
(c) (i)
(d) (ii)

Q.22 Interpolation search is better than binary search if:-

(a) The keys are uniformly distributed between k(0) and k (n–1) in the table.
(b) The keys are non uniformly distributed between k(0) and k (n–1)
(c) The keys are uniformly distributed between k(0) and k (n+1) in the table
(d) The keys are non-uniformly distributed between k (0) and k (n+1) in the table.

Q.23 Match the following:

Group I		Group II	
(a)	Only arithmetic on integer indexes and division by 2	(i)	interpolation search
(b)	Arithmetic on key 3 and complex multiplications and divisions	(ii)	binary search
		(iii)	transposition search

(a) a-ii, b-iii
(b) a-iii, b-ii
(c) a-ii, b-i
(d) a-iii, b-i

Q.24 How many operations are required in Round Robin algorithm used in minimum spanning tree, if an appropriate implementation of the priority queue is used.

(a) O(e log n)
(b) O(n log n)
(c) O(n log e)
(d) O(e log log n)

Q.25 A binary search tree enables one to search for and find an element with an average running time $f(n) = O(\log_2 n)$. This structure is compared with a sorted linear array and linked list. Which of the following statements are correct ?

(i) In a sorted linear array one can search for and find an element with a running time $f(n) = O(n)$ but it is expansive to insert and delete elements.

(ii) In a linked list one can easily insert and delete elements, but it is expensive to find an element since one must use a linear search with running time $f(n) = O(\log_2 n)$.

(a) (i)
(b) (ii)
(c) Both (i) and (ii)
(d) Neither (i) nor (ii)

Q.26 If the data is not in proper order and we apply a binary tree sort (with depth d and level t of tree), then the total number of comparisons to do the same is in between_____

(a) $d+\sum_{t=1}^{d-1} 2^t*(\ell+1)$ and $\sum_{t=1}^{d} 2^t*(\ell-1)$

(b) $\sum_{t=1}^{d-1} 2^t*(\ell+1)$ and $\sum_{t=1}^{d} 2^t*(\ell-1)$

(c) $d+\sum_{t=1}^{d-1} 2^t*(\ell+1)$ and $\sum_{t=1}^{d} 2^t*(\ell+1)$

(d) $\sum_{t=1}^{d-1} 2^t*(\ell+1)$ and $\sum_{t=1}^{d} 2^t*(\ell-1)$

Q.27 The average running time to search an item in the binary search tree is proportional to tree T become _____ nodes in the extended tree.

(a) n^2
(b) $\log_2 n$
(c) $\log_2 n^2$
(d) n

Q.28 A complete binary tree with 'n' non leaf nodes contains how many nodes.

(a) $\log_2 n$
(b) 2n
(c) n+1
(d) 2n+1

Q.29 An extended binary tree is equivalent to

(i) 2 tree
(ii) each node with 0 or 2 child node

(a) only ii
(b) only i
(c) both i and ii
(d) neither i nor ii

Q.30 What is the maximum total number of nodes in a tree that has N levels? Note that the root is level (zero)

(a) 2^N-2N
(b) $2^{N+1}-1$
(c) $2^{2N}-1$
(d) 2^N+1

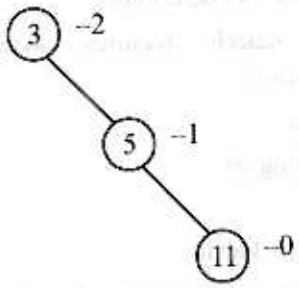
Q.31 A tree with N node has

(a) N edges
(b) N^2-1 edges
(c) N–1 edges
(d) N–2 edges

Q.32 The height of a full binary tree with N internal nodes is about

(a) $\lceil \log_2 N \rceil$
(b) N log N
(c) $\log_2 N$
(d) 2N

Q.33 In the following AVL tree, structure needs to be balanced so we have to rotate it



(a) clockwise
(b) counter clockwise
(c) in both direction
(d) none

Q.34 A binary tree in which if all its levels except possibly the last, have the maximum number of nodes and all the nodes at the last level appear as far left as possible, is called.

(a) Full binary tree

(b) 3-tree

(c) traded tree

(d) complete binary tree

Q.35 In a complete binary tree of n nodes, how far are the most distant two nodes? Assume that each edge in the path counts as 1. Assume log (n) as log base 2.

(a) About log (n)

(b) About 2 log (n)

(c) About 3 log (n)

(d) About 4 log (n)

Q.36 A binary tree having n nodes and depth d will be about complete binary tree if

(a) any node at level less than d-1 has two sons

(b) it contains $2^{d+1}-1$ nodes

(c) for any node nd in the tree with a right descendent at level d, nd must have a left son

(d) all of these

Q.37 Number of possible binary trees with 3 nodes is

(a) 13

(b) 14

(c) 16

(d) 15

Q.38 A binary search tree is generated by inserting the following integers :

50,15,62,5,20,58,91,3,8,37,60,24

The number of nodes in the left subtree and right subtree of the root are respectively :

(a) (4,7)

(b) (7,4)

(c) (8,3)

(d) (3,8)

Q.39 If only the root node does not satisfy the heap property, the algorithm to convert the complete binary tree into a heap has the best asymptotic time complexity of

(a) O(n)

(b) O(log n)

(c) O(n log n)

(d) O (n log log n)

Q.40 What is the largest integer m such that every simple connected graph with n vertices and n edges contains at least m different spanning trees ?

(a) 1

(b) 2

(c) 3

(d) n

Q.41 Which of the following statements is/are true?

(i) Spanning tree will always have |V|- 1 edge

(ii) Adding an edge to the spanning tree always creates a cycle

(iii) The intermediate graph obtained from both Prim's and Kruskal's algorithms in finding minimum cost spanning tree are always trees.

(a) Both (i) and (ii)

(b) Only (i)

(c) Only (ii)

(d) All of the above

LEVEL-2

Q.42 A binary search tree contains the numbers 1, 2, 3, 4, 5, 6, 7, 8. When the tree is traversed in pre-order and the values in each node are printed out, the sequence of values obtained is 5, 3,1, 2, 4, 6, 8, 7. If the tree is traversed in post-order, the sequence obtained would be

(a) 8,7,6,5,4,3,2,1

(b) 1,2,3,4,8,7,6,5

(c) 2,1,4,3,6,7,8,5

(d) 2,1,4,3,7,8,6,5

Common Data For Questions 43 & 44:

A Binary Search Tree (BST) stores values in the range 37 to 573. Consider the following sequences of keys

I. 81, 537, 102, 439, 285, 376, 305

II. 52, 97, 121, 195, 242, 381, 472

III. 142, 248, 520, 386, 345, 270, 307

IV. 550, 149, 507, 395, 463, 402, 270

Q.43 Which of the following statements is TRUE ?

(a) I, II and IV are inorder sequences of three different BSTs

(b) I is a preorder sequence of some BST with 439 as the root

(c) II is an inorder sequence of some BST where 121 is the root and 52 is a leaf

(d) IV is a postorder sequence of some BST with 149 as the root

Q.44 How many distinct BSTs can be constructed with 3 distinct keys ?

(a) 4

(b) 5

(c) 6

(d) 9

Q.45 The following key values are inserted into a B+-tree in which order of the internal nodes is 3, and that of the leaf nodes is 2, in the sequence given below. The order of internal nodes is the maximum number of tree pointers in each node, and the order of leaf nodes is the maximum number of data items that can be stored in it. The B+ -tree is initially empty. 10,3,6,8,4,2,1

The maximum number of times leaf nodes would get split up as a result of these insertions is

(a) 2

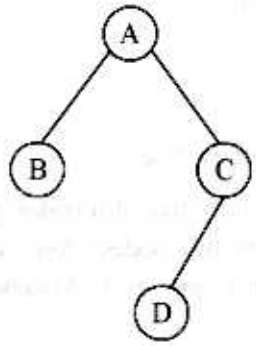
(b) 3

(c) 4

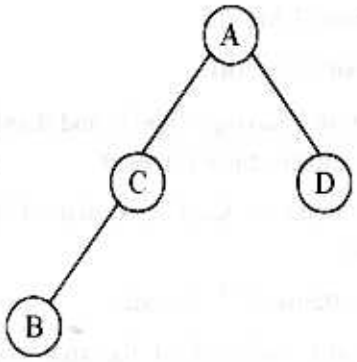
(d) 5

Q.46 Which one of the following binary trees has its inorder and preorder traversals as BCAD and ABCD, respectively ?

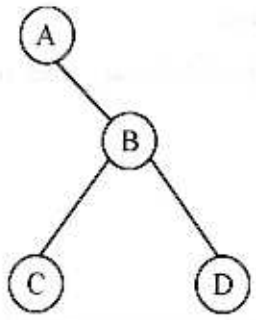
(a)



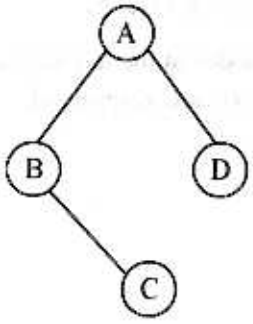
(b)



(c)



(d)



Q.47 In which order the following numbers, 10, 3, 2, 4, 6, 7, 5, 1 should be inserted in an empty binary search tree to get a binary search tree of height 5?

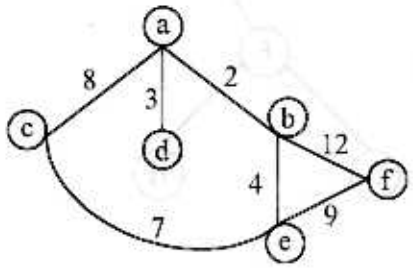
(a) 1, 2, 3, 6, 5, 7, 10, 4

(b) 1, 2, 3, 5, 10, 6, 7, 4

(c) 1, 2, 3, 4, 6, 7, 10, 5

(d) 1, 2, 3, 10, 4, 7, 6, 5

Q.48 If an edge with weight 5 is to be inserted in the following simple graph to get 2 minimal spanning tree then where should it be inserted?



(a) df

(b) de

(c) ce

(d) Not possible to get two minimal spanning trees

Q.49 In which order the following numbers, 7, 5, 1, 8, 3 and 2, should be inserted into an empty binary search tree to get in order and preorder traversal sequences as same

(a) 3, 2, 1 8, 7, 5

(b) 8, 7, 5, 3, 2, 1

(c) 1, 2, 3, 5, 7, 8

(d) 3, 2, 8, 7, 5, 1

Q.50 If n numbers are inserted into a binary search tree then which of the following traversal outputs the numbers in sorted order?

(a) preorder

(b) in order

(c) post order

(d) breadth first

Q.51 Every binary tree whose preorder search produces the string J B A C D I H E G F must have 10 vertices. What else do the trees have in common?

(i) The root must be labeled J.

(ii) If J has a right child it must be labeled B, otherwise the left child is labeled B.

(iii) If J has a left child, it must be labeled B, otherwise the right child is labeled B.

(iv) F will always be at the lower most level.

(a) (i) (iv)

(b) (i),(ii)

(c) (i),(iii)

(d) (iii), (iv)

Q.52 A binary search tree contains the numbers 1,2,3,4,5,6,7,8. When the tree is traversed in preorder and the values in each node printed out, the sequence of values obtained is 5, 3, 1, 2, 4, 6, 8, 7. If the tree is traversed in post order, the sequence obtained would be

(a) 8, 7, 6, 5, 4, 3, 2, 1

(b) 1, 2, 3, 4, 8, 7, 6, 5

(c) 2, 1, 4, 3, 6, 7, 8, 5

(d) 1, 2, 4, 3, 8, 7, 6, 5

Q.53 Let T_n is a complete binary tree having 1050 nodes, where all the nodes are sequentially ordered from 1 to 1050 from left to right generation by generation. The children of nodes 550,

(a) 680 and 691

(b) 1099 and 1100

(c) 1001 and 1002

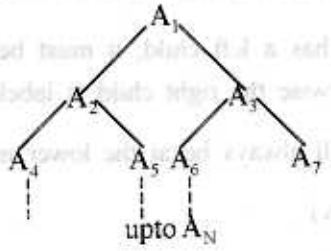
(d) 1100 and 1101

Common Data For Questions 54 & 55:

Suppose n data items $A_1, A_2, \ldots A_N$ are already sorted i.e. $A_1 < A_2 < \ldots A_N$.

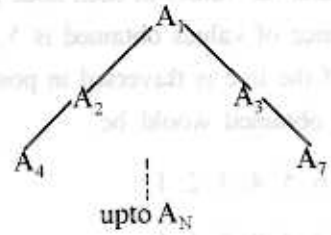
Q.54 Assuming that the items are inserted in order into an empty binary search tree, the final tree will be:

(a)



(b)

(c)



(d)

Q.55 Assuming the items are inserted in order sorted into an empty binary search tree T, then the depth of such a tree is

(a) n

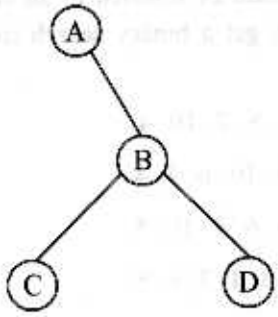
(b) n–1

(c) n+1

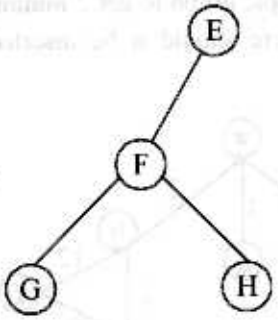
(d) $\log_2 n$

Q.56 Consider the following diagram

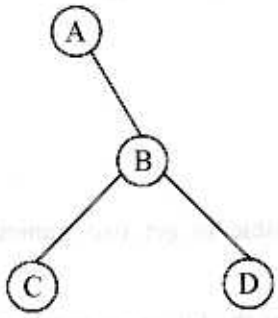
(i)



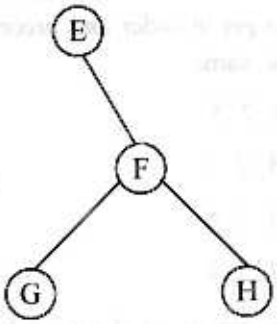
(ii)



(iii)



(iv)



Select the correct statements?

(a) (i), (iii), (iv) are similar trees.

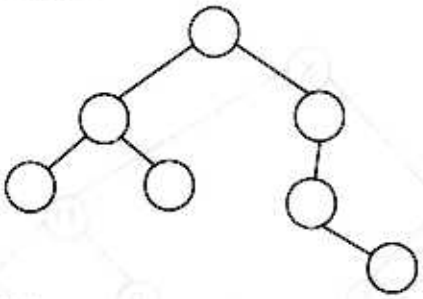
(b) (i) and (iii) are copies of each other

(c) (ii) is neither similar to nor a copy of the tree (iv)

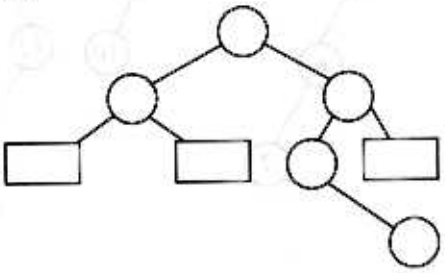
(d) all of the above

Q.57 For the following binary tree, the corresponding 2 tree is

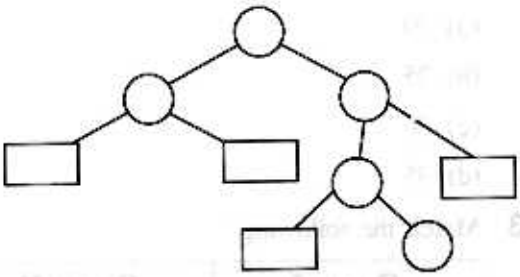
(a)



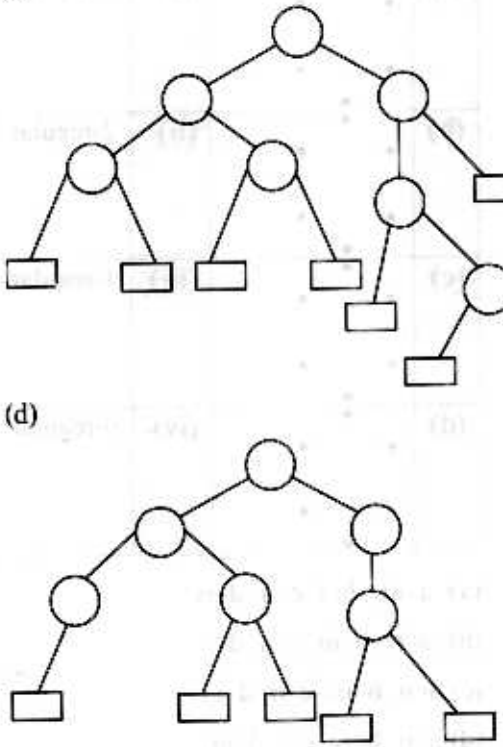
(b)



(c)



(d)



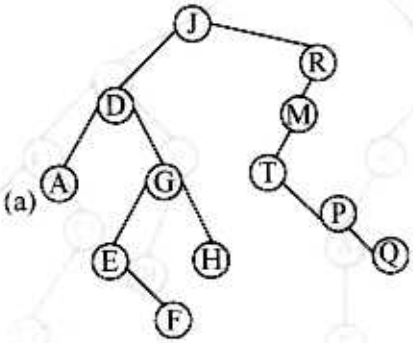
Common Data For Questions 58 & 59:

Suppose the following list of letters is inserted in order into an empty binary search tree

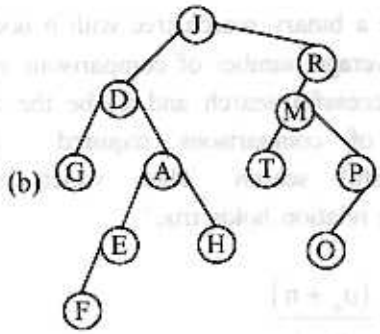
J, R, D, G T, E, M, H, P, A F, Q

Q.58 Find the final tree T.

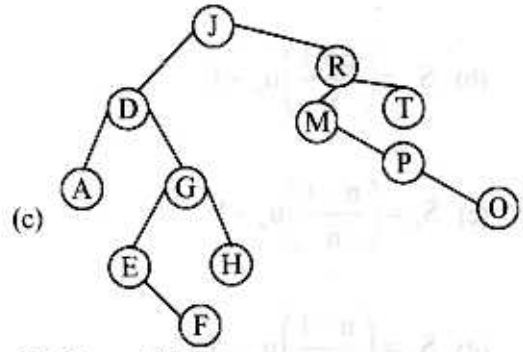
(a)



(b)



(c)



(d) None of these

Q.59 Find the average number of comparisons required for an unsuccessful search of a random binary search tree with 14 nodes and 50 as external path length.

(a) 4.2

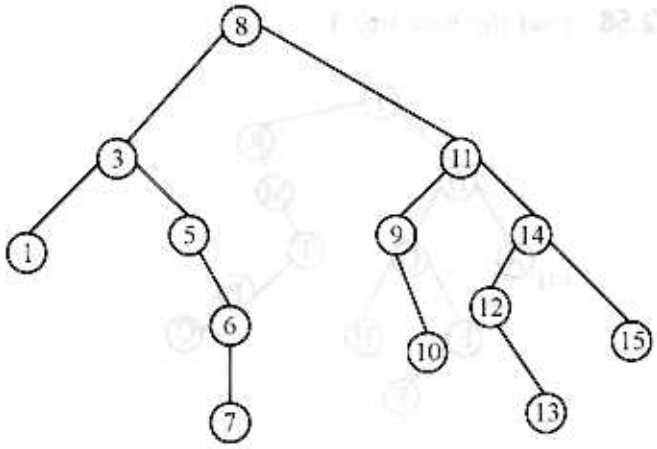
(b) 3.3

(c) 3.9

(d) 4.9

Common Data For Questions 60 & 61:

Consider the following binary tree.



Q.60 Let T be a binary search tree with n nodes and S_a the average number of comparisons required for a successful search and u_a be the average number of comparisons required for an unsuccessful search. Then which of the following relation holds true?

(a) $S_a = \frac{(u_a + n)}{n-1}$

(b) $S_a = \left(\frac{u+1}{n}\right) u_a + 1$

(c) $S_a = \left(\frac{n-1}{n}\right) u_a + 1$

(d) $S_a = \left(\frac{n-1}{n}\right) u_a - 1$

Q.61 The time required to search a binary search tree varies between _____ and _____ depending in the structure of the tree.

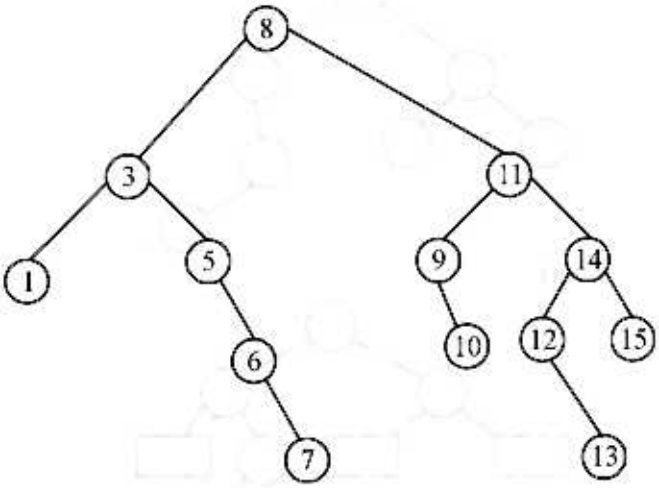
(a) $O(n)^2$ O(n)

(b) O(n), O (log n)

(c) O (n^2), O (log n)

(d) O ($\log n^2$), O (log n)

Q.62 Consider the following binary tree



The internal path length for the above tree is

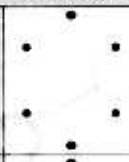
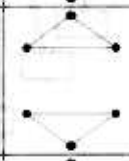
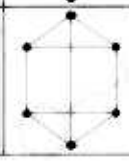
(a) 20

(b) 25

(c) 30

(d) 35

Q.63 Match the following

Group I		Group II	
(a)		(i)	3-regular
(b)		(ii)	2-regular
(c)		(iii)	1-regular
(d)		(iv)	0-regular

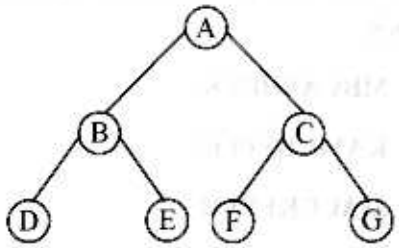
(a) a–iii, b–i, c–ii, d–iv

(b) a–iv, b–iii, c–ii, d–i

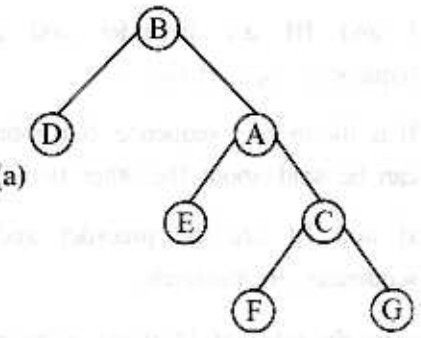
(c) a–ii, b–iii, c–ii, d–i

(d) a–ii, b–iv, c–i, d–iii

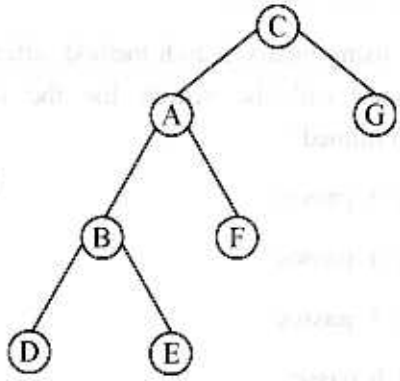
Q.64 Consider the following tree, the corresponding left rotation for the tree is



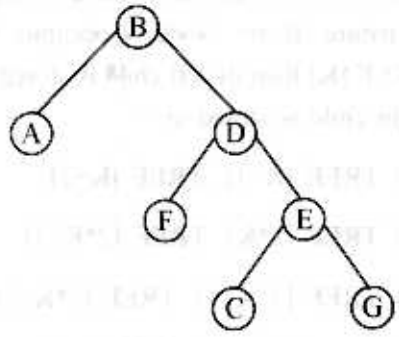
(a)



(b)

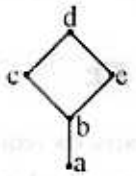


(c)

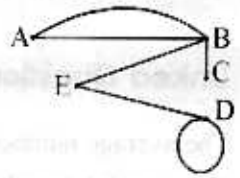


(d) None of these

Q.65 Consider the graphs below and choose the correct option:



(i)



(ii)

(a) Hamiltonian path and circuit is possible for (i) and (ii)

(b) Hamiltonian path and circuit is possible for (i) and not for (ii)

(c) Hamiltonian path is possible for (i) and not for (ii) but circuit is not possible for both (i) and (ii)

(d) Hamiltonian path is not possible for (i) but possible for (ii) and circuit is possible for (i), but not possible for (ii)

Q.66 Let E denote the following algebraic expression

$$\left[a+(b-c)\right]*\left[\frac{(d-e)}{(f+g-h)}\right]$$

If we form a binary tree from E and apply a postorder traversal, then we get

(a) * + a - bc/- de - + f g h

(b) *a + - b c -/de + - g f h

(c) a b c - + d e - f g + h - / *

(d) a b c + - ed - f g h + -/*

Q.67 If a heap has 7 nodes, the depth of the tree would be

(a) 5

(b) 2

(c) 3

(d) 6

Q.68 Let T be a binary search tree with 14 node and 60 as the external path length. Then the internal path length will be

(a) 28

(b) 50

(c) 32

(d) 11

LEVEL–3

Linked Questions 69 & 70:

Q.69 The average number of comparisons required for a successful search in a binary search tree with n nodes

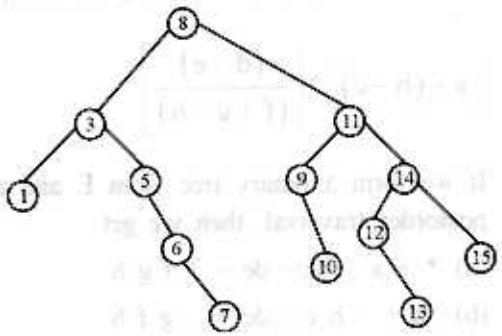
(a) i/n

(b) (i+n)/n

(c) (i+2n)

(d) $(i+n)^2/n$

Q.70 With reference to question 67, the average number of comparisons for the graph shown below will be:



(a) 4.31

(b) 3.31

(c) 2.41

(d) 2.71

Q.71 How many edges are possible in a forest with n vertices and K trees ?

(a) n – K + 1

(b) n – K – 1

(c) n–K

(d) Insufficient data

Q.72 The following three representation are known to be the preorder, inorder and postorder sequences of a binary tree. But it is not known which is which.

I. MBCAFHPYK

II. KAMCBYPFH

III. MACCKFYPH

(a) I and II are preorder and inorder sequences, respectively

(b) I and III are preorder and postorder sequences, respectively

(c) II is the inorder sequence, but nothing more can be said about the other two sequences

(d) II and III are the preorder and inorder sequences, respectively

Q.73 Consider the table of 15 items as given below:

AL, EX, FN, FU, IF, IW, LE, LO, NI, OP, OR, RD, RN, TE, TI

By using binary search method, after how many passes will the search for the item IF be determined?

(a) 3 passes

(b) 4 passes

(c) 5 passes

(d) 6 passes

Q.74 In a sequential representation of a binary tree in memory, let TREE be an array which is linear in nature. If any Node N occupies the position TREE [K] then its left child is stored in and it is right child is stored in –

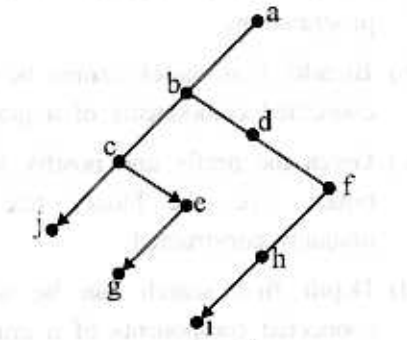
(a) TREE [K+1], TREE [K+2]

(b) TREE [2*K], TREE [2*K+1]

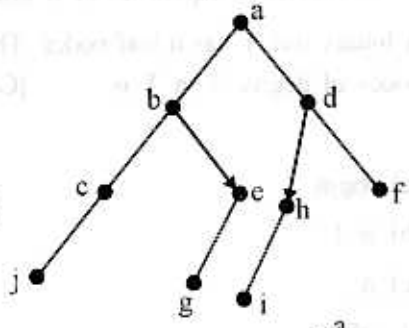
(c) TREE [2*K+1], TREE [2*K+2]

(d) TREE [K+2], TREE [K+1]

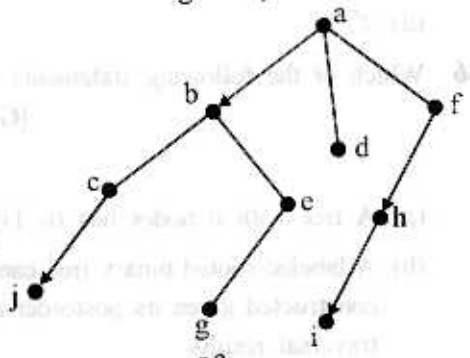
Q.75 Consider the given diagraph of the labeled binary positional tree. If this tree is the binary form B(T) of some tree T, then the digraph of the labeled tree T would be



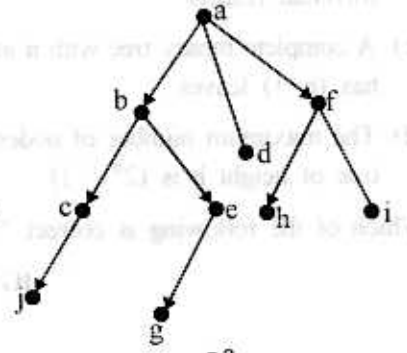
(a)



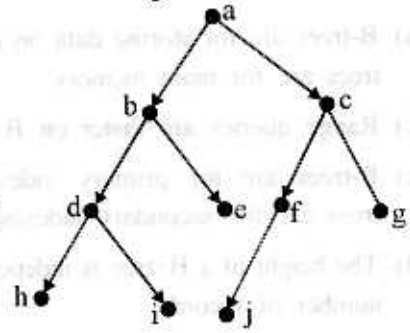
(b)



(c)



(d)



Common Data For Questions 76 & 77:

A binary tree with $n > 1$ nodes has n_1, n_2 and n_3 nodes of degree one, two and three respectively. The degree of a node is defined as the number of its neighbours.

Q.76 n_3 can be expressed as

(a) n_1-3

(b) n_1-2

(c) $\left(\frac{n_1+n_2}{2}\right)$

(d) n_2-1

Q.77 Starting with the above tree, while there remains a node v of degree two in the tree, and an edge between the two neighbours of v and then remove v from the tree. How many edges will remain all of the end of the process ?

(a) $2*(n_1-1)$

(b) n_2+2*n_1-2

(c) n_3-n_2

(d) n_2+n_1-2

Q.78 Let T be binary search tree with 15 nodes. If 4.31 is the average number of comparisons required for a successful search in a random binary search tree then what will be the internal path length?

(a) 30

(b) 50

(c) 40

(d) 60

Q.79 Let T be a binary search tree with n nodes and 56 as its external path length and the average number of comparisons required for an unsuccessful search is 4.0. Then find the value of n?

(a) 22

(b) 29

(c) 13

(d) 15

GATE QUESTIONS

Q.80 The number of rooted binary trees with n nodes is **[GATE 1990]**

(a) equal to the number of ways of multiplying (n+1) matrices.

(b) equal to the number of ways of arranging n out of 2n distinct elements

(c) equal to $\frac{1}{(n+1)}=\left(\frac{2n}{n}\right)$

(d) equal to n!

Q.81 The total external path length, EPL, of a binary tree with n external nodes is, $EPL = \sum_{W} I_W$, where I_W is the path length of external node w. **[GATE 1990]**

(a) $\leq n^2$ always

(b) $\geq n \log_2 n$ always

(c) equal to n^2 always

(d) O(n) for some special trees

Q.82 Kruskal's algorithm for finding a minimum spanning tree of a weighted graph G with 'n' vertices and m edges has the time complexity of **[GATE 1990]**

(a) $O(n^2)$

(b) O(m n)

(c) O(m+n)

(d) O(m log n)

(e) $O(m^2)$

Q.83 Following algorithm (s) can be used to sort n integers in the range [1.....n^3] in O (n) time **[GATE 1992]**

(a) Heapsort

(b) Quicksort

(c) Mergesort

(d) Radixsort

Q.84 Which one of the following statements is false? **[GATE 1994]**

(a) Optimal binary search tree construction can be performed efficiently using dynamic programming

(b) Breadth first search cannot be used to find connected components of a graph

(c) Given the prefix and postfix forms over a binary tree, the binary tree cannot be uniquely constructed.

(d) Depth first search can be used to find connected components of a graph.

Q.85 A binary tree T has n leaf nodes. The number of nodes of degree 2 in T is **[GATE 1995]** **[1-Mark]**

(a) $\log_2 n$

(b) n–1

(c) n

(d) 2^n

Q.86 Which of the following statements is false? **[GATE 1998]** **[1-Mark]**

(a) A tree with n nodes has (n–1) edges

(b) A labeled rooted binary tree can be uniquely constructed given its postorder and preorder traversal results.

(c) A complete binary tree with n internal nodes has (n+1) leaves.

(d) The maximum number of nodes is a binary tree of height h is $(2^{h+1}-1)$

Q.87 Which of the following is correct ? **[GATE 1999]** **[1-Mark]**

(a) B-trees are for storing data on disk and B+ trees are for main memory.

(b) Range queries are faster on B+trees

(c) B-trees are for primary indexes and B+ trees are for secondary indexes.

(d) The height of a B+tree is independent of the number of records.

Q.88 The most appropriate matching for the following pairs is

Group I		Group II	
X.	depth first search	1.	heap
Y.	breadth first search	2.	queue
Z.	sorting	3.	stack

[GATE 2000]

[1-Mark]

(a) X–1,Y–2,Z–3

(b) X–3,Y–1,Z–2

(c) X–3,Y–2,Z–1

(d) X–2,Y–3,Z–1

Q.89 Consider the following nested representation of binary trees: (X Y Z) indicates Y and Z are the left and right subtrees, respectively, of node X. Note that Y and Z may be NULL, or further nested. Which of the following represents a valid binary tree?

[GATE 2000]

[1-Mark]

(a) (1 2(4 5 6 7)

(b) (1((2 3 4)5 6)7)

(c) (1(2 3 4)(5 6 7))

(d) (1(2 3 NULL) 4 5))

Q.90 Let LASTPOST, LASTIN and LASTPRE denote the last vertex visited in a postorder, inorder and preorder traversal, respectively, of a complete binary tree. Which of the following is always true? **[GATE 2000]**

[2-Marks]

(a) LASTIN = LASTPOST

(b) LASTIN= LASTPRE

(c) LASTPRE = LASTPOST

(d) None of the above

Q.91 The following numbers are inserted into an empty binary search tree in the given order: 10,1, 3, 5, 15, 12, 16. What is the height of the binary search tree (the height is the maximum distance of a leaf node from the root)?

[GATE 2004]

[2-Marks]

(a) 2

(b) 3

(c) 4

(d) 6

Q.92 What is the maximum number of edges in an acyclic undirected graph with n vertices?

[IT-GATE 2004]

[1 Mark]

(a) n – 1

(b) n

(c) n + 1

(d) 2n – 1

Q.93 An array of integers of size n can be converted into a heap by adjusting the heaps rooted at each internal mode of the complete binary tree starting at the node $\lfloor (n-1)/2 \rfloor$, and doing this adjustment up to the root node (root node is at index 0) in their order $\lfloor (n-1)/2 \rfloor$, $\lfloor (n-3)/2 \rfloor$, ..., 0. The time required to construct a heap in this manner is

[IT-GATE 2004]

[2-Marks]

(a) O(log n)

(b) O(n)

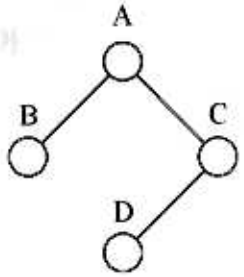
(c) O(n log log n)

(d) O(n log n)

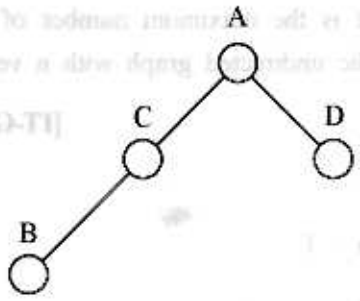
Q.94 Which one of the following binary trees has its in-order and pre-order traversals as BCAD and ABCD, respectively? **[IT-GATE 2004]**

[2-Marks]

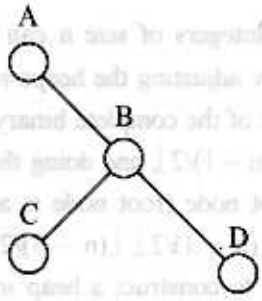
(a)



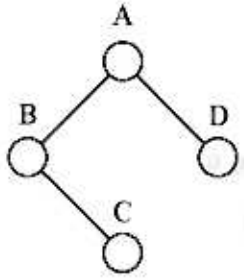
(b)



(c)



(d)



Q.95 The numbers 1, 2, ...n are inserted in a binary search tree in some order. In the resulting tree, the right subtree of the root contains p nodes. The first number to be inserted in the tree must be

[IT-GATE 2005]

[1 Mark]

(a) p

(b) p + 1

(c) n – p

(d) n – p + 1

Q.96 In a depth first traversal of a graph G with n vertices, k edges are marked as tree edges. The number of connected components in G is

[IT-GATE 2005]

[1 Mark]

(a) k

(b) k + 1

(c) n – k – 1

(d) n – k

Q.97 In the following table, the left column contains the names of standard graph algorithm, and the right column contains the time complexities of the algorithms. Match each algorithm with its time complexity.

(1) Bellman Ford algorithm	(A) O (mlog n)
(2) Kruskal's algorithm	(B) O (n^3)
(3) Floyd-Warshall algorithm	(C) O (nm)
(4) Topological Sorting	(D) O (n + m)

[IT-GATE 2005]

[1 Mark]

(a) 1-C 2-A 3-B 4-D

(b) 1-B 2-D 3-C 4-A

(c) 1-C 2-D 3-A 4-B

(d) 1-B 2-A 3-C 4-D

Q.98 A B-tree used as an index for a large database table has four levels including the root node. If a new key is inserted in this index, then the maximum number of nodes that could be newly created in the process are **[IT-GATE 2005]**

[1 Mark]

(a) 5

(b) 4

(c) 3

(d) 2

Q.99 In a binary tree, for every node the difference between the number of nodes in the left and right subtrees is at most 2. If the height on the tree is $h > 0$, then the minimum number of nodes in the tree is: **[IT-GATE 2005]**

[2-Marks]

(a) 2^{h-1}

(b) $2^{h-1} + 1$

(c) $2^h - 1$

(d) 2^h

Q.100 Let G be a weighted undirected graph and e be an edge maximum weight in G. Suppose there is a minimum weight spanning tree in G containing the edge e. Which of the following statements is always TRUE? **[IT-GATE 2005]**

[2-Marks]

(a) There exists a cutest in G having all edges of maximum weight

(b) There exists a cycle in G having all edges of maximum weight

(c) Edge e cannot be contained in a cycle

(d) All edges in G have the same weight

Q.101 A binary search tree contains the numbers 1, 2, 3, 4, 5, 6, 7, 8. When the tree is traversed in pre-order and the values in each node printed out, the sequence of values obtained is 5, 3, 1, 2, 4, 6, 8, 7. If the tree is traversed in post-order, the sequence obtained would be **[IT-GATE 2005]**

[2-Marks]

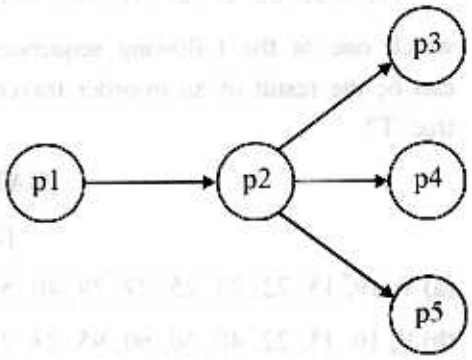
(a) 8, 7, 6, 5, 4, 3, 2, 1

(b) 1, 2, 3, 4, 8, 7, 6, 5

(c) 2, 1, 4, 3, 6, 7, 8, 5

(d) 2, 1, 4, 3, 7, 8, 6, 5

Q.102 In a data flow diagram, the segment shown below is identified as having transaction flow characteristics, with p2 identified as the transaction center



A first level architectural design of this segment will result in a set of process modules with an associated invocation sequence. The most appropriate architecture is: **[IT-GATE 2005]**

[2-Marks]

(a) p1 invokes p2, p2 invokes either p3, or p4, or p5

(b) p2 invokes p1, and then invokes p3, or p4, or p5

(c) A new module Tc is defined to control the transaction flow. This module Tc first invokes p1 and then invokes p2. p2 in turn invokes p3, or p4, or p5.

(d) A new module Tc is defined to control the transaction flow. This module Tc invokes p2, p2 invokes p1, and then invokes p3, or p4, or p5.

Q.103 How many distinct binary search trees can be created out of 4 distinct keys? **[GATE 2005]**

[2-Marks]

(a) 5

(b) 14

(c) 24

(d) 35

Q.104 In a complete k ary, every internal node has exactly k childern. the number of leaves in such a tree with n internal nodes is **[GATE 2005]**

[2-Marks]

(a) n k

(b) (n–1) k+1

(c) n (k–1) + 1

(d) n (k–1)

Q.105 Postorder traversal of a given binary search tree, T produces the following sequence of keys

10, 9, 23, 22, 27, 25, 15, 50, 95, 60, 40, 29

which one of the following sequences of keys can be the result of an in-order traversal of the tree T?

[GATE 2005]

[2-Marks]

(a) 9, 10, 15, 22, 23, 25, 27, 29, 40, 50, 60, 95

(b) 9, 10, 15, 22, 40, 50, 60, 95, 23, 25, 27, 29

(c) 29, 15, 9, 10, 25, 22, 23, 27, 40, 60, 50, 95

(d) 95, 50, 60, 40, 27, 23, 22, 25, 10, 9, 15, 29

Q.106 A scheme for storing binary trees in an array X is as follows. Indexing of X starts at 1 instead of 0, the root is stored at X[1]. For a node stored at X[i], the left child, if any, is stored in X[2i] and the right child, if any, in X[2i+1]. To be able to store any binary tree on n vertices the minimum size of X should be

[GATE 2006]

[1-Mark]

(a) log 2n

(b) n

(c) 2n+1

(d) 2^n-1

Common Data For Questions 107 & 108:

A 3–ary max heap is like a binary max heap, but instead of 2 children nodes have 3 children. A 3-ary heap can be represented by an array as follows. The root is stored in the first location, a [0], nodes in the next level, from left to right, is stored from a [1] to a [3]. The nodes from the second level of the tree from left to right are stored from a [4] location onward. An item x can be inserted into a 3-ary heap containing n items by placing x in the location a [n] and pushing it up the tree to satisfy the heap property.

Q.107 Which one of the following is a valid sequence of elements in an array representing 3-ary max heap?

[GATE 2006]

[2-Marks]

(a) 1, 3, 5, 6, 8, 9

(b) 9, 6, 3, 1, 8, 5

(c) 9, 3, 6, 8, 5, 1

(d) 9, 5, 6, 8, 3, 1

Q.108 Suppose the elements 7, 2, 10 and 4 are inserted, in that order, into the valid 3- ary max heap found in the above question. Which one of the following is the sequence of items in the array representing the resultant heap?

[GATE 2006]

[2-Marks]

(a) 10, 7, 9, 8, 3, 1, 5, 2, 6, 4

(b) 10, 9, 8, 7, 6, 5, 4, 3, 2, 1

(c) 10, 9, 4, 5, 7, 6, 8, 2, 1, 3

(d) 10, 8, 6, 9, 7, 2, 3, 4, 1, 5

Q.109 Let T be a depth search tree in an undirected graph G. Vertices u and v are leaves of this tree, T. The degrees of both u and v in G are at least 2. Which one of the following statements is true?

[GATE 2006]

[2-Marks]

(a) There must exist a vertex w adjacent to both u and v in G

(b) There must exist a vertex w whose removal disconnects u and v in G

(c) There must be exist a cycle in G containing u and v

(d) There must exist a cycle in G containing u and all its neighbours in G

Q.110 The height of a binary tree is the maximum number of edges in any root to leaf path. The maximum number of nodes in a binary tree of height h is: **[GATE 2007]**

[1-Mark]

(a) 2^h-1

(b) $2^{h-1}-1$

(c) $2^{h+1}-1$

(d) 2^{h+1}

Q.111 The inorder an preorder traversal of a binary tree are d b e a f c g and a b d e c f g, respectively. The postorder traversal of the binary tree is **[GATE 2007]**

[2-Marks]

(a) d e b f g c a

(b) e d b g f c a

(c) e d b f g c a

(d) d e f g b c a

Q.112 The maximum number of binary trees that can be formed with three unlabeled nodes is: **[GATE 2007]**

[1-Mark]

(a) 1

(b) 5

(c) 4

(d) 3

Q.113 A complete n-ary tree is a tree in which each node has n children or no children. Let I be the number of internal nodes and L be the number of leaves in a complete n-ary tree. If L = 41, and I = 10, what is the value of n? **[GATE 2007]**

[2-Marks]

(a) 3

(b) 4

(c) 5

(d) 6

Q.114 What is the maximum height of any AVL-tree with 7 nodes ? Assume that the height of a tree with a single node is 0.

[GATE 2009]

[2-Marks]

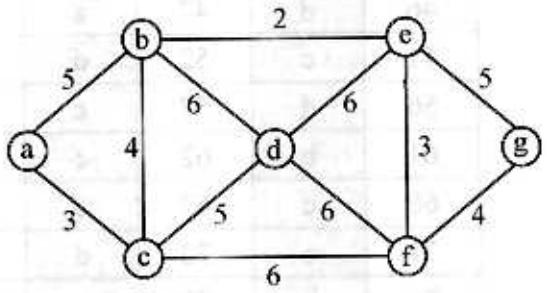
(a) 2

(b) 3

(c) 4

(d) 5

Q.115 Consider the following graph:



Which one of the following is NOT the sequence of edges added to the minimum spanning tree using Kruskal's algorithm ?

[GATE 2009]

[2-Marks]

(a) (b, e) (e, f) (a, c) (b, c) (f, g) (c, d)

(b) (b, e) (e, f) (a, c) (f, g) (b, c) (c, d)

(c) (b,c) (a,c) (e,f) (b,c) (f,g) (c,d)

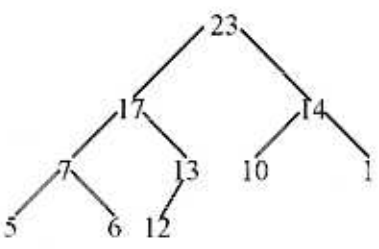
(d) (b, e) (e, f) (b, c) (a, c) (f, g) (c, d)

ANSWER KEY

1	d	2	c	3	b	4	c	5	c
6	c	7	c	8	a	9	c	10	b
11	d	12	b	13	d	14	a	15	c
16	c	17	d	18	b	19	a	20	a
21	a	22	a	23	c	24	d	25	d
26	c	27	b	28	d	29	c	30	b
31	c	32	a	33	b	34	d	35	b
36	a	37	d	38	b	39	c	40	d
41	d	42	d	43	c	44	b	45	b
46	d	47	a	48	d	49	c	50	b
51	c	52	d	53	d	54	d	55	b
56	d	57	c	58	c	59	b	60	b
61	b	62	c	63	b	64	b	65	c
66	c	67	c	68	c	69	b	70	b
71	c	72	d	73	b	74	b	75	b
76	b	77	a	78	b	79	c	80	d
81	a, b	82	d	83	d	84	b	85	b
86	b	87	b	88	c	89	c	90	b
91	b	92	a	93	b	94	d	95	c
96	c	97	a	98	a	99	b	100	c
101	d	102	c	103	b	104	c	105	c
106	d	107	d	108	a	109	a	110	c
111	a	112	b	113	c	114	b	115	d

SOLUTIONS

S.4 (c)



S.5 (c)

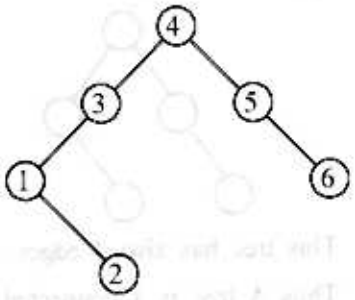
Inorder traversal of BST is sorted array of element. {9, 85, 47, 68, 43, 57, 55} does't give value in sorted order.

S.6 (c)

Let a sequence

4, 3, 1, 2, 5, 6

BST:



Here, $p = 2$

and $n = 6$

So, first number to the inserted in the tree is $n - p = 6 - 2 = 4$

S.7 (c)

Tree contains 7 nodes ie. 10, 20, 40, 50, 70, 80, 90. ∴ when searching key value 60 it have 2^7 different order in which these key can occur on search path.

$2^7 = 128$.

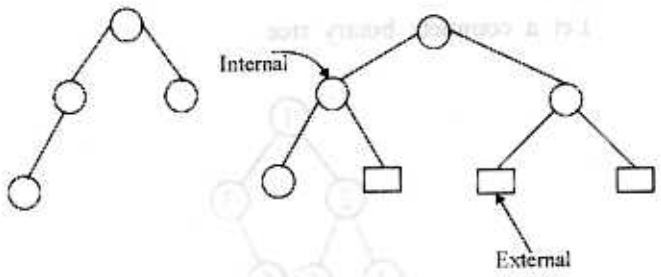
S.11 (d)

There is minimum comparison and swapping required to insert an element in **ordered linked list** in comparison to BST, ordered array and Heap.

S.12 (b)

In the extended tree converted from binary tree then the original nodes become the **internal nodes** in the extended tree and the new nodes are the external nodes in the extended tree.

For example:



S.14 (a)

(i) and (iii) are correct (Basic terminology)

(ii) ⇒ The nodes with same level are said to belong to the same generation.

S.15 (c)

In the complete binary tree T_n, specifically the left and right children of the node k are respectively 2* k and 2*k+1.

∴ for nodes 24 and 25 parent would be $\frac{24}{2}$ and $\frac{25-1}{2} = 12$

S.17 (d)

Binary search requires the list to be a sorted list. So if the list is unsorted, then we cannot determine the number of comparisons equired. It may perform $\log_2 n$ comparison (maximum number of comparison for binary search) and not find the element though the element may be present, as the list is not sorted.

S.18 (b)

For a sorted list of n elements, the maximum number of comparisons required is $\log_2 n$. So for two different mutually exclusive lists, the number of comparisons required is

$\log_2 n + \log_2 n = 2 \log_2 n$

$= \log_2 n^2$

S.21 (a)

In a top down multiways tree a semileaf must be either full or a leaf.

S.22 (a)

If the keys are **uniformly distributed between k(0) and k (n–1)** the interpolation search is even more efficient than binary search.

S.23 (c)

(i) ⇒ It involves arithmetic on keys and complex multiplications and divisions.

(ii) ⇒ It requires only arithmetic on integer indexes and division by.

S.25 (d)

(i) and (ii) are incorrect.

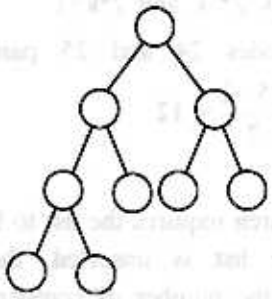
The correct f(n) values for (i) and (ii) are O ($\log_2 n$) and O (n) respectively.

S.27 (b)

The average running time to search an item in the binary search tree is proportional to $\log_2 n$ in the extended tree.

S.28 (d)

Let a complete binary tree is



non-leaf node = 4

leaf node = 5

total node = 4 + 5 = 9

$= 2 \times 4 + 1$

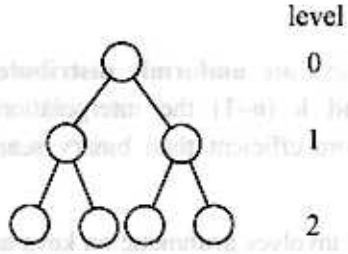
So, if non-leaf node is n then total node in the tree is **2n + 1.**

S.29 (c)

An Extended binary tree tree is a binary tree in which each node has 0 or 2 children, such a tree is called a 2 tree.

S.30 (b)

Let a full tree

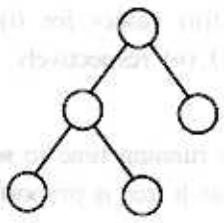


For level 2, max node = $7 = 2^n - 1$

So, for N levels max nodes are = $2^{N+1} - 1$

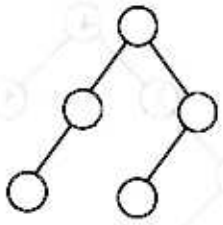
S.31 (c)

(i) Let a tree with 5 nodes.



This tree has 4 edges.

(ii) Again let tree is

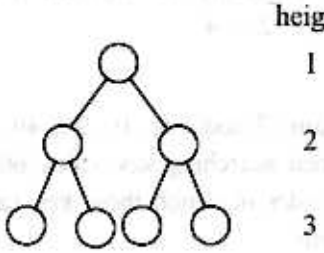


This tree has also 4 edges.

Thus A tree is a connected component, so there will be **(n–1) edges** for n nodes.

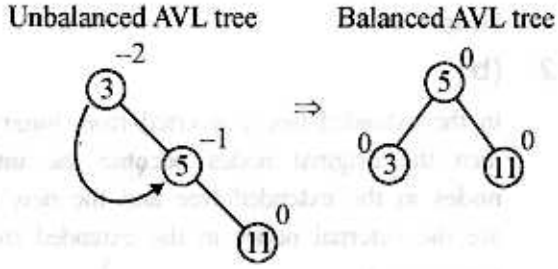
S.32 (a)

Let a full binary tree with three internal nodes.



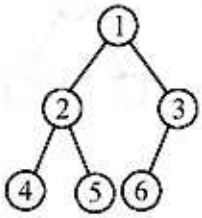
So for 3 internal nodes height is 2 that is equal to $\lceil \log_2 3 \rceil$ similarly for N internal nodes height of a full binary tree is $\lceil \log_2 N \rceil$.

S.33 (b)



S.35 (b)

Let a complete binary tree.



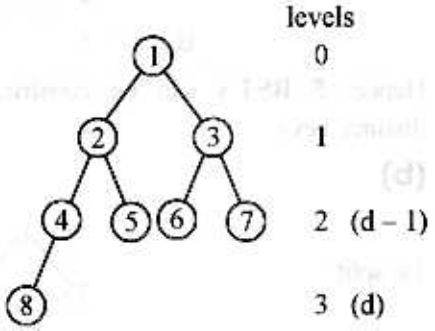
Most distant two nodes in this tree are (4) and (6) and distance between them is 4 and total nodes in this tree is 6.

So, maximum distance can be deducted as $2 \log_2 6 = 4$.

So, for n nodes the distance will be = **$2 \log_2 n$.**

S.36 (a)

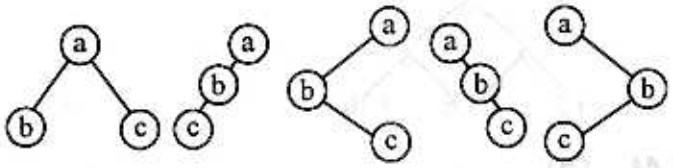
Let a complete binary tree.



From the diagram, it is clear that **node at level less than (d – 1) has two sons.** As node (2) and (3) have two sons.

S.37 (d)

Let three nodes (a), (b), (c), taking (a) as root node, no. of possible trees are



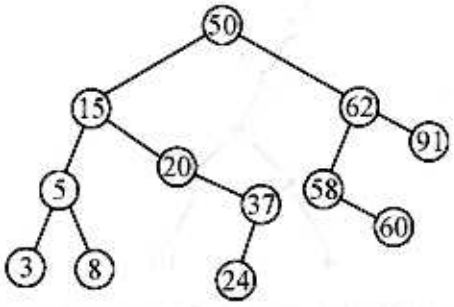
total = 5

Similarly for (b) and (c) as root node.

So, total binary trees are = 5 + 5 + 5 = **15.**

S.38 (b)

Binary search tree after insertion is:



So nodes in left and right subtree are **(7, 4)** respectively.

S.39 (c)

Complexity is multiple of n.

If root node does not satisfy the heap property of a complete binary tree we have to visit all nodes of the binary tree to find max node of tree.

So complexity of this operation is n and after searching max node, required time to place it at root node is log n.

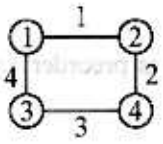
So total time complexity is O(n log n).

S.40 (d)

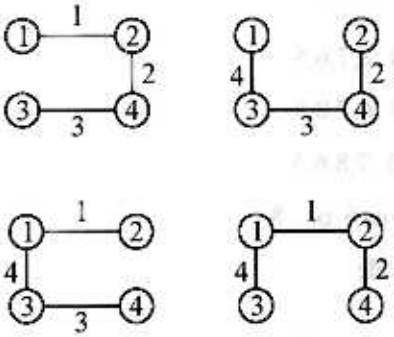
Simple connected graph n-vertices and n-edges.

Let n = 4

Then connected graph is



Possible spanning trees of this graph are:



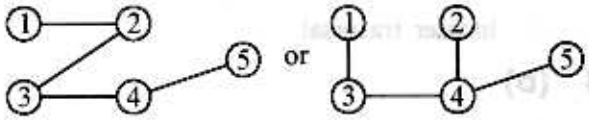
Total trees, m = 4

So, maximum value of m = n.

S.41 (d)

(i) Spanning tree always have |V| – 1 edges.

Ex.:

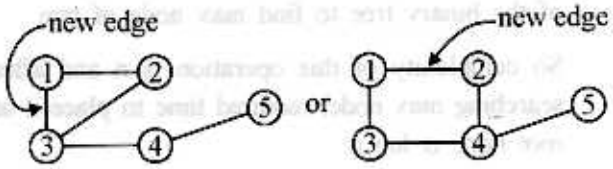


|V| = 5

edges = |V| – 1 = 4, that is true.

(ii) Adding an edge to spanning tree always creates a cycle.

Ex.: From above example, adding an edge to the spanning tree.

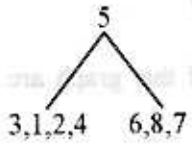


So, (ii) is true.

(iii) The intermediate graph obtained from both Prim's and Kruskal's algorithm in finding minimum cost spanning tree are always trees, there is no cycle in intermediate graph, so it will be in the form of tree.

S.42 (d)

5,3,1,2,4,6,8,7 is preorder ⇒ Root Node is 5



B 1,2,3,4 8,7,6,5

C 2,1,4,3 6,7,8,5

D 2,1,4,3 7,8,6,5

⇒ node is 6 or 8

if 6 is node

S.43 (c)

Option II → 52, 97, 121, 195, 242, 381, 472.

(Increasing order)

∴ Inorder traversal.

S.44 (b)

Total number of distinct binary search tree with 3 keys is

$$B_n = \frac{1}{n+1}{}^{2n}C_n$$

$$= \frac{1}{4}{}^{6}C_3$$

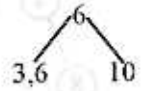
$$= \frac{1}{4} \times \frac{\lfloor 6}{\lfloor 3 \lfloor 3}$$

$$B_n = 5$$

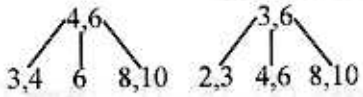
Hence, 5 **BST's** can be constructed with 3 distinct keys.

S.45 (b)

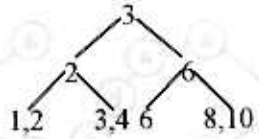
1st split:



2nd split:



3rd split:



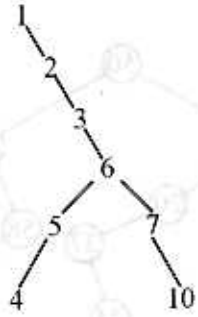
S.46 (d)

(b) and (d) has inorder as BCAD out of these two (d) has the preorder as ABCD.

S.47 (a)

Arranging the number

1,2,3,6,5,7,10,4 into a BST



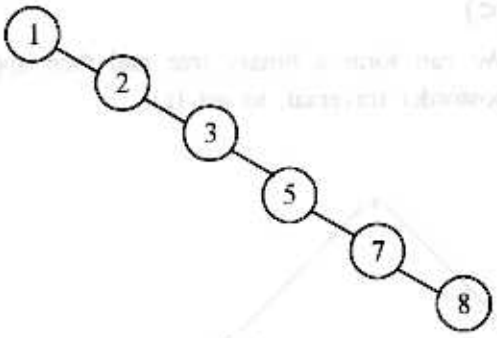
Insert the choices according to the binary search tree construction algorithm. Only choice (A) gives a binary search tree of height 5.

S.48 (d)

At least 2 edges should have same weight to get multiple minimum spanning trees. Here we are not adding any duplicate weighted edge to the graph. So it is **not possible to get more than one minimal spanning tree.**

S.49 (c)

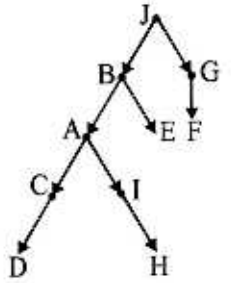
The choice (c) gives the following binary search tree in which both in order and preorder traversals are same. The sequence should always be in increasing or decreasing order to get such property.



S.50 (b)

According to the binary search tree the left child value should always be less than root node value and the right node value should always be greater than the root node. So, the tree should be visited **in order to** get the numbers in ascending order.

S.51 (c)



as here F is not at the lowermost level, so statement (iv) is false

S.52 (d)

BST has 1,2,3,4,5,6,7,8 as its node values.

Traversal is

5 3 1 2 4 6 8 7

The inorder traversal will be always in order hence inorder is

1 2 3 4 5 6 7 8

Now create the tree with the help of these

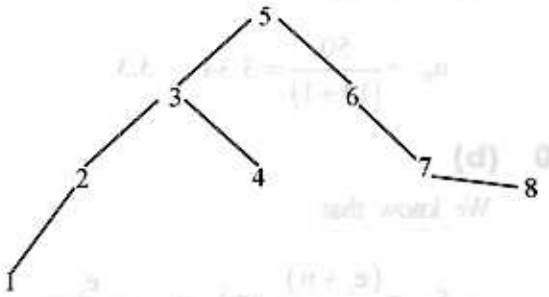
PRE–NLR IN→LNR

⇒ 5 is the node

5 3 1 2 4 6 8 7 – PRE

1 2 3 4 5 6 8 7 – IN

3 will be node 6 will be node



Post order LRN = **1 2 4 3 8 7 6 5**

S.53 (d)

Let k = 550 for complete binary tree T_n children can be found by

2 * k = 2 * 550 = **1100** and 2*k + 1 = **1101**

S.54 (d)

Since all of the items are sorted we are inserting these sorted items into an empty BST. Hence they will always be inserted into an increasing order like a single branch towards right, every element is inserted to its right side forming a single branch.

S.55 (b)

The depth of such a single branch of n node will be **n–1.**

S.56 (d)

(a), (b), (c) are correct statement.

(b) ⇒ (i), (ii), (iv) are similar tree ⇒ Definition

(b) ⇒ (i), (iii) are copies of each other ⇒ Definition

(c) because in a binary tree, we distinguish between a left successor and a right successor even when there is only one successor.

S.57 (c)

By applying rules for converting a binary tree to a 2-tree, we get the tree as shown in option (C).

S.59 (b)

We have $u_n = e_n / (n + 1)$

where u_n = Average number of comparisons required for an unsuccessful search

e_n = external path length

n = number of nodes.

Using this we get,

$$u_n = \frac{50}{(14+1)} = 3.34 = \mathbf{3.3}$$

S.60 (b)

We know that

$$S_n = \frac{(e_s + n)}{n} \text{ and } u_s = \frac{e_s}{(n+1)}$$

$$\therefore \mathbf{S_s = \left(\frac{u+1}{n}\right) u_s + 1}$$

S.61 (b)

The time required to search a binary search tree variers between **O(n) and O (log n)**, depending on the structure of the tree.

S.62 (c)

The internal path length of binary tree is the sum of the levels of all the nodes in the tree. In the given tree, path length is evaluated as follows.

One node at level 0,2 nodes at level 1,4 nodes at level 2,4 nodes at level 3 and 2 nodes at level 4.

$\therefore$ 1 * 0 + 2 * 1 + 4 * 2 + 4 * 3 + 2 * 4 = **30**

S.63 (b)

a – iv, b – iii, c – ii, d – i

0 – regular $\Rightarrow$ consists of disconnected vertices i.e., zero edges.

1 – regular $\Rightarrow$ consists of disconnected edges, i.e., may be one edge between, any two vertices.

2 – regular $\Rightarrow$ consists of disconnected cycles

3 – regular $\Rightarrow$ is known as cubic graph.

S.64 (b)

The left rotation of a tree is found by the algorithm:

q = right (p)

hold = left (q)

left (q) = p

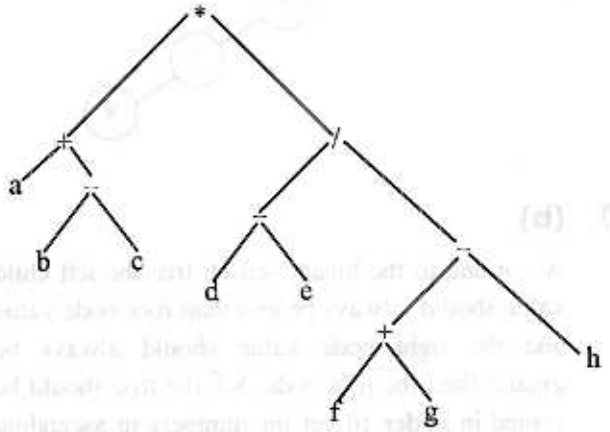
right (p) = hold;

S.65 (c)

A Hamiltonian path is one that contains each vertex exactly once. A Hamiltonian circuites a circuit that contains each vertex exactly once except for the first vertex.

S.66 (c)

We can form a binary tree and then apply a postorder traversal, to get (c).



The postorder traversal, we traverse nodes in the order of Left node, right node then root node.

Now traversal sequence is

a

$\Rightarrow$ ab

$\Rightarrow$ abc

$\Rightarrow$ abc –

$\Rightarrow$ abc – +

$\Rightarrow$ abc – + d

$\Rightarrow$ abc – + de

Similarly, applying, finally we get

abc – + de – fg + h – / *

S.67 (c)

If a heap has 7 nodes, then

Depth $= \log_2 n$

$= \log_2 7$

$= 3$ (approx). (n is no of nodes)

Thus has a depth of **3.**

S.68 (c)

We know that

$$e = i + 2n$$

where

$e \Rightarrow$ external path length

$i \Rightarrow$ internal path length

$n \Rightarrow$ number of nodes

$e = i + 2n$

$i = e - 2n = 60 - 2*14 = 60-28$

$= \mathbf{32}$

S.69 (b)

The average number of comparisons for a successful search in binary search tree is **(i + n)/n.**

S.70 (b)

With reference to (a) $\Rightarrow$ (i+n)/n

$\Rightarrow$ (30 + 13)/ 13

$\Rightarrow$ **3.31**

S.71 (c)

As the problem definition says 'n' vertices are divided into 'k' trees. Let these trees are having $x_1, x_2, x_3 \ldots\ldots x_k$ vertices respectively. The sum of x1, x2.....xk should be **n** as we have only 'n' vertices in total. The definition of tree says that any tree with 'u' vertices will have **u** –1 edges. So all the 'k' tree will have

'xl–1+ x2 – 1+.....+ xk – 1' edge.

Result = x1 –1 + x2 – 1+.....+xk – 1

= x1 + x2+.....+xk–K

= **n–K**

S.72 (d)

In order : L, Root, R

Preorder : Root, L, R

Postorder : L, R, Root

Thus we see that in inorder root is in middle. In postorder, root is end and in preorder, root is at the starting.

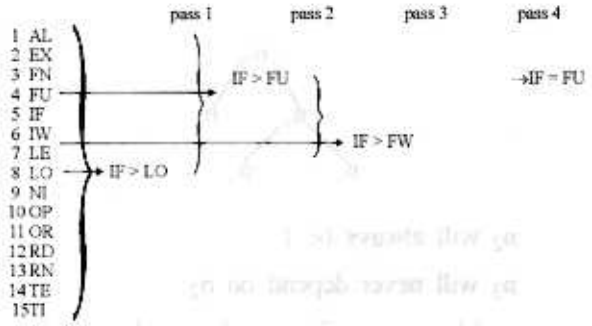
From given data we see K is in middle in III, starting in II and end in (I)

$\therefore$ I $\rightarrow$ Postorder

II $\rightarrow$ **Preorder**

III $\rightarrow$ **Inorder**

S.73 (b)



15 item are there let an array a [15].

$\Rightarrow$ a [0] = AL and a [14] = TI

$\Rightarrow$ mid = (0+15)/2 = 7

$\therefore$ a[7] = LO

Now compare LE with IF

$\therefore$ IF < LO $\Rightarrow$ choose 0–7 range

mid = (0+7)/2 = 3

a[3]= FU

$\therefore$ IF > FU $\Rightarrow$ choose 4 to 7 range

mid = (4+7)/2 = 5

a [5] = IW

$\therefore$ IF < IW $\Rightarrow$ choose 4 – 5 range

$$\text{mid} = \frac{4+5}{2} = 4$$

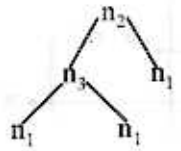
S.74 (b)

Here we assume the root R of the tree T is stored at TREE [1]. In the sequential/ single array representation, if a node is at K position then its left child is stored at **2* K** and its right child is stored at **2 * K + 1** position.

S.75 (b)

Let T be any ordered tree and let A be the set of vertices of T. Define a binary positional tree B(T) on the set of vertices A, as follows. If $v \in A$, then the left offspring v_l of v in B (T) is the first offspring of v in T (in the given order of siblings in T), if it exists. The right offspring v_g of v in B(T) is the next sibling of v in T (in the given order of siblings in T), if it exits.

S.76 (b)



n_2 will always be 1.

n_3 will never depend on n_2.

and here: $n_1 = 3$, $n_2 = 1$, $n_3 = 1$

$n_3 = n_1 - 2 = 3 - 2 = 1$

S.77 (a)

2 degree mode is n_1 and $n_2 = 1$

so, removing n_2, 2 edges will be removed.

Acc. to ques. 1 new edge is added.

Total No. of edges in binary tree

$$= \frac{2n_2 + 3n_3 + n_1}{2}$$

Here $n_2 = 1$

$n_3 = n_1 - 2$

$n_1 = n_1$

$\therefore$ No of edges

$$= \frac{\left(2 + 3(n_1 - 2)\right) + n_1}{2}$$

$$= \frac{2 + 3n_1 - 6 + n_1}{2}$$

$$= 2\,(n_1 - 1)$$

S.78 (b)

We know,

$$S_n = \frac{(i_n + n)}{n}$$

where,

S_n = average number of comparisons

i_n = Average internal path length

n = number of nodes

$$4.31 = \frac{i_n + 15}{15}$$

$i_n = 4.31 * 15 - 15 = 49.65 \cong \mathbf{50}$.

S.79 (c)

We know that

$$u_e = \frac{e_n}{(n+1)}$$

Where, u_n =Average number of comparisons required for an unsuccessful search.

e_n = Average external path length, and

n = nodes

So, here

$$4.0 = \frac{56}{n+1}$$

$\Rightarrow$ **n = 13**

S.82 (d)

In Kruskal's algorithm, locating the root of a tree has time complexity O(log n), where n is the number of vertices. If m is the number of edges then time complexity of Kruskal's algorithm is **O(m log n).**

S.84 (b)

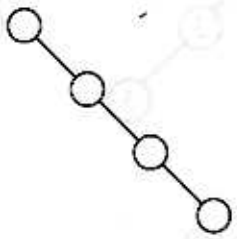
Optimal binary search tree is constructed efficiently by using dynamic programming.

If prefix and postifix forms are given, one cannot uniquely construct the binary tree for the given forms. Depth first search is used to find the connected components of a graph. Breadth first search can also be used for finding connected components.

S.85 (b)

We will consider two binary tree and evaluate each of the given options.

Binary Tree 1



n = 1, Number of nodes of degree '2' = 0

(a) $\log_2 n = \log_2 1 = 0$, hence True.

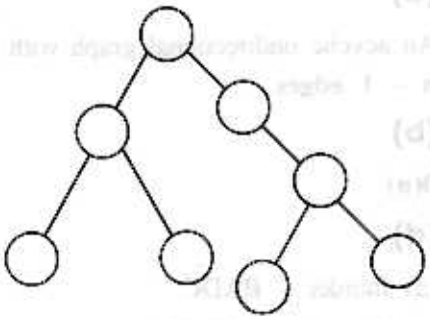
(b) n – 1 = 1 – 1 = 0, hence True.

(c) n = 1 , hence false.

(d) $2^n = 2^1 = 2$, hence false.

Hence option (c) and (d) are ruled out.

Binary Tree 2



n = 4, number of nodes of degree '2' = 3

(a) $\log_2 n = \log_2 4 = \log_2 2^2 = 2$, hence false.

(b) n – 1 = 4 – 1 = 3, hence true

(c) n = 4, hence false

(d) $2^n = 2^4 = 16$ hence false.

Thus option (a) is also ruled out.

So for any binary tree with n leaf nodes, number of nodes of degree 2 will be **(n – 1).**

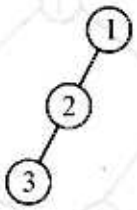
S.86 (b)

We will check each of the given options one by one.

(i) A tree with n nodes, can have (n – 1) edges as shown in the below example.

Number of nodes = 3

Number of edges = (3 – 1) = 2, which is true.



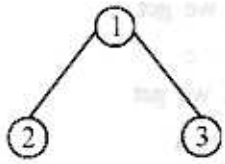
(ii) For any rooted binary tree, if its postorder and preorder traversals are given , it cannot be uniquely constructed. So, it is false.

(iii) for a complete binary tree with n internal nodes can always have (n+1) leaves.

eg. If internal node is only one.

Number of leaves = 1+1 = 2 [True]

(iv) Maximum number of nodes in a binary tree of height h is always $(2^{h+1}-1)$



So, it is true.

S.87 (b)

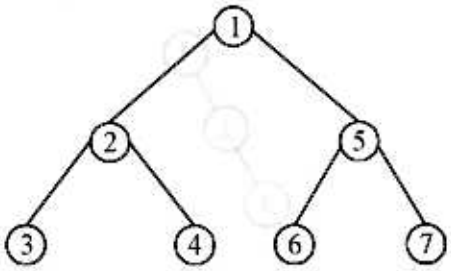
In a B^+ Tree, the leaf nodes are linked together to provide order access, hence **range queries are faster for B^+ trees.**

S.88 (c)

Heap is a way of sorting. In breadth first search one need to guarantee that no node is processed more than once which is accomplished by using a queue to hold nodes that are waiting to be processed. Depth first search uses a stack instead of the queue.

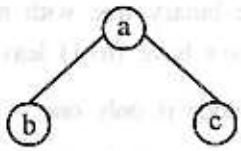
S.89 (c)

Based on the given representation (1(2 3 4)(5 4 6)) can be represented into binary tree as:



S.90 (b)

Consider the following binary tree.



(i) Preorder traversal would result into abc.

(ii) Postorder traversal would result into bca.

(iii) Inorder traversal would result into bac.

From (i), we get

Last pre = c .....(i)

From (ii), we get

Last post = a .....(ii)

From (iii), we get

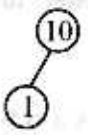
Last in = c .....(iii)

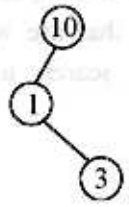
From (i), (ii) and (iii) we get

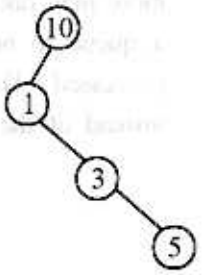
Last pre = Last in.

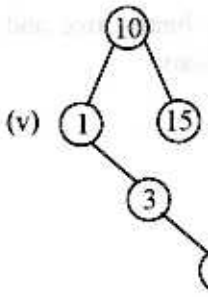
S.91 (b)

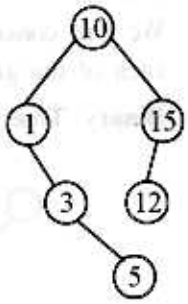
(i) 

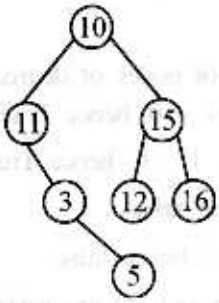
(ii) 

(iii) 

(iv) 

(v) 

(vi) 

(vii) 

So, the height of tree is **3.**

S.92 (a)

An acyclic undirectional graph with n node has **n – 1 edges.**

S.93 (b)

0(n)

S.94 (d)

(a) Inorder – BADC
Preorder – ABCD

(b) Inorder – BACD
Postorder – ABCD

(c) Inorder – ACBD
Preorder – ABCD

(d) Inorder – BCAD
Preorder – ABCD

S.95 (c)

Number inserted in binary search tree are

1, 2, 3, ..., n

In resulting tree, the right subtree of the root contains p nodes.

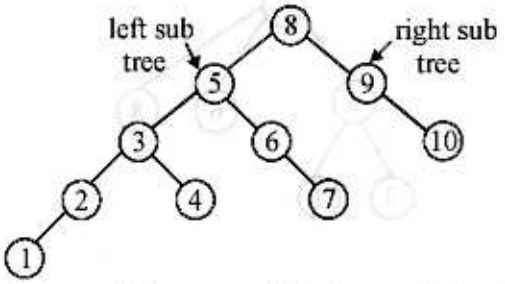
Let number inserted are 1,2,3,4,5,6,7,8,9,10

and p = 3 (number of nodes in right subtree of the root)

(i) if first inserted number is n – p + 1

$= 10 - 3 + 1 = 8$

the possible binary search tree is

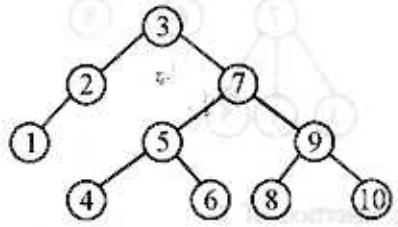


n – p + 1 does not satisfy the required condition.

(ii) if first inserted number is p

$= 3$

Then in possible binary tree 7 node will be in right subtree as



p does not satisfy the required condition.

(iii) if first inserted number is p + 1 = 4, then 6 nodes will be in right subtree, that does not satisfy the condition.

(iv) Similarly **n – p = 10 – 3 = 7, then 3 nodes will be right subtree that satisfy the condition.**

S.96 (c)

n vertices & k edges are marked as tree edges then in depth first traversal of a graph, **the number of connected components will be n – k – 1**

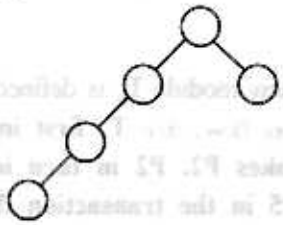
S.98 (a)

In B-tree if N-pointers of a node then there will be (N – 1) keys.

So in a four level B-tree if a new key inserted then newely created nodes will be 5.

S.99 (b)

Let a Binary tree having difference between the number of nodes in the left and right subtree is at most 2.



No. of nodes in tree is n = 5 and height is h = 3

So $\mathbf{n = 2^{h-1} + 1}$

$= 2^{3-1} + 1$

$= 5$

S.100 (c)

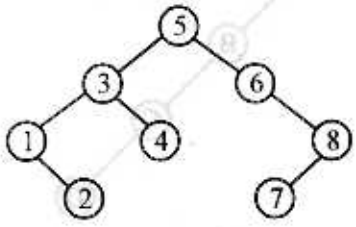
If we apply algorithm of finding minimum spanning tree, then only reason of having maximum weight edge 'e' in minimum spanning tree is that graph itself a tree, means only n–1 edges are there with n nodes. In all above there is no cycle.

Graph which is tree

There is loop, so node n can be through only e

S.101 (d)

In pre-order traversal of binary tree we traverse the nodes in the sequence of root node, left node then right node sequence values obtained by pre-order traversal is 5, 3, 1, 2, 4, 6, 8, 7. So possible binary tree is



In post-order, we traverse the binary tree, in the sequence of left node, right node then root node.

So sequence obtained would be 2, 1, 4, 3, 7, 8, 6, 5.

S.102 (c)

When a new module T_c is defined to control the transaction flow, this **T_c first invokes P1 and then invokes P2. P2 in turn invokes P3, or P4, or P5 in the transaction flow.**

S.103 (b)

Total number of distinct binary search tree with n nodes (key).

$$B_n = \frac{1}{n+1}{}^{2n}C_n$$

$$B_4 = \frac{1}{5}\frac{\lfloor 8}{\lfloor 4 \lfloor 4}$$

$$B_4 = \frac{1}{5} \times \frac{8 \times 7 \times 6 \times 5 \times \lfloor 4}{4 \times 3 \times 2 \times 1 \times \lfloor 4}$$

$$B_4 = 14$$

S.104 (c)

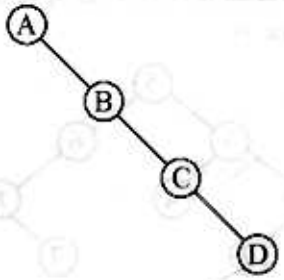
In a complete k ary tree. if every internal node has exactly k children then (k–1) key contains no leaves. If there are n internal nodes, then number of leaves is **n(k–1) +1.**

S.105 (a)

In-order traversal of a binary search tree, tree is always in sorted order. i.e. increasing order.

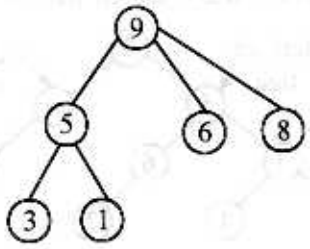
S.106 (d)

For a right sewed binary tree, number of nodes will be $2^n - 1$. For example, in below binary tree, nod 'A' will be stored at index 1, 'B' at index 3, 'C' at index 7 and 'D' at index 15.



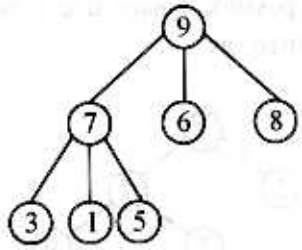
S.107 (d)

Upper node, should be greater than its, subsidiary nodes ∴ By observation (a), (b), (c) are not correct.

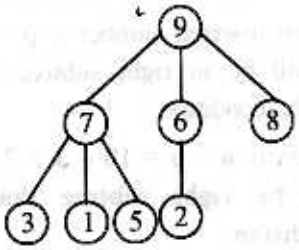


S.108 (a)

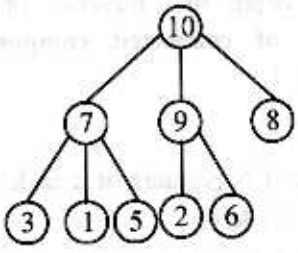
After insertion of 7



After insertion of 2

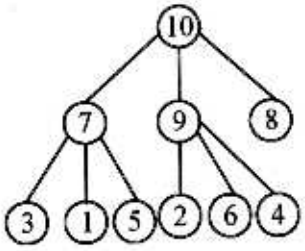


After insertion of 10



After insertion of 4

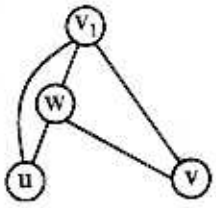
Resultant heap = **10, 7, 9, 8, 3, 1, 5, 2, 6, 4**



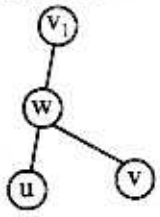
S.109 (a)

Consider the following graph G

G



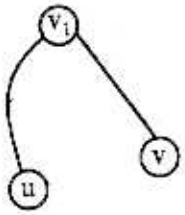
T: Depth First Search of G



According to the equation u and v are leaves of the tree T and the degree of both u and v in G are 2. Consider each choice separately T.

Choice (a). There must exist a vertex w adjacent to both u and v in G, so it is true.

Choice (b). There must exist a vertex w whose removal disconnects u and v in G is false because when we remove w from G graph is connected as



Choice(c). There must exist a cycle in G containing u and v is false because in given graph there are two cycles v_1 wvv_1 and v_1 wuv_1 So both vertex u and v lie in different cycle.

Choice (d). There must exist a cycle in G containing u and all its neighbours in G is also false because both vertex u & v do not lie in a single cycle of G.

S.110 (c)

For a single node tree consisting only of the root h = 0, nodes = 1. So this reduces the choice to a or c. For a three node complete binary tree h = 1 and nodes = 3 so this reduces the choice to c.

or

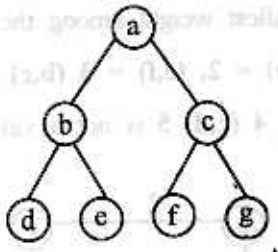
Maximum number of nodes will be there for a complete tree.

Number of nodes in a complete tree of height $h = 1 + 2 + 2^2 + 2^3 + \ldots + 2^h = 2^{(h+1)} - 1$

S.111 (a)

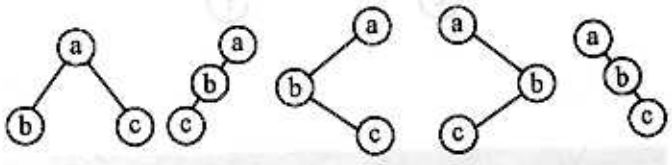
From the preorder traversal the root of the tree is evidently a. The left subtree from the inorder traversal is dbe and the right subtree from the inorder traversal is fcg.

From the preorder traversal we see cfg i.e. the root of the right subtree is c and the left and right children are f and g. from the preorder traversal we see that for the inorder dbe, b is the root of the left subtree and d and e are the left and right children. From a postorder traversal of the right subtree we must have deb and only (a) satisfies this.The entire tree is drawn below.



S.112 (b)

Possible trees with three unlabeled nodes are



S.113 (c)

For an n-ary tree where each node has n children or no children, following relation holds

$$L = (n - 1)*I + 1$$

where, L is the number of leaf nodes and I is the number of internal nodes.

$\therefore$ for n = 41, I = 10

$41 = 10*(n-1) + 1$

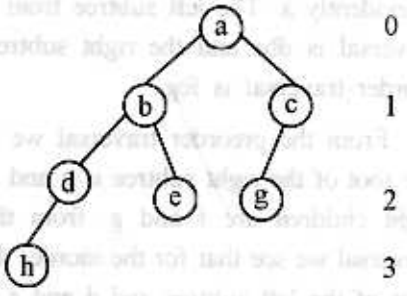
$\Rightarrow$ **n = 5**

S.114 (b)

AVL trees are binary trees with the following restrictions:

(i) the height difference of the children is at most 1.

(ii) both children are AVL trees.

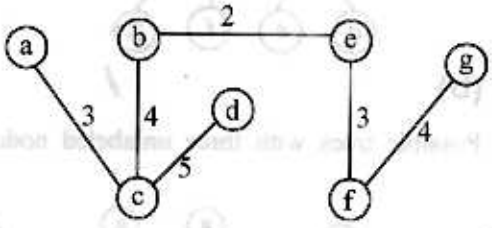


S.115 (d)

In kruskal's algo we chose an edge of G which has smallest weight among the edges of G.

So, **(b,e) = 2, (e,f) = 3 (b,c) = 4 (a,c) = 3**

(f, g) = 4 (c,d) 5 is not a valid sequence.



5.3 SORTING AND SEARCHING

LEVEL–1

Q.1 Arrange the following functions in increasing asymptotic order

(a) $n^{1/3}$

(b) c^n

(c) $n^{7/4}$

(d) nlog n

(e) 1.0000001^n

(a) a, d, c, e, b

(b) d, a, c, e, b

(c) a, c, d, e, b

(d) a, c, d, b, e

Q.2 Consider the undirected graph G defined as follows. The vertices of G are bit strings of length n. We have an edge between vertex u and vertex v if u and v differ in exactly one bit position (in other words, v can be obtained from u by flipping a single bit). The ratio of chromatic number of G to the diameter of G is

(a) $\frac{1}{2^{n-1}}$

(b) $\frac{1}{n}$

(c) $\frac{2}{n}$

(d) $\frac{3}{n}$

Q.3 Maximum number of edges in an n-node undirected simple graph without cycles is

(a) n–1

(b) $\frac{n(n+1)}{2}$

(c) $\frac{n(n-1)}{2}$

(d) n

Q.4 How many simple distinct unlabeled graphs are possible with 5 vertices and 2 edges?

(a) 1

(b) 2

(c) 10

(d) 8

Q.5 What is the time complexity for selection sort to sort array of n elements?

(a) O (logn)

(b) O(n)

(d) O (n log n)

(d) O (n^2)

Q.6 Number of comparisons required by bubble sort to sort an array of n element is

(d) $n^2/4$

(b) O(n)

(c) n (n–1)/2

(d) (n–1)

Q.7 What is the order of complexity in bubble sort in worst case.

(a) O(n)

(b) $O(n^2)$

(c) O(n log n)

(d) O (log n)

Q.8 How can we calculate the total number of comparisons in bubble sort with n elements after k iterations.

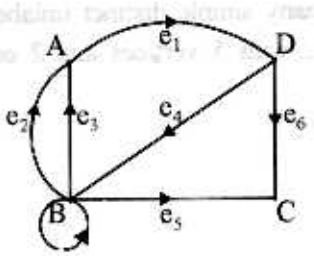
(a) $\frac{(k^2-k)}{2}$

(b) $\frac{(2kn-k^2+k)}{2}$

(c) $\frac{(2kn-k^2-k)}{2}$

(d) $\frac{(2kn+k^2-k)}{2}$

Q.9 Consider the following graph G,



Which of the following statement is correct?

(i) Graph G is strongly connected

(ii) Graph G is unilaterally connected

(iii) Node C is a sink and A is a source node

(a) (i), (ii), (iii)

(b) (ii), (iii)

(c) (ii)

(d) (i), (iii)

Q.10 Searches in which the entire table is constantly in main memory are called–

(a) External searches

(b) Internal searches

(d) Linear time

(d) Quardratic time

Q.11 What is the largest integer m such that every simple connected graph with n vertices and n edges contains at least m different spanning trees?

(a) 1

(b) 2

(c) 3

(d) n

Q.12 If all the edge weights of an undirected graph are positive, then any subset of edges that connects all the vertices and has minimum total weight is a

(a) Hamiltonian cycle

(b) grid

(c) hypercube

(d) tree

Q.13 The tightest lower bound on the number of comparisons, in the worst case, for comparison based sorting is of the order of :

(a) n

(b) n^2

(c) n log n

(d) $n \log^2 n$

Q.14 A sorting technique is called stable when

(where k denotes key and r denote records)

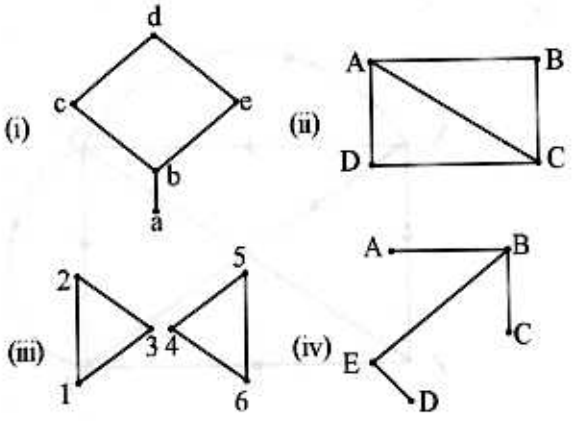
(a) for all records i and j such that k[i] equals k[j]

(b) for all records if r[i] precedes r[j] in the original file, then r[i] precedes r[j] in the sorted file.

(c) (a) and (b)

(d) Neither (a) nor (b)

Q.15 For the following graphs, which can have a possible Hamiltonian path?



(a) (iii)
(b) (i), (ii)
(c) (i), (ii),(iv)
(d) (ii),(iv)

Q.16 The best average behavior is shown by
(a) Quick sort
(b) Merge sort
(c) Insert sort,
(d) Heap sort

Q.17 The average computing time of Heap sort is
(a) O (n^2)
(b) O (n log n)
(c) O (log n)
(d) O (n^3)

Q.18 The worst case time complexity of insertion sort is
(a) O (n^2)
(b) O (n^3)
(c) O (n log n)
(d) O (log n)

Q.19 Example (s) of O (N^2) algorithms (are)
(a) initializing all the elements in a two dimensional array to zero.
(b) printing out all the elements in a two dimensional array
(c) searching for the smallest element in an unsorted two dimensional array
(d) all of the above.

Q.20 A sorting technique is called stable if
(a) It takes (n log n) time
(b) It maintains the relative order of occurrence of non distinct elements
(c) It uses divide and conquer paradigm
(d) It takes O (n^2) space.

Q.21 In a directed graph G(V,E); always.
(a) $E \subseteq V \times V$
(b) $E \not\subset V \times V$
(c) $E = V \times V$
(d) $E \neq V \times V$

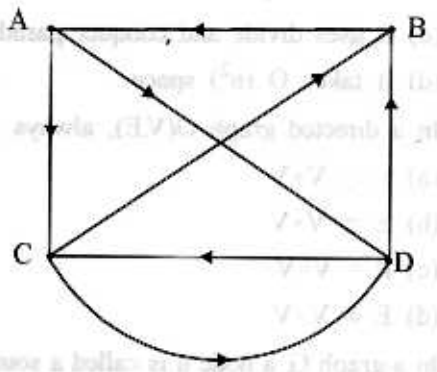
Q.22 In a-graph G, a node u is called a source if it has a
(i) positive outdegree
(ii) zero indegree
(iii) zero outdegree
(iv) positive indegree
(a) i, ii
(b) iii, iv
(c) i, iii
(d) ii, iv

Q.23 O (n^2) means
(i) "oh of n squared"
(ii) "big–oh of n squared"
(a) Only i
(b) Only ii
(c) Both i and ii
(d) neither i nor ii

Q.24 A path P of length 5 from a node k to a node m in a graph G is a sequence of –
(a) 4 nodes
(b) 5 nodes
(c) 6 nodes
(d) nodes is not possible

Q.25 Which of the following sorting method is called diminishing increment sort
(a) Merge sort
(b) Shell sort
(c) Quick sort
(d) Insertion sort

Q.26 The adjacency matrix A of the graph G is



G

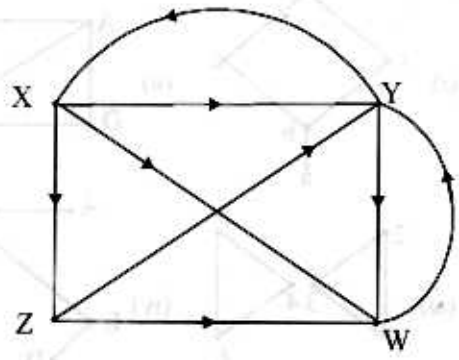
(a) $\begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix}$

(b) $\begin{pmatrix} 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix}$

(c) $\begin{pmatrix} 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix}$

(d) $\begin{pmatrix} 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix}$

Q.27 Consider the graph G in following figure.



Suppose **the nodes** are stored in memory in an array DATA as follows

DATA: x, y, z, w

Find the adjacency matrix A of the graph G.

(a) $A = \begin{pmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{pmatrix}$

(b) $A = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 \end{pmatrix}$

(c) $A = \begin{pmatrix} 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix}$

(d) $A = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 \end{pmatrix}$

Q.28 Let G be a simple graph with 20 vertices and 100 edges. The size of the minimum vertex cover of G is 8. Then the size of the maximum independent set of G is

(a) 12

(b) 8

(c) Less than 8

(d) More than 12

Q.29 In general, in bubble sort, after iteration i, the element will be placed at its proper position.

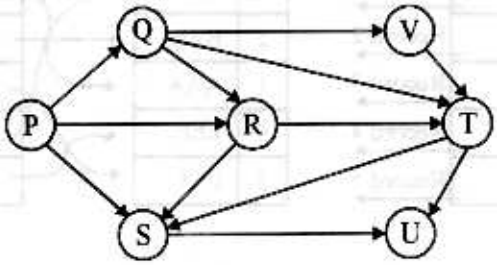
(a) x [n+ i]

(b) x [n–1]

(c)x [n –i]

(d) x [n–i–1]

Q.30 Which of the following is the correct decomposition of the directed graph given below into its strongly connected components?



(b) {P,Q,R,S},{T}, {U}, {V}

(b) {P,Q,R,S,T,V}, {U}

(c) {P,Q,S,T,V}, {R}, {U}

(d) {P,Q,R,S,T,U,V}

LEVEL–2

Q.31 For string operation, DELETE operation is defined is follows. DELETE (text, position, length)

By assuming text = 'GENIUS' , position = 0 and length = 2, we get output as

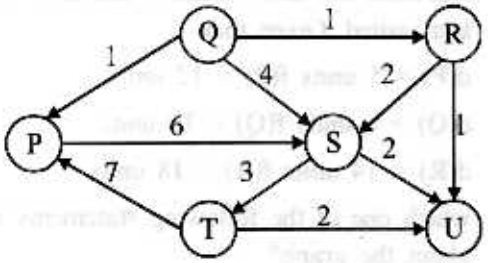
(a) GENI

(b) GE

(c) GENIUS

(d) None of the above

Q.32 Suppose we run Dijkstra single source shortest path algo on the following edge weighted directed graph with vertex P as the source.



In what order do the nodes get included into the set of vertices for which the shortest path distances are finalized?

(a) P, Q, R, S, T, U

(b) P, Q, R, U, S, T

(c) P, Q, R, U, T, S

(d) P, Q, T, R, U, S

Q.33 The time complexity of the following C function is (assume n > 0)

```
int rec (int n)
{
    if (n==1)
    return 1;
    else
    return n+rec (n–1),
}
```

(a) O(n)

(b) O(nlogn)

(c) $O(n^2)$

(d) $O(2^n)$

Q.34 Let G be a directed graph whose vertex set is the set of numbers from 1 to 100. There is an edge from a vertex i to a vertex j iff either j = i + 1 or j = 3i. The minimum number of edges in a path in G from vertex 1 to vertex 100 is

(a) 4

(b) 7

(c) 23

(d) 99

Q.35 Consider the depth-first-search of an undirected graph with 3 vertices P, Q and R. Let discovery time d(u) represent the time instant when the vertex u is first visited and finish time f(u) represent the time instant when the vertex u is last visited. Given that

d(P) = 5 units f(P) = 12 units

d(Q) = 6 units f(Q) = 10 units

d(R) = 14 units f(R) = 18 units

which one of the following statements is TRUE about the graph?

(a) There is only one connected component

(b) There are two connected components, and P and R are connected

(c) There are two connected components, and Q and R are connected

(d) There are two connected components, and P and Q are connected

Q.36 Consider a hash table of size 11 that uses open addressing with linear probing.

Let h(k) = k mod 11 be the hash function used. A sequence of records with keys

43 36 92 87 11 47 11 13 14

is inserted into an initially empty hash table, the bins of which are indexed from zero to ten.

What is the index of the bin into which the last record is inserted ?

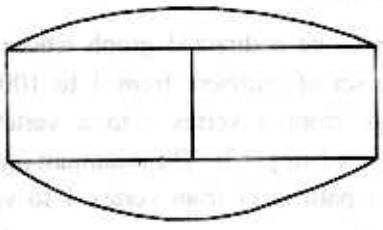
(a) 3

(b) 4

(c) 6

(d) 7

Q.37 The minimum number of colors required to color the following graph is:



(a) 1

(b) 2

(c) 3

(d) 4

Q.38 Let G be a weighted undirected graph and e be an edge with maximum weight in G. Suppose there is a minimum weight spanning tree in G containing the edge e. Which of the following statements is always TRUE?

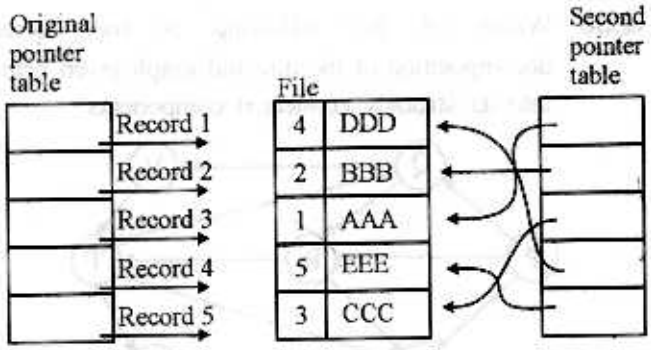
(a) There exists a cutset in G having all edges of maximum weight.

(b) There exists a cycle in G having all edges of maximum weight.

(c) Edge e cannot be contained in a cycle.

(d) All edges in G have the same weight.

Q.39 Consider the following diagram and select the correct statement.



(a) The above technique can be used for sorting by an address

(b) The above technique can be used for sorting by stable storage

(c) The above technique can be used for sorting by a value

(d) None of these

Q.40 If 25 is the number of filled elements in the hashed list and 100 is the total number of elements that can be allocated to the hashed list, then what will be load factor?

(a) 0.25%

(b) 0.33%

(c) 0.75%

(d) 0.667%

Q.41 Suppose Text T = 'ABCDEFG' and P = 'CD'. Then the value of INDEX (T,P), LENGTH (P) and DELETE ('ABCDEFG', 3,2) will be

(a) 4, 2, 'ABCD'

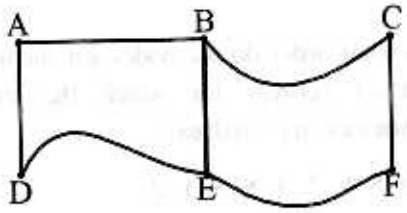
(b) 3, 2, 'ABEFG'

(c) 4, 2 'ABCDE'

(d) 3, 23 'EFG'

Common Data For Questions 42 & 43:

Consider the following connected graph G and answer the following question:



Q.42 The number of simple paths from node A to node F are–

(a) 5

(b) 6

(c) 7

(d) 4

Q.43 The distance between node A and F is --

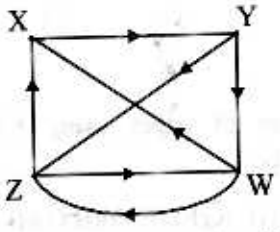
(a) 4

(b) 3

(c) 5

(d) not possible to find

Q.44 In the following graph which of the following statement is correct?



(a) X is a source and W is sink node

(b) Z is a source and there is no sink

(c) Z is a source and X is sink node

(d) none of these

Q.45 For certain string processing operations, we have to develop an INSERT operation by using SUBSTRING and LENGTH operations. Which of the following is correct way to do so?

(a) INSERT(T,K,S) = SUBSTRING(T,1,K–1) || S || SUBSTRING(T,K,LENGTH(T)–K)

(b) INSERT(T,K,S) = SUBSTRING(T,1,K–1) || SUBSTRING(T,K,LENGTH (T)–K)

(c) INSERT(T,K,S) = SUBSTRING(T,1,K–1) || S || SUBSTRING(T,K,LENGTH (T)–K+1)

(d) INSERT(T,K,S) = SUBSTRING (T,1,K–1) || S || SUBSTRING(T,K,LENGTH (T)–K–1)

where T is text in which we want to insert a string S beginning in position K, and || is the concatenation operation.

Q.46 Exponentiation is a heavily used operation in public key cryptography. Which of the following options is the tightest upper bound on the number of multiplications required to

compute b^n modulo m, $0 < b, n < m$?

(a) O(logn)

(b) $O\left(\sqrt{n}\right)$

(c) O(n/logn)

(d) O(n)

Q.47 Consider a hash function that distributes keys uniformly. The hash table size is 20. After hashing of how many keys will the probability that any new key hashed collides with an existing one exceed 0.5.

(a) 5

(b) 6

(c) 7

(d) 10

Common Data For Questions 48 & 49:

An array X of n distinct integers is interpreted as a complete binary tree. The index of the first element of the array is 0.

Q.48 The index of the parent of element X[i], $i \neq 0$ is

(a) $\lceil i/2 \rceil$

(b) $\lceil (i-1)/2 \rceil$

(c) $\lfloor i/2 \rfloor$

(d) $\lfloor (i/2) - 1 \rfloor$

Q.49 If the root node is at level 0, the level of element X [i], $i \neq 0$ is

(a) $\lceil \log_2 i \rceil$

(b) $\lfloor \log_2 (i+i) \rfloor$

(c) $\lceil \log_2 (i+1) \rceil$

(d) $\lfloor \log_2 i \rfloor$

Q.50 If the following degree vertex, (5, 3, 2, 4, 3) can't form a simple graph then which of the fallowing degree vertex should be added to make it form a simple graph?

(a) 1

(b) 5

(c) 2

(d) None of the above

Q.51 Consider a weighted undirected graph with positive edge weights and let uv be an edge in the graph. It is known that the shortest path from the source vertex s to u has weight 53 and the shortest path from s to v has weight 65. Which one of the following statements is always true?

(a) weight (u,v)< 12

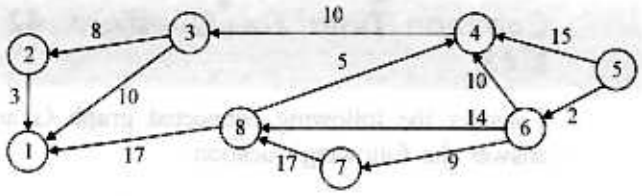
(b) weight (u,v) $\leq$ 12

(c) weight (u, v) > 12

(d) weight (u, v) $\geq$ 12

LEVEL–3

Q.52 If we run Dijkstra's single source shortest path algorithm for the following edge weighted directed graph with vertex 5 as the source.



In what order do the nodes get included into the set of vertices for which the shortest path distances are finalized.

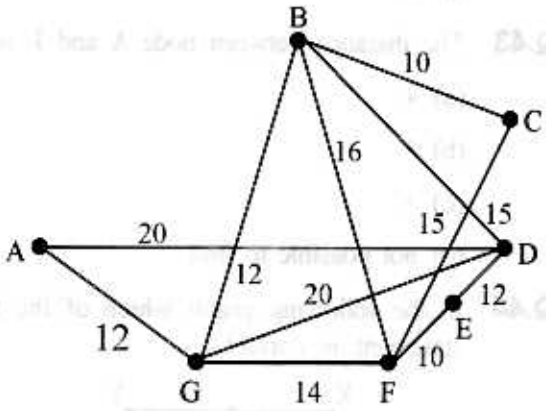
(a) 5, 6, 7, 4, 8, 3, 1, 2

(b) 5, 6, 7, 8, 4, 3, 2, 1

(c) 5, 6, 7, 4, 8, 2, 3, 1

(d) 5, 6, 7, 4, 8, 2, 1, 3

Q.53 Consider the following weighted graph



Selection of edges using Kruskals algorithm would be

(a) (A,G), (G,F),(E,D)(B,G)(B,C)

(b) (E,F), (B,C),(B,G)(A,G)(D,E)

(c) (B,C), (E,F),(A,G)(B,G)(F,G)

(d) None of the above

Q.54 Consider following unsorted sequence

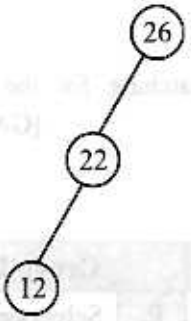
26 33 35 29 19 12 22

If at any stage of Heap sort the list is

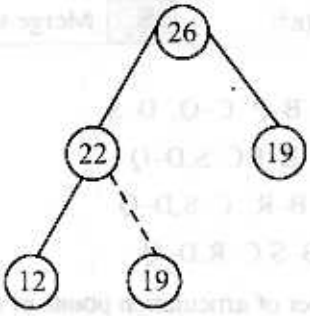
26	22	19	12		29	33	35
1	2	3	4		5	6	7

, the subtree would be

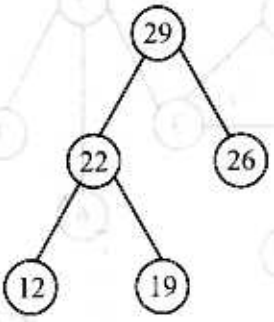
(a)



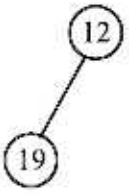
(b)



(c)



(d)



Q.55 For the table of 12 items given below, we use Radix sort to sort them.

19, 13, 05, 27, 01, 26, 31, 16, 02, 09, 11, 21

What will be the output, when items of buckets are merged after the first distribution?

(a) 19, 13, 02, 09, 01, 21, 31, 16, 05, 24, 11, 26

(b) 01, 02, 05, 09, 11, 13, 16, 19, 21, 26, 24, 31

(c) 09, 01, 02, 19, 11, 05, 24, 13, 21, 31, 16, 26

(d) 01, 31, 11, 21, 02, 13, 05, 26, 16, 27, 19, 09

Q.56 For the sequence

500, 535, 512, 721, 436, 611, 624, 632, 643

Lexicographic sort gives time complexity of:

(a) O (39)

(b) O (29)

(c) O (28)

(d) O (27)

GATE QUESTIONS

Q.57 The complexity of comparison based sorting algorithms is : **[GATE 1990]**

(a) O (n log n)

(b) θ (n)

(c) θ (2n)

(d) θ (n^2)

Q.58 Consider a simple connected graph G with n vertices and n-edges (n>2). Then, which of the following statements are true? **[GATE 1993]**

(a) G has no cycles

(b) The graph obtained by removing any edge from G is not connected

(c) G has at least one cycle

(d) The graph obtained by removing any two edges from G is not connected

(e) None of the above

Q.59 Linked lists are not suitable data structures of which one of the followings problems ? **[GATE 1994]**

(a) Insertion sort
(b) Binary search
(c) Radix sort
(d) Selection sort

Q.60 The number of distinct simple graphs with upto three nodes is **[GATE 1994]**

(a) 15
(b) 10
(c) 7
(d) 9

Q.61 The minimum number of edges in a connected cyclic graph on n vertices is **[GATE 1995]**

(a) n–1
(b) n
(c) n+1
(d) None of the above

Q.62 For merging two sorted lists of sizes m and n into a sorted list of size m+n, no. of required comparisons are: **[GATE 1995]**

(a) O (m)
(b) O (n)
(c) O (m+n)
(d) O (log m + log n)

Q.63 Merge sort uses **[GATE 1995]**

(a) Divide and conquer strategy
(b) Backtracking approach
(c) Heuristic search
(d) Greedy approach

Q.64 An advantage of chained hash table (external hashing) over the open addressing scheme is **[GATE 1996]**

(a) Worst case complexity of search operations is less
(b) Space used is less
(c) Deletion is easier
(d) None of the above

Q.65 Let G be a graph with 100 vertices numbered 1 to 100. Two vertices i and j are adjacent if $|i-j| = 8$ or $|i-j|=12$. The number of connected components in G is **[GATE 1997]**

[2-Marks]

(a) 8
(b) 12
(c) 4
(d) 25

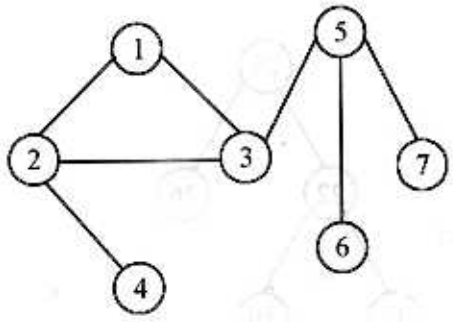
Q.66 Give the correct matching for the followings pairs: **[GATE 1998]**

[1-Mark]

Group I		Group II	
A.	O (log n)	P.	Selection
B.	O (n)	Q.	Insertion sort
C.	O (n log n)	R.	Binary search
D.	O (n^2)	S.	Merge sort

(a) A–R, B–P, C–Q, D–S
(b) A–R, B–P,C–S,D–Q
(c) A–P, B–R, C–S,D–Q
(d) A–P,B–S,C–R,D–Q

Q.67 The number of articulation points of the following graph is



[GATE 1999]

[1-Mark]

(a) 0
(b) 1
(c) 2
(d) 3

Q.68 If one uses straight two-way merge sort algorithm to sort the following elements in ascending order: 20,47,15,8,9,4,40,30,12,17 then the order of these elements after second pass of the algorithm is :

[GATE 1999]

[1-Mark]

(a) 8,9,15,20,47,4,12,17,30,40

(b) 8,15,20,47,4,9,30,40,12,17

(c) 15,20,47,4,8,9,12,30,40,17

(d) 4,8,9,15,20,47,12,17,30,40

Q.69 Let G be an undirected graph. Consider a depth first traversal of G, and let T be the resulting depth first search tree. Let u be a vertex in G and let v be the first new (unvisited) vertex visited after visiting u in the traversal. Which of the following statements is always true?

[GATE 2000]

[2-Marks]

(a) {u,v} must be an edge in G, and u is a descendant of v in T.

(b) {u,v} must be an edge in G, and v is a descendant of u in T.

(c) If {u,v} is not an edge in G then u is a leaf in T.

(d) If {u,v} is not an edge in G then u and V must have the same parent in T.

Q.70 Let G be an undirected connected graph with distinct edge weight. Let e_{max} be the edge with maximum weight and e_{min} the edge with minimum weight. Which of the following statements is false? [GATE 2000]

[2-Marks]

(a) Every minimum spanning tree of G must contain e_{min}

(b) If e_{max} is in a minimum spanning tree, then its removal must disconnect G.

(c) No minimum spanning tree contains e_{max}

(d) G has a unique minimum spanning tree

Q.71 Consider an undirected unweighted graph G. Let a breadth first traversal of G be done starting from a node r. Let d (r,u) and d (r,v) be the lengths of the shortest paths from r to u and v respectively in G. If u is visited before v during the breadth first traversal which of the following statements is correct ? [GATE 2001]

[2-Marks]

(a) d (r,u)<d(r,v)

(b) d (r,u)>d (r,v)

(c) d (r,u) ≤d(r,v)

(d) None of the above

Q.72 Randomized quicksort is an extension of quicksort where the pivot is chosen randomly. What is the worst case complexity of sorting n number using randomized quick sort?

[GATE2001]

[1-Mark]

(a) O (n)

(b) O (n log n)

(c) $O(n^2)$

(d) O (n!)

Q.73 The usual $\Theta(n^2)$ implementation of insertion sort to sort an array uses linear search to identify the position where an element is to be inserted into the already sorted part of the array. If, instead, we use binary search to identify the position, the worst case running time will [GATE 2003]

[1-Mark]

(a) remain $\Theta(n^2)$

(b) become Θ $(n(\log n)^2)$

(c) become Θ (n log n)

(d) become Θ (n)

Q.74 The cube root of a natural number n is defined as the largest natural number m such that $m^3 \le n$. The complexity of computing the cube root of n (n is represented in binary notation) is

[GATE 2003]

[2-Marks]

(a) O(n) but not $O(n^{0.5})$

(b) $O(n^{0.5})$ but not $O((\log n)^k)$ for any constant k>0.

(c) $O((\log n)^k)$ for some constant k>0, but not $O((\log \log n)^m)$ for any constant m>0

(d) $O((\log \log n)^k)$ for some constant k > 0.5, but not $O((\log \log n)^{0.5})$

Q.75 The following are the starting and ending times of activities A,B,C,D,E,F,G and H respectively in chronological order; "a_s b_s c_s a_e d_s c_e e_s f_s b_e d_e g_s e_e f_e h_s g_e h_e". Here ,x_s denotes the starting time and x_e denotes the ending time of activity X. We need to schedule the activities in a set of rooms available to us. An activity can be scheduled in a room only if the room is reserved for the activity for its entire duration. What is the minimum number of rooms required?

[GATE 2003]

[2-Marks]

(a) 3

(b) 4

(c) 5

(d) 6

Q.76 Let G = (V,E) be an undirected graph with a subgraph $G_1 = (V_1, E_1)$. Weights are assigned to edges of G as follows

$$w(e) = \begin{cases} 0 & \text{if } e \in E_1 \\ 1 & \text{otherwise} \end{cases}$$

A single source sorted path algorithm is executed on the weighed graph (V, E, w) with an arbitrary vertex V_1 of V as the source. Which of the following can always be inferred from the path costs computed? **[GATE 2003]**

[2-Marks]

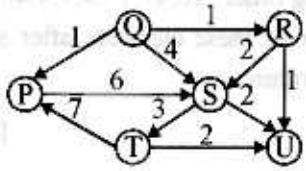
(a) The number of edges in the shortest paths from V_1 to all vertices of G.

(b) G_1 is connected

(c) V_1 forms a clique in G

(d) G_1 is a tree.

Q.77 Suppose we run Dijikstra single source shortest path algo on the following edge weighted directed graph with vertex P as the source.



In what order do the nodes get included into the set of vertices for which the shortest path distances are finalized? **[GATE 2004]**

[2-Marks]

(a) P, Q, R, S, T, U

(b) P, Q, R, U, S, T

(c) P, Q, R, U, T, S

(d) P, Q, T, R, U, S

Q.78 Given the following input (4322, 1334, 1471, 9679, 1989, 6171, 6173, 4199) and the hash function x mod 10, which of the folllowing statements are true ? **[GATE 2004]**

[1-Mark]

1. 9679, 1989, 4199 hash to the same value
2. 1471, 6171 hash to the same value
3. All elements hash to the same value
4. Each element hash to a different value

(a) 1only

(b) 2 only

(c) 1 and 2 only

(d) 3 and 4 only

Q.79 Let P be a singly-linked list. Let Q be the pointer to an intermediate node x in the list. What is the worst-case time complexity of the best known algorithm to delete the node x from the list?

[IT-GATE 2004]

[1 Mark]

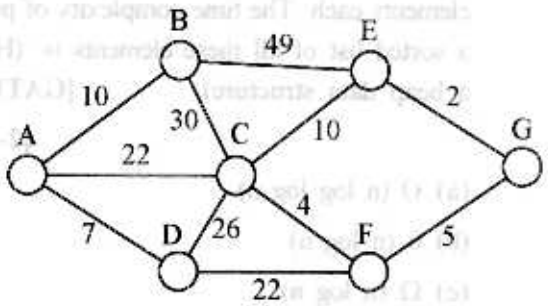
(a) O(n)

(b) $O(\log_2 n)$

(c) O(log n)

(d) O(1)

Q.80 Consider the undirected graph below:



Using Prim's algorithm to construct a minimum spanning tree starting with node A, which one of the following sequences of edges represents a possible order in which the edges would be added to construct the minimum spanning tree?

[IT-GATE 2004]

[2-Marks]

(a) (E, G), (C, F), (F, G), (A, D), (A, B), (A, C)

(b) (A, D), (A, B), (A, C), (C, F), (G, E), (F, G)

(c) (A, B), (A, D), (D, F), (F, G), (G, E), (F, C)

(d) (A, D), (A, B), (D, F), (F, C), (F, G), (G, E)

Q.81 Consider a list of recursive algorithms and a list of recurrence relations as shown below. Each recurrence relation corresponds to exactly one algorithm and is used to derive the time complexity of the algorithm.

Recursive Algorithm		Recurrence Relation	
P	Binary search	I	$T(n) = T(n-k) + T(k) + cn$
Q	Merge sort	II	$T(n) = 2T(n-1) + 1$
R	Quick sort	III	$T(n) = 2T(n/2) + cn$
S	Tower of Hanoi	IV	$T(n) = T(n/2) + 1$

Which of the following is the correct match between the algorithm and their recurrence relations?

[IT-GATE 2004]

[2-Marks]

(a) P – II, Q – III, R – IV, S – I

(b) P – IV, Q – III, R – I, S – II

(c) P – III, Q – II, R – IV, S – I

(d) P – IV, Q – II, R – I, S – III

Q.82 A hash table contains 10 buckets and uses linear probing to resolve collisions. The key values are integers and the hash function used is key % 10. If the values 43, 165, 62, 123, 142 are inserted in the table, in what location would the key value 142 be inserted?

[IT-GATE 2005]

[1 Mark]

(a) 2

(b) 3

(c) 4

(d) 6

Q.83 Let T(n) be a function defined by the recurrence

$$T(n) = 2T\left(\frac{n}{2}\right) + \sqrt{n} \text{ for } n \geq 2, \text{ and}$$

$$T(1) = 1$$

Which of the following statements is TRUE?

[IT-GATE 2005]

[2-Marks]

(a) $T(n) = \theta(\log n)$

(b) $T(n) = \theta(\sqrt{n})$

(c) $T(n) = \theta(n)$

(d) $T(n) = \theta(n \log n)$

Q.84 Let G be a directed graph whose vertex set is the set of numbers from 1 to 100. There is an edge from a vertex i to a vertex j iff either $j = i + 1$ or $j = 3i$. The minimum number of edges in a path in G from vertex 1 to vertex 100 is:

[IT-GATE 2005]

[2-Marks]

(a) 4

(b) 7

(c) 23

(d) 99

Statement for Linked Answer Questions 85(i) and 85(ii):

A sink in a directed graph is a vertex i such that there is an edge from every vertex $j \neq i$ to i and there is no edge from i to any other vertex. A directed graph G with n vertices is represented by its adjacency matrix a, where A[i][j] = 1 if

there is an edge directed from vertex i to j and 0 otherwise. The following algorithm determines whether there is a sink in the graph G.

```
i = 0;
do{
    j=i+1;
    while((j<n) && E1)j++;
    if (j<n) E2;
} while (j<n)
flag = 1;
for(j=0;j<n;j++)
if (j!=i)&&E3) flag = 0;
if(flag) printf("Sink exists") else printf("Sink does not exist");
```

Q.85 (i) Choose the correct expressions for E_1 and E_2 **[IT-GATE 2005]** **[2-Marks]**

(a) E_1 : A[i][j] and E_2 : i = j;

(b) E_1 : !A[i][j] and E_2 : i = j + 1;

(c) E_1 : !A[i][j] and E_2 : i = j;

(d) E_1 : A[i][j] and E_2 : i = j + 1;

Q.85 (ii) Choose the correct expression for E_3 **[IT-GATE 2005]** **[2-Marks]**

(a) (A[i][j] && !A[j][i])

(b) (!A[i][j] && !A[j][i])

(c) (!A[i][j] || !A[j][i])

(d) (A[i][j] || !A[j][i])

Q.86 Suppose T (n) = 2T (n/2) + n,

T (0) = T (1) = 1

Which one of the following is FALSE? **[GATE 2005]** **[2-Marks]**

(a) $T(n) = O(n^2)$

(b) $T(n) = \theta(n \log n)$

(c) $T(n) = \Omega(n^2)$

(d) $T(n) = O(n \log n)$

Q.87 Suppose there are $\lceil \log n \rceil$ sorted lists of $\lfloor n/\log n \rfloor$ elements each. The time complexity of producing a sorted list of all these elements is: (Hint: Use a heap data structure) **[GATE 2005]** **[2-Marks]**

(a) O (n log log n)

(b) θ (n log n)

(c) Ω (n log n)

(d) Ω $(n^{3/2})$

Q.88 The time complexity of computing the transitive closure of a binary relation on a set of n elements is known to be **[GATE 2005]** **[1-Mark]**

(a) O (n)

(b) O (n log n)

(c) O $(n^{3/2})$

(d) O (n^3)

Common Data For Questions 89 and 90:

Consider the following C - function:

```
double foo (int n){
    int i;
double sum;
if (n = = 0) return 1. 0;
    else{
    sum = 0.0;
    for (i = 0; i < n; i ++)
    sum + = foo (i);
    return sum;
    }}
```

Q.89 The space complexity of the above function is **[GATE 2005]** **[2-Marks]**

(a) O(1)

(b) O(n)

(c) O(n!)

(d) $O(n^n)$

Q.90 Suppose we modify the above function foo() and store the values of foo(i), $0 <= i <n$, as and when they are computed. With this modification, the time complexity for function foo() is significantly reduced. The space complexity of the modified function would be: **[GATE 2005]**

[2-Marks]

(a) O(1)

(b) O(n)

(c) $O(n^2)$

(d) O(n!)

Q.91 Let G (V, E) be an undirected graph with positive edge weights. Dijkstra's single source shortest path algorithm can be implemented using the binary heap data structure with time complexity: **[GATE 2005]**

[2-Marks]

(a) O $(V|^2)$

(b) O (| E | + | V | log | V |)

(c) O(| V | log | V |)

(d) O ((| E | + | V |) log | V |)

Common Data For Questions 92 & 93:

A sink in a directed graph is a vertex i such that there is an edge from every vertex $j \neq i$ and there is no edge from i to any other vertex. A directed graph G with n vertices is represented by its adjacency matrix A, where A [i][j] = 1 if there is an edge directed from vertex i to j and 0 otherwise. The following algorithm determines whether there is a sink in the graph G.

```
i = 0
do {
    j =i+1
    while ((j<n)&&E1)j++;
    if (j<n) E2
} while(j<n)
flag = 1;
for (j=0;j<n;j++)
if (j!=i)&&E3)flag =0;
if (flag) printf ("Sink exists") else printf("Sink does not exist");
```

Q.92 Choose the correct expression for E_1 and E_2. **[IT-GATE 2005]**

[2-Marks]

(a) E_1: A[i][j] and E_2: i = j

(b) E_1: !A[i][j] and E_2: i = j+1;

(c) E_1: !A[i][j] and E_2: i = j;

(d) E_1: A[i][j] and E_2: i = j+1;

Q.93 Choose the correct expression for E_2. **[IT-GATE 2005]**

[2-Marks]

(a) (A[i][j] &&!A[j][i])

(b) (!A[i][j]&&A[j][i])

(c) (!A[i][j] ||A [j][i])

(d) (A[i][j] || !A[j][i])

Common Data For Questions 94 & 95:

Let s and t be two vertices in an undirected graph G = (V, E) having distinct positive edge weights. Let [X, Y] be a partition of V such that $s \in X$ and $t \in Y$. Consider the edge e having the minimum weight amongst all those edges that have one vertex in X and one vertex in Y.

Q.94 The edge e must definitely belong to: **[GATE 2005]**

[2-Marks]

(a) the minimum weighted spanning tree of G

(b) a weighted shortest path from s to t

(c) each path from s to t

(d) the weighted longest path from s to t

Q.95 Let the weight of an edge e denote the congestion on that edge. The congestion on a path is defined to be the maximum of the congestions on the edges of the path. We wish to find the path from s to t having minimum congestion. Which one of the following path is always such a path of minimum congestion? **[GATE 2005]**

[2-Marks]

(a) a path from s to t in the minimum weighted spanning tree

(b) a weighted shortest path from s to t

(c) an Euler walk from s to t

(d) a Hamiltonian path from s to t

Q.96 Consider a weighted complete graph G on the vertex set $\{v_1, v_2, \ldots, v_n\}$ such that the weight of the edge (v_i, v_j) is $2 \mid i - j \mid$. The weight of a minimum spanning tree of G is **[GATE 2006]**

[1-Mark]

(a) $n - 1$

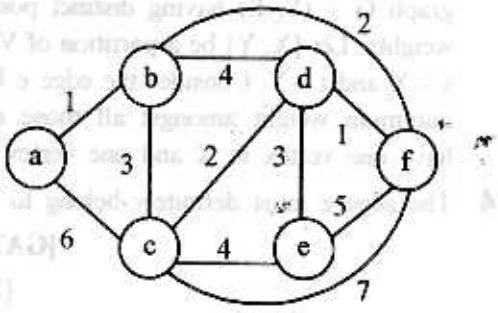
(b) $2n - 2$

(c) $\left(\frac{n}{2}\right)$

(d) n^2

Q.97 Consider the following graph. Which one of the following cannot be the sequence of edges added, in that order, to a minimum spanning tree using Kruskal's algorithm ? **[GATE 2006]**

[2-Marks]



(a) (a - b), (d - f), (b -f), (d - c), (d - e)

(b) (a - b), (d - f), (d -c), (b - f), (d - e)

(c) (d - f), (a - b), (d - c), (b - f), (d - e)

(d) (d - f), (a - b), (b - f), (d - e), (d - c)

Q.98 The median of n elements can be found in O(n) time. Which one of the following is correct about the complexity of quick sort, in which median is selected as pivot ? **[GATE 2006]**

[2-Marks]

(a) $\theta(n)$

(b) $\theta(n \log n)$

(c) $\theta(n^2)$

(d) $\theta(n^3)$

Q.99 Consider the process of inserting an element into a Max Heap, where the Max Heap is represented by an array. Suppose we perform a binary search on the path from the new leaf to the root to find the position for the newly inserted element, the number of comparisons performed is **[GATE 2007]**

[2-Marks]

(a) $\theta(\log_2 n)$

(b) $\theta(\log_2 \log_2 n)$

(c) θn

(d) $\theta(n \log_2 n)$

Q.100 Consider the following C program segment where CellNode represents a node in a binary tree:

```
struct CellNode {
    struct CellNode *leftchild;
        int element;
    struct CellNode *rightChild;
};
int GetValue (struct CellNode  *ptr)     {
    int value  = 0;
    if  (ptr! = NULL)                    {
        if  ((ptr->leftchild == NULL) &&
                (ptr->rightchild == NULL)
        value = 1;
    else
        value = value + GetValue(ptr ->leftchild)
        + GetValue (ptr -> rightchild);
    }
return (value);
}
```

The value returned by GetValue when a pointer to the root of a binary tree is passed as its argument is: **[GATE 2007]**

[2-Marks]

(a) the number of nodes in the tree

(b) the number of internal nodes in the tree

(c) the number of leaf nodes in the tree

(d) the height of the tree

Q.101 Which of the following sorting algorithms has the lowest worst-case complexity? **[GATE 2007]**

[1-Mark]

(a) Merge sort

(b) Bubble sort

(c) Quick sort

(d) Selection sort

Q.102 Consider a hash table of size seven, with starting index zero, and a hash function (3x + 4) mod 7. Assuming the hash table is initially empty, which of the following is the contents of the table when sequence 1, 3, 8, 10 is inserted into the table using closed hashing? Note that - denotes an empty location in the table. **[GATE 2007]**

[2-Marks]

(a) 8, -, -, -, -, -, 10

(b) 1, 8, 10, -, -, - 3

(c) 1,-,-,-,-,-, 3

(d) 1,10,8,-,-,-, 3

Q.103 What is the time complexity of the following recursive function?

```
int DoSomething (int n)  {
if(n<=2)
    return 1;
else
    return (DoSomething (floor(sqrt (n))) + n); }
```

[GATE 2007]

[2-Marks]

(a) $\theta(n^2)$

(b) $\theta(n \log n)$

(c) $\theta(\log n)$

(d) $\theta(\log \log n)$

Q.104 In an unweighted, undirected connected graph, the shortest path from a node S to every other node is computed most efficiently, in terms of time complexity, by **[GATE 2007]**

[2-Marks]

(a) Dijkstra's algorithm starting from S.

(b) Warshall's algorithm

(c) performing a DFS starting from S

(d) performing a BFS starting form S

Q.105 Let w be the minimum weight among all edge weights in an undirected connected graph. Let e be a specific edge of weight w. Which of the following is FALSE?

[GATE 2007]

[2-Marks]

(a) There is a minimum spanning tree containing e.

(b) If e is not in a minimum spanning tree T, then in the cycle formed by adding e to T, all edges have the same weight.

(c) Every minimum spanning tree has an edge of weight w.

(d) e is present in every minimum spanning tree.

Q.106 The most efficient algorithm for finding the number of connected components in an undirected graph of n vertices and m edges has time complexity

[GATE 2008]

[1-Mark]

(a) θ (n)

(b) θ (m)

(c) θ (m + n)

(d) θ (mn)

Q.107 Consider the following functions:

$f(n) = 2^n$

$g(n) = n!$

$h(n) = n^{\log n}$

which of the following statements about the asymptotic behaviour of f(n), g(n), and h(n) is true?

[GATE 2008]

[2-Marks]

(a) $f(n) = O(g(n)); g(n) = O(h(n))$

(b) $f(n) = \Omega(g(n)); g(n) = O(h(n))$

(c) $g(n) = O(f(n)); h(n) = O(f(n))$

(d) $h(n) = O(f(n)); g(n) = \Omega(f(n))$

Q.108 Consider the Quicksort algorithm. Suppose there is a procedure for finding a pivot element which splits the list into two sub-lists each of which contains at least one-fifth of the elements. Let T(n) be the number of comparisons required to sort n elements Then **[GATE 2008]**

[2-Marks]

(a) $T(n) \leq 2T(n/5) + n$

(b) $T(n) \leq T(n/5) + T(4n/5) + n$

(c) $T(n) \leq 2T(4n/5) + n$

(d) $T(n) \leq 2T(n/2) + n$

Q.109 The subset-sum problem is defined as follows : Given a set S of n positive integers and a positive integer W, determine whether there is a subset of S whose elements sum to W. An algorithm Q solves this problem in O(nW) time. Which of the following statements is false ? **[GATE 2008]**

[2-Marks]

(a) Q solves the subset-sum problem in polynomial time when the input is encoded in unary

(b) Q solves the subset-sum problem in polynomial time when the input is encoded in binary

(c) The subset sum problem belongs to the class NP

(d) The subset sum problem is NP-hard

Q.110 What is the chromatic number of an n-vertex simple connected graph which does not contain any odd length cycle ? Assume $n > 2$.

[GATE 2009]

[1-Mark]

(a) 2

(b) 3

(c) n-1

(d) n

Q.111 What is the number of swaps required to sort n elements using selection sort, in the worst case?

[GATE 2009]

[1-Mark]

(a) θ (n)

(b) θ (nlog n)

(c) θ (n^2)

(d) θ (n^2log n)

Q.112 In quick sort, for sorting n elements, the $(n/4)^{th}$ smallest element is selected as pivot using an O(n) time algorithm. What is the worst case time complexity of the quick sort ? **[GATE 2009]**

[2-Marks]

(a) θ (n)

(b) θ (nlog n)

(c) θ (n^2)

(d) θ (n^2log n)

Q.113 Which of the following statement(s) is/are correct regarding Bellmanford shortest path algorithm?

P. Always finds a negative weighted cycle, if one exists.

Q. Finds whether any negative weighed cycle is reachable from the source

[GATE 2009]

[1-Mark]

(a) P only

(b) Q only

(c) both P and Q

(d) neither P nor Q

Q.114 The keys 12,18,13,2,3,23,5 and 15 are inserted into an initially empty hash table of length10 using open addressing with hash function h(k) = k mod 10 and linear probing. What is the resultant hash table ? **[GATE 2009]**

[2-Marks]

(a)

0	
1	
2	2
3	23
4	
5	15
6	
7	
8	18
9	

(b)

0	
1	
2	12
3	13
4	
5	5
6	
7	
8	18
9	

(c)

0	
1	
2	12
3	13
4	2
5	3
6	23
7	5
8	18
9	15

(d)

0	
1	
2	12,2
3	13,3,23
4	
5	5,15
6	
7	
8	18
9	

Q.115 The running time of an algorithm is represented by the following recurrence relation.

$$T(n) = \begin{cases} n & n \le 3 \\ T\left(\frac{n}{3}\right) + cn & \text{otherwise} \end{cases}$$

Which one of the following represents the time complexity of the algorithm? **[GATE 2009]**

[2-Marks]

(a) θ (n)

(b) θ (n log n)

(c) θ (n^2)

(d) θ (n^2 log n)

ANSWER KEY

1	c	2	c	3	c	4	b	5	d
6	c	7	b	8	b	9	c	10	b
11	d	12	b	13	b	14	c	15	b
16	a	17	b	18	a	19	a, c	20	b
21	a	22	a	23	b	24	c	25	b
26	a	27	a	28	a	29	c	30	c
31	d	32	b	33	a	34	b	35	d
36	d	37	c	38	c	39	a	40	a
41	b	42	d	43	b	44	d	45	c
46	a	47	c	48	b	49	c	50	a
51	b	52	b	53	c	54	b	55	d
56	d	57	a	58	c, d	59	b	60	c, d
61	b	62	c	63	a	64	c	65	c
66	b	67	d	68	b	69	b	70	c
71	c	72	c	73	a	74	c	75	b
76	b	77	a	78	c	79	a	80	a
81	b	82	d	83	d	84	c	85(i),(ii)	a, c
86	b	87	a	88	d	89	c	90	b
91	b	92	a	93	c	94	a	95	b
96	b	97	d	98	b	99	a	100	c
101	a	102	b	103	d	104	d	105	d
106	c	107	d	108	b	109	b	110	a
111	a	112	b	113	c	114	c	115	a

SOLUTIONS

S.1 (c)

$n^{1/3} < n^{7/4} < n \log n < (1.0000001)^n < e^n$.

$\because \quad n < n \log n < e^n$.

S.2 (c)

The longest path that is Diameter of the graph is n (think about a chain in which every node is different in one bit). For such a graph the chromatic number is always 2. We can color them alternately using two colors.

S.3 (c)

Maximum number of edges

$$n_{c_2} = \frac{n!}{(n-2)!2!} = \frac{n(n-1)}{2}$$

S.4 (b)

As they are unlabeled graphs only **two** distinct graphs are possible.

S.5 (d)

The time complexity for selection sort to sort an array of n elements is $O(n^2)$.

S.6 (c)

Bubble sort requires $(n(n-1)/2)$ comparisons.

S.8 (b)

In bubble sort, with n elements and k iterations the total number of comparisons is

$(n-1) + (n-2) + (n-3) + ..+ (n-k)$ which equals

$$nk - \frac{k(k-1)}{2} = \frac{2nk - k^2 + k}{2}$$

$$= \frac{(2kn - k^2 + k)}{2}$$

S.9 (c)

Graph C is unilaterally connected.

There is no cycle from C to any other node, so G is not strongly connected. However, G is unilaterally connected. Also, index (A) = 2, so it is not a source. Node C is sink, since indeg(C) = 2 but outdeg (C) =0. No node in G is a source.

S.11 (d)

If we have simply connected graph with n vertices and n edges, it contains atleast n different spanning trees.

e.g If n =3 △

(i) > (ii) < (iii) ∧

If n =4

□ (i) ⊓ (ii) ⊐ (iii) ⊏

(iv) ⊓

S.12 (b)

A **grid** connects all the vertices like Hamiltonian cycle but has minimum total weight unlike Hamiltonian cycle.

S.13 (b)

All comparision based sorting has tightest lower bound as n^2.

S.14 (c)

A sorting technique is called stable if for all records i and j such that k[i] equals k[j], if r[i] precedes r[j] in the original file, r[i] precedes r[j] in the sorted file.

S.15 (b)

A Hamiltonian path is a path that contains each vertex exactly once.

S.16 (a)

Quick sort shows best average as it uses partitioning concept.

S.17 (b)

The worst and average case complexity of heap sort is **O(n log n)**.

S.18 (a)

Worst case complexity of insertion sort $O(n^2)$.

S.19 (a, c)

(a), (c) are properties of $O(N^2)$ algorithm.

S.20 (b)

Stable sorting means relative order is same.

S.21 (a)

Edges are subsets equal to V × V.

S.22 (a)

A node u is called source if it has **positive outdegree and zero indegree** in a graph G.

S.26 (a)

The nodes are normally ordered according to the way they appear in memory, i.e.we assume

$v_1 = X$, $v_2 = Y$, $v_3 = Z$, $v_4 = W$

The adjacency matrix A to G follows:

$$\begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix}$$

Here $a_{ij} = 1$ if there is a node from v_i to v_j otherwise $a_{ij} = 0$.

S.27 (a)

The nodes are normally ordered according to the way the appear in memory i.e. we assume

$v_1 = X$, $v_2 = Y$, $v_3 = Z$ and $v_4 = W$

S.28 (a)

Maximum vertex cover = 8

Total vertex = 20

Then the size of the maximum independent set of G is = 20 – 8 = **12**

S.29 (c)

Let n = 10, i = 3 (number of iteration) and elements are

X[10] = 12, 5, 3, 8, 7, 2, 9, 15, 11, 13

After first iteration of bubble sort

= 2, 12, 5, 8, 7, 3, 9, 15, 11, 13

Second iteration

= 2, 3, 12, 8, 7, 5, 9, 15, 11, 13

Third iteration

= 2, 3, 5, 12, 8, 7, 9, 15, 11, 13
(2, 3, 5: sorted)

∵ after i = 3, three elements are sorted, so element will be placed at its proper position is x[i] or x[n – i].

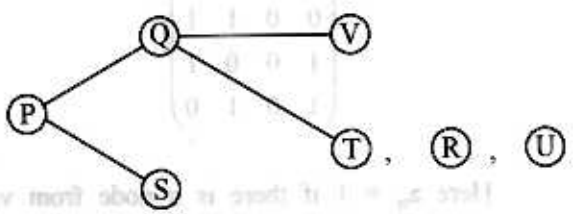
x[i] if we sort from left to right or x[n – i] if we sort from right to left.

S.30 (c)

To make connected components of the directed graph, we decompose the graph such that there is no cycle in connected components.

So best decomposition satisfying this properly is:

{{P, Q, S, T, V}, {R}, {U}}



S.31 (d)

DELETE ('GENIUS',0,2) = NIUS

Text will delete of length 2 from the position 0.

Text → G E N I U S

Position → 0 1 2 3 4 5

(0 1: will delete)

Nothing is deleted if position = 0

S.32 (b)

Start from node P, explore its branches, then select minimum cost branch, apply this technique to all nodes including one by one.

Thus we get the node sequence

P, Q, R, U, S, T

S.33 (a)

The time complexity of the C-function could be shown as:

$$T_{rec(n)} = \begin{cases} 2 & h = 1 \\ T_{rec(n-1)} + 2 & h > 1 \end{cases}$$

Now for h > 1

$T_{rec(n)} = T_{rec(n-1)} + 2$

$= T_{rec}(n-2) + 2 + 2$

$= T_{rec}(n-3) + 2 + 2 + 3$

$= T_{rec}(n-(n-1)) + 2 + 2 + \ldots + (n-1)\text{time}$

$= T_{rec}(1) + 2(n-1)$

$= 2 + 2n - 2$

$= 2n$

$= \mathbf{O(n)}$

S.34 (b)

j = i+1 or j = 3i

for i = 1 to 100

we need

100→99→33→11→10

↓

1←3←9

⇒ **7 paths**

S.35 (d)

Here d(Q) > d(P)

∴ P's immediate neighbour is Q. After P is finished, Q is visited immediately and then R is discovered, which clearly shows that **P and Q are connected.**

S.36 (d)

Given sequence is

43 36 92 87 11 47 11 13 14

h(K) = K mod 11

∴ 43 mod 11 → 10

36 mod 11 → 3

92 mod 11 → 4

87 mod 11 → 10 → 0 [∵ 10 is occupied]

11 mod 11 → 0 → 1 [∵ 0 is occupied]

47 mod 11 → 3 → 2 [∵ 3 & 4 is occupied]

11 mod 11 → 0 → 5 [∵ 0 is occupied]

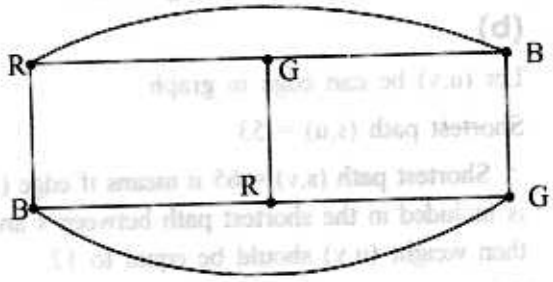
13 mod 11 → 2 → 6 [∵ 2 is occupied]

14 mod 11 → 3 → 7 [∵ 3 is occupied]

0	1	2	3	4	5	6	7	8	9	10
87	11	47	36	92	11	13	14			43

S.37 (c)

Let's say we have **3 color** namely R, G and B.



S.38 (c)

An edge with the maximum weight is always excluded from the minimum weight.

⇒ **e can't be contained in the cycle.**

S.39 (a)

In the given diagram, an auxiliary table of pointers may be used so that these pointers are moved instead of the actual data. It is also called as **sorting by an address.**

The pointers are given for the proper data. So the data is sorted using pointers called sorting by address.

S.40 (a)

$$\text{Load factor} = \frac{k}{n}\%$$

where, k is the number of filled elements in the list and n is the total number of elements

$$\text{Load factor} = \frac{25}{100}\% = \frac{1}{4}\%$$

$$= \mathbf{0.25\%}$$

S.41 (b)

T = 'ABCDEFG', P = 'CD', then INDEX (T,P) = 3 {pattern matching} LENGTH (P) = 2 DELETE ('ABCDEFG',3,2) = **ABEFG,** since we delete CD i.e. 3rd and 4th character from the string.

S.42 (d)

A simple path from A to F is a path such that no node and hence no edge is repeated.

There are four such simple paths:

(A,B,C,F), (A,B,E,F), (A,D,E,F), (A,D,E,B,C,F)

S.43 (b)

The distance from A to F equals 3, since there is a simple path (A,B,C,F) from A to F to length 3 and there is no shorter path from A to F (when no weight is given consider all are of length 1).

S.44 (d)

No node is a sink and no node is a source. As all the edges from any nodes are not either entering or leaving it.

S.45 (c)

INSERT (T,K,S) = SUBSTRING (T, 1 K–1) ||S|| SUBSTRING (T,KLENGTH (T) – K+1)

The initial substring of T before the position K, which has length K–1, is concatenated with the string S and the result is concatenated with the substring of T which has length LENGTH (T) – (K–1) = LENGTH (T) – K + 1.

S.47 (c)

Hash table size = 20

value after hasing = 0,1,2,....9

After inserting another key (2^{nd} key) probability of colliding = 1/10 = .1

⇒ After inserting a new key, the probability of colliding increase by 0.1.

⇒ After inserting a new 3^{rd} key, the probability of colliding 0.1+0.1 =0.2

⇒ After inserting a new 4^{th} key, the probability of colliding = 0.3

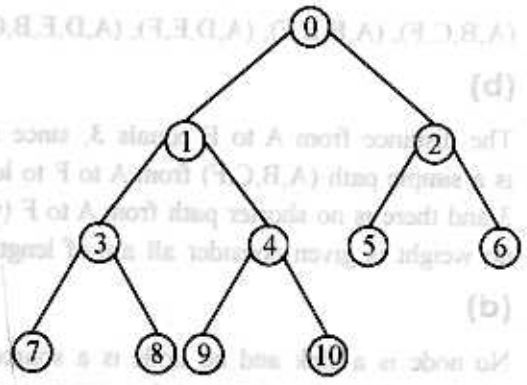
⇒ After inserting a new 5^{th}key, the probability of colliding = 0.4

⇒ After inserting a new 6^{th}key, the probability of colliding = 0.5

⇒ After inserting a new 7^{th}key, the probability of colliding = 0.6 > 0.5

⇒ **7 keys** will be the required key

S.48 (b)



For any elements x[i], the index of its parent = either i/2 or i/2 –1 then $\left\lceil \frac{i-1}{2} \right\rceil$.

S.49 (c)

at level 0, number of elements = 2^0=1

at level 1, number of elements = 2^1=2

at level 2, number of elements = 2^2=4

:

:

at level k, number of elements = 2^k

Let the element x[i] occur at level k

Maximum element upto level k–1 = $2^0+2^1+.....2^{k-1}= 2^k-1$

$\Rightarrow 2^k-1 \le i$

$\Rightarrow 2^k \le i + 1$

$\Rightarrow k \le \log_2 (i+1)$

$\Rightarrow k = \lceil \log_2 (i+1) \rceil$

S.50 (a)

First sort the degree vertex which gives us (5, 4, 3, 3, 2). Let label these vertices as (a, b, c d, e) respectively and the new vertex (to be added) as 'f'. The vertex 'a' (degree 5) should be adjacent to 5 vertices. This means 'f' vertex should have a degree of at least 1. The second vertex should be adjacent to another 3 vertices as vertex 'a' is already adjacent to this contributing 1 degree. We will take these 3 vertices as (c, d, e). The vertex 'c' is already adjacent to this contributing 1 degree. We will take these 3 vertices adjacent to this. This we can take as 'd'. With this all degree are satisfied for all vertices. So the new vertex 'f' should have a **degree of 1**.

S.51 (b)

Let (u,v) be can edge in graph

Shortest path (s,u) = 53

∴ Shortest path (s,v) = 65 it means if edge (u,v) is included in the shortest path between s and v, then weight (u,v) should be equal to 12.

If it is not included it means weight (u,v) ≤12.

S.52 (b)

Start from node (5) and apply Dijkstra's algorithm, explore all branches of node (5) and select minimum of them and add the node of minimum edge cost.

Similarly, explore all branches of added nodes to add a new node in the group.

After this process we get the sequence of node of the given is **5, 6, 7, 8, 3, 4, 2, 1.**

S.53 (c)

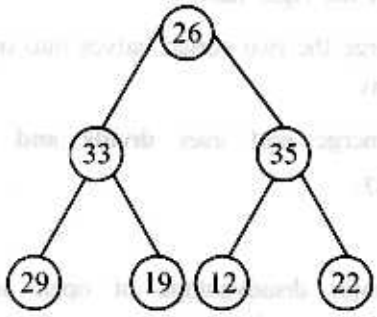
Kruskal's Algorithm:

Step 1 : Choose an edge e_1 in S of least weight. Let E = $\{e_1\}$. Replace S with S–$\{e_1\}$.

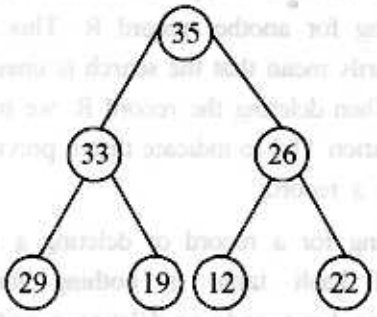
Step 2 : Select an edge e_1 in S of least weight that will not make a cycle with members of E. Replace E with E∪$\{e_1\}$ and S with S–$\{e_1\}$.

Step 3: Repeat step 2 until |E| = n–1.

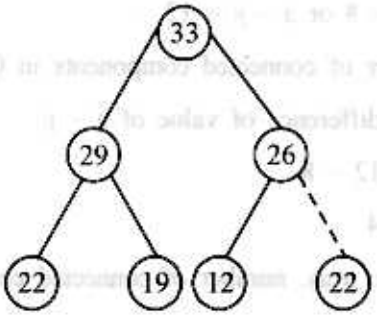
S.54 (b)



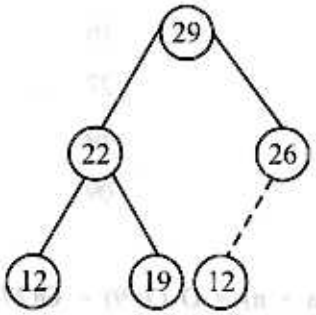
Step (i)



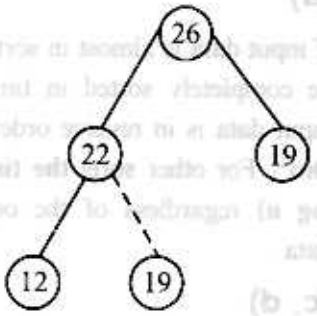
Step (ii)



Step (iii)



Step (iv)



Step (v)

as children of a [1]

its children a[2] and a[3] = 22 and 19.

S.55 (d)

Radix sort is also called bucket sort. The sort involves examining the least significant digit of the keyword and assigned the item to a bucket uniquely dependent on the value that of digit. After all items have been distributed the "buckets" items are merged in order and the process is repeated with the next significant digit until no more digits are left.

First Distribution	Merge
0)	01
1) 01,31,11,21	31
2) 02	11
3) 13	21
4)	02
5) 05	13
6) 26,16	05

7) 27 26

8) 16

9) 19, 09 27

19

09

S.56 (d)

$T(n) = O(m \times n) = O(3 \times 9) = \mathbf{O(27)}$

where m: the number of digits on each element of sequence

n: number of elements in sequence

S.57 (a)

If input data is almost in sorted order, then it can be completely sorted in time O(n), whereas if input data is in reverse order, then it needs time $O(n^2)$. For other sorts **the time required is O(n log n)** regardless of the original order of the data.

S.58 (c, d)

As simple connected graph G has n vertices and n edges, so it must have atleast one cycle. Now as one edge is removed from G, it still can remain connected graph. But if two edges are removed it will never remain connected.

S.59 (b)

In binary search tree implemented using linked list, one can easily insert and delete elements by changing certain pointers. The main drawback of the binary search tree is that the tree may be very unbalanced, so that the length of a path may be O(n) rather than O(log n). This will reduce the searching to approximately a linear search.

S.60 (c, d)

The no. of graphs having n-nodes is

$= 2^n - 1$ or $2^n + 1$

= 7, 9 for n = 3 nodes.

S.61 (b)

For any connected cyclic graph with n vertices one must have n edges if no loops are present.

Thus minimum **n edges** will be there.

S.62 (c)

Suppose input consists of the total number X = m + n of elements in A and B. Each comparison assigns an element to the array C, which eventually has X elements. Thus number of comparisons cannot exceed X.

$\therefore$ No of comparison $\leq X = O(X) = \mathbf{O(m + n)}$

S.63 (a)

The merge sort algorithm performs:

(a) Cut the array in half.

(b) Sort the left half

(c) Sort the right half.

(d) Merge the two sorted halves into one sorted array.

Thus merge sort uses **divide and conquer strategy.**

S.64 (c)

One major disadvantage of open addressing procedure is in the implementation of **deletion.** Suppose a record R is deleted from the location T[r]. Afterwards, suppose we meet T[r] while searching for another record R. This does not necessarily mean that the search is unsuccessful. Thus when deleting the record R, we must label the location T[r] to indicate that it previously did contain a record.

Searching for a record or deleting a record in chained hash table is nothing more than searching for a node or deleting a node from a linked list.

S.65 (c)

$|i - j| = 8$ or $|i - j| = 12$

Number of connected components in G

= difference of value of $|i - j|$

= 12 – 8

= 4

In other way, number of connected components in

$G = 100 - 12 \times 8 = 4$

S.66 (b)

Number of comparisons required for the best case in

(i) Selection sort is O(n)

(ii) Insertion sort is O (n^2)

(iii) Binary search is O (log n)

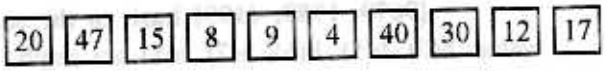
(iv) Merge sort is O (n log n).

S.67 (d)

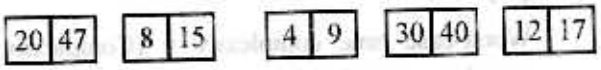
There are **three** articulation points in the graph namely 2,3 and 5.

S.68 (b)

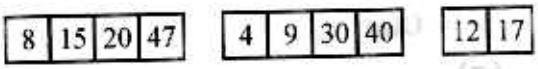
Splitting will result into



First pass would result into



Secondpass would result into



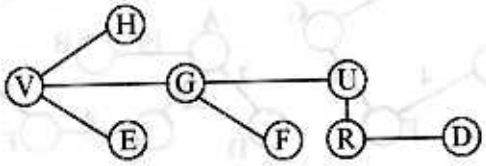
∴ order of these elements after second pass would be **8, 15, 20, 47, 4, 9, 30, 40, 12, 17.**

S.70 (c)

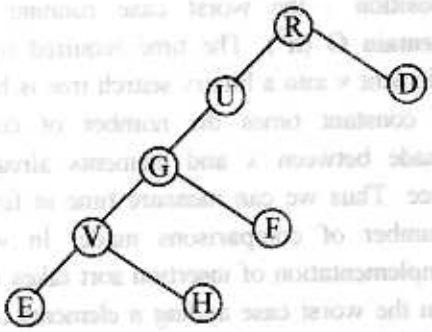
For any undirected connected graph G with e_{max} and e_{min} be the maximum and minimum weight of the edges, every minimum spanning tree must contain e_{min}. It may happen that e_{max} be the only weighted edge joining two vertices.

S.71 (c)

Consider the following graph



Breadth First traversal would give the following tree if starting from node R

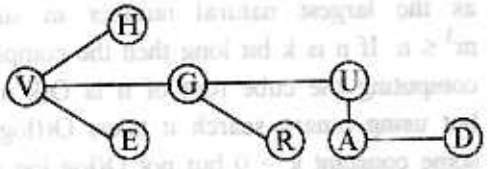


For the above graph

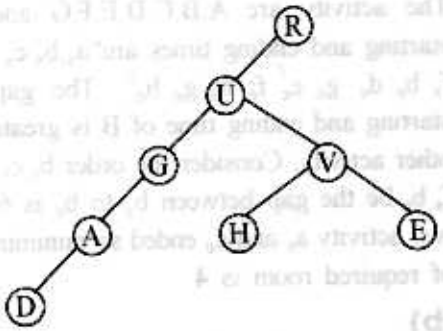
d (r,u) < d (r, v) .....(i)

because u and v are not at the same level.

Again consider following graph



Breadth first traversal would give the following tree if starting from node R.



For the above graph

d (r,u) = d (r,v) .....(ii)

Combining (i) and (ii) will give

d (r,u) d ≤ (r,v)

S.72 (c)

Worst case complexity of sorting n numbers using quicksort is $\mathbf{O(n^2)}$. Same is with randomized quicksort.

S.73 (a)

The usual O (n^2) implementation of insertion sort to sort an array uses binary search to identify the position , the worst case running time will **remain O (n^2)**. The time required to insert an element x into a binary search tree is bounded by a constant times the number of comparisons made between x and elements already in the tree. Thus we can measure time in terms of the number of comparisons made. In worst case implementation of insertion sort takes O (n^2). So in the worst case adding n elements to a binary search tree could require $O(n^2)$ time. In expected case means elements are sorted in increasing order adding n elements to a binary search tree could require O (n log n) time.

S.74 (c)

The cube root of a natural number n is defined as the largest natural number m such that $m^3 \le n$. If n is k bit long then the complexity of computing the cube root of n is $O(k^3n^2\log(B))$ but using binary search it takes $O((\log n)^k)$ for some constant $k > 0$ but not $O(\log \log n)^m)$ for nay constant $m > 0$ because it require more multiplication then addition.

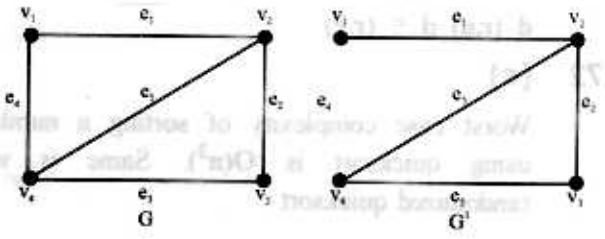
S.75 (b)

The activity are A,B,C,D,E,F,G and H. The starting and ending times are"a_s,b_s c_s a_e d_s c_e e_s f_s b_e d_e g_s c_e f_e h_s g_e h_e". The gap between starting and ending time of B is greater than all other activity. Consider the order b_s c_s a_e d_s c_e e_s f_s b_e be the gap between b_s to b_e is 6 in which two activity a_e and c_e ended so minimum number of required room is 4.

S.76 (b)

Let = (V,E) be an undirected graph $G_1 = (V_1, E_1)$ such that $E_1 \subseteq E$ and $V_1 \subseteq V$

Consider as example:



Consider the w(e) = 1 if $e \in E_1$ it means the cost of V_1 to V_4 is only 1 other edges having cost 0. It is notated that **G_1 is connected** V_1 does not form any clique in G as well as G_1 is not a tree.

S.78 (c)

Input (4322, 1334, 1471, 9679, 1989, 6171, 6173, 4199)

Given Hash function,

h(a) = x mod 10

h(a) = h(1)

= **{1471, 6171} hash to the same vlaue**

h(a) = h(2) = {4322}

h(a) = h(3) = {6173}

h(a) = h(9)

= **{9679, 1989, 4199} hash to the same** value: h(0), h(5), h(6), h(7), h(8) are empty for given input.

S.79 (a)

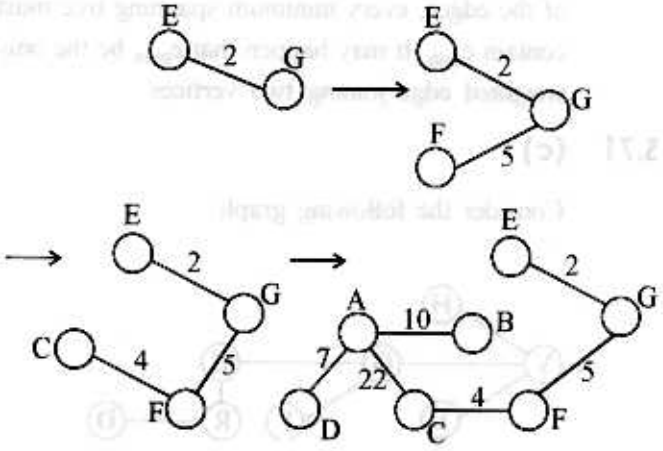
Worst-case time complexity = (Complexity of searching (x)) * {delete(x)}

= O(n) * O(1)

= **O(n)**

S.80 (a)

In Prim's algorithm, the edge with smallest weight added first, so sequence of addition of edge in spanning tree so that cycle does not exist, will be-



Hence sequence = **(E, G), (C, F), (F, G), (A, D), (A, D), (A, B), (A, C)**

S.81 (b)

P. Binary search – $T(n) = T(n/2) + 1$

Q. Merge sort – $T(n) = 2T(n/2) + Cn$

R. Quick sort – $T(n) = T(n - K) + T(K) + Cn$

S. Tower of Hanoi – $T(n) = 2T(n - 1) + 1$

S.82 (d)

hash table

0	
1	
2	62
3	43
4	123
5	165
6	142
7	
8	
9	

Apply hash function %10 on inserted values

When 43 inserted then 43%10 = 3

It will insert at 3rd position.

When 165 inserted then 165%10 = 5

It will insert at 5th position.

When 62 inserted then 62%10 = 2

It will insert at 2nd position.

When 123 inserted then 123%10 = 3

It will insert at 3rd position and collision will occur. So it will insert at next vacant position i.e. 4.

When 142 inserted then 142%10 = 2

It will insert at 2nd position and collision will occur. So it will insert at next vacant position i.e. 6.

S.83 (d)

$$T(n) = 2T(n/2) + \sqrt{n} \quad \forall\, n \geq 2 \;\; T(1) = 1$$

$$= 2\left[2T(n/4) + \sqrt{n/2}\right] + \sqrt{n}$$

$$= 4T(n/4) + 2\sqrt{n/2} + \sqrt{n}$$

$$= 4\left[2T(n/8) + \sqrt{n/4}\right] + 2\sqrt{n/4} + \sqrt{n}$$

$$= 8T(n/8) + 4\sqrt{n/4} + 2\sqrt{n/2} + \sqrt{n}$$

$$\Rightarrow \quad T(n) = n.T(n/n) + \ldots 8T(n/8) + 4\sqrt{n/4} + 2\sqrt{n/2} + \sqrt{n}$$

put n = 8

$$T(8) = 8.T(1) + 4\sqrt{8/4} + 2\sqrt{8/2} + \sqrt{8}$$

$$= 8 + 4\sqrt{2} + 4 + 2\sqrt{2}$$

$$= 2 + 6\sqrt{2}$$

That is equivalent to

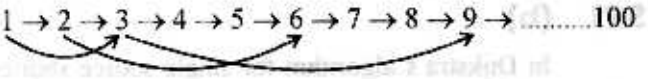
$$T(8) = \theta(8\log 8)$$

So **$T(n) = \theta(n\log n)$**

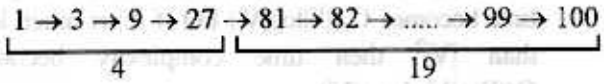
S.84 (c)

There is an edge from a vertex i to a vertex j iff either j = i + 1 or j = 3i

So possible set of edge is



So minimum number of edges in a path or from vertex 1 to vertex 100 is



So total = 4 + 19 = 23

S.86 (b)

The given recurrence relation can be solved using Master Theorem. It lies in case 2 of Master Theorem or if we remember recurrence relation of merge sort or best case quick sort, we can guess the value of T(n).

$T(n) = \theta(n \log n)$

By definition of Big O notation, we can say

$\theta(n \log n) = O(n \log n) = O(n^2)$

$\therefore$ $\theta(n \log n)$ can be equal to $\Omega(n)$ or $\Omega(n \log n)$, but not $\Omega(n^2)$.

S.87 (a)

The time complexity of heap sort datastructure is O(n log (n)). As, here we have $\lceil \log n \rceil$ sorted list of $\lfloor n/\log n \rfloor$ elements each. So, its time complexity will become **$O(n \log \log n)$.**

S.88 (d)

The time complexity of computing the transitive closure of a binary relation on a set of n elements is $O(n^3)$ apply the Warshall's algorithm to compute the transitive closure. The algorithm contains three nested for loops each having frequency n, so time complexity is **$O(n^3)$**.

S.89 (c)

The given C function is recursive and the recurrence equation is

$$T(0) = 1$$

$$T(n) = T(n) + S \text{ where S is a constant}$$

The function is recursive and the size of longest double is $1 \cdot 2 \cdot 3 \cdot \cdot n = n!$ **So the space complexeity is O(n!).**

S.90 (b)

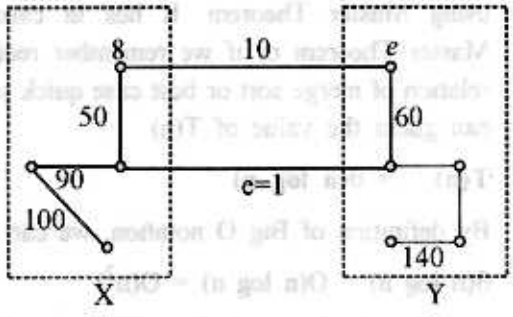
If the value of foo(i) such that $0 < i < n$ then the longest size can be store in n bit string so the space complexity becomes **O(n)**.

S.91 (b)

In Dijkstra's algorithm for single source shortest path when we implement the algorithm using the binary heap data structure then total updates of E edges require O (logv) time and total spend time becomes O (|E|log|V|) but if |E| is much less than $|V|^2$ then time complexity becomes **O(|E|+|V| log |V|)**.

S.94 (a)

Consider an example:



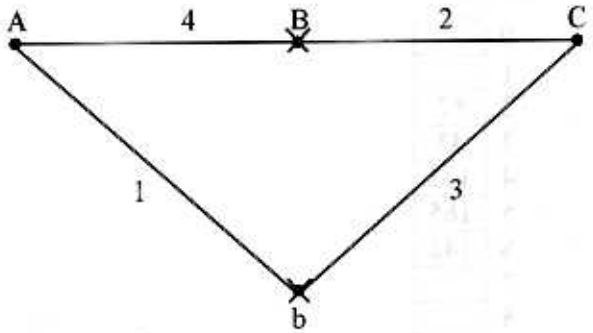
Here, minimum cost of edge joining X and Y is one (1). We called it as e.

Here, actually two edge that join X and Y one cost '10'

Here, if we consider weighted shortest path between s and i, we find it as an edge of cost 10.

We can't consider e edge here while its cost is minimum.

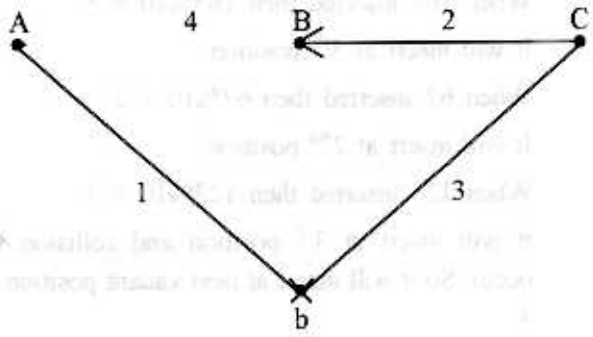
S.95 (b)



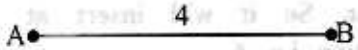
Consider here a path with minimum congestion between A and B.

We follow any minimum spanning algorithm

Here, Resultant tree is



While, a minimum path between A and B is 4.



S.96 (b)

There are n vertices hence n edges are there. for the minimum spanning tree, we need n–1 edges

The weight of an edge is 2 |i–j|

Hence the total weight of n–1 edges = Σ 2|i–j|

$\Rightarrow$ weight of minimum spanning tree

= Min Σ 2 |i–j|

the min value of $|i-j| = 1$

Hence min $\Sigma 2|i-j| = 2$ (1+1+1....n–1 term)

$= 2n-2$

S.97 (d)

Ordering the weights, the sequence obtained is 1,1,2,2,3,3,4,4,5,5,6,6,7,7 now, in order to obtain the minimum spanning tree using Kruskal's Algorithm we have to add edge with weights in increasing order such that the weight of the spanning tree is minimum. In (d) the weights of the edges are 1,1,2,3,2 which contradicts Krushkal's Algorithm.

S.98 (b)

After the median is found out in O(n) time, each reduction step divides the list into 2 sublists so that at each of the (log n) reduction steps n comparisons will be needed at most.

$\therefore$ Time taken $= O(n) + O(\log n)$

$= O(n \log n) = \theta(n \log n)$.

S.99 (a)

Inserting an element into a max heap takes O(1) time. When we perform a binary search on the path from the new leaf to the root to find the position for the newly inserted element takes $\theta(\log n)$ time. **So, total time = $\theta(\log n)$.**

S.100 (c)

This program contain the code for counting the **number of leaf nodes in the tree.**

S.101 (a)

Worst case complexity of bubble sort, quick sort and selection sort are same i.e., $O(n^2)$ while **merge sort** is O(n log n).

S.102 (b)

Simulate the hashing of the inputs

Input = 1, hash value = (3*1+4)mod 7 =0

Hash table index	Hash table entry
0	1
1	
2	
3	
4	
5	
6	

Input = 3, hash value = (3*3+4)mod 7=6

Hash table index	Hash table entry
0	1
1	
2	
3	
4	
5	
6	3

Input = 8, hash value = (3*8+4) mod 7 =0

Hash table index	Hash table entry
0	1
1	8
2	
3	
4	
5	
6	3

Input =10, hash value = (3*10+4)mod 7 =6

Hash table index	Hash table entry
0	1
1	8
2	10
3	
4	
5	
6	3

S.103 (d)

As per the given recursive function, the recurrence relation for the time complexity T(n) is given below:

$T(0) = T(1) = T(2) = 0$

$T(n) = 1 + T(\sqrt{n})$

Take n as n^k where k is 2^r

$T(n) = 1 + T\left(\left(\sqrt{n}\right)\right)$

$T(\sqrt{n}) = 1\ T\left(\sqrt{\sqrt{n}}\right)$

$$T\left(\sqrt{\sqrt{n}}\right) = 1 + T\left(\sqrt{\sqrt{\sqrt{n}}}\right)$$

$$T(1) = 0$$

So 1 is to be added r times, which logarithemic complexeity i.e., θ(log logn).

S.104 (d)

In an unweighted graph **performing a BFS starting from S takes O(n) time** if graph contains n vertices. Dijkstra's algorithm only find the single source shortest path and takes $O(n^2)$ time but it is good for weighted graph. Warshall's algorithm for all pair shortest path takes $O(n^3)$ time but it is also applicable for weighted graph.

S.105 (d)

w be the minimum weight among all edge weights in an undirected connected graph. e is the specific edge of weight w. It may be possible that another edge in the graph is present having weight w which had been added to minimum spanning tree and when we add to minimum spanning tree it forms a simple circuit. So **we can't include e in every minimum spanning tree.**

S.106 (c)

The most efficient algorithm for finding the number of connected components in an undirected graph on n-vertices and m edges using depth first search takes **O(m+n)** time. Let $n \le m$.

Or

Connected components can be found in **O(m+n)** using Tarjan's algorithm. Once we have connected components, we can count them.

S.107 (d)

$$f(n) = 2^n$$

$$\Rightarrow \quad f(n) = O(2^n)$$

$$g(n) = n!$$

$$\Rightarrow \quad g(n) = O(n!)$$

$$h(n) = n^{\log n}$$

$$\Rightarrow \quad h(n) = O(n^{\log n})$$

The asymmptotic order of function is as follows $1 < \log \log n < \log n < n^e < n^c\, n^{\log n} < c^n < n^n < c^{cn} < n!$

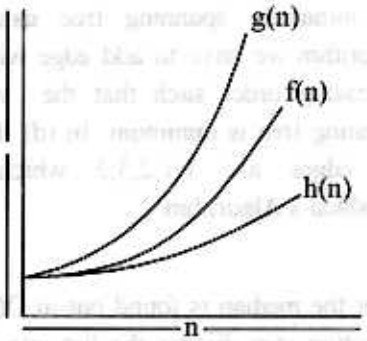
[$n < n^2$ means n grows more slowly than n^2]

So $\quad n^{\log n} < c^n < n!$

$\Rightarrow \quad n^{\log n} < 2^n < n!$

Assume $\quad C = 2$

$$h(n) < f(n) < g(n)$$



So, in worst case

$$h(n) \in O(f(n))$$

$$h(n) = O(f(n))$$

In best case when n is small then

$$h(n) \in \Omega(f(n))$$

$$h(n) = \Omega(f(n))$$

But in worst case

$$f(n) \subseteq O(g(n)) \text{ but } g(b) \not\subset O(h(n))$$

$$h(n) = n^{\log n}$$

$$f(n) = 2^n$$

$$n \log n \le C2^n$$

$$C = 1 \text{ and } n_0 = 2$$

$$2 \log 2 \le 2^2$$

So $\quad$ **h(n) = O(f(n)**

$$g(n) = n!$$

$$f(n) = 2^n$$

$$n! = C2^n \quad \text{for all } n \le n_0$$

$$C = 1 \text{ and } n = 4$$

$$4! = 24 \ge 2^4$$

$\Rightarrow \quad 24 \ge 16$

$\therefore \quad$ **g(n) = Ω (f(n))**

S.108 (b)

For the case where n/5 elements are in one subset, T(n/5) comparisons are needed for the first subset, while T(4n/5) is for the rest 4n/5 elements, and n is for finding the pivot.

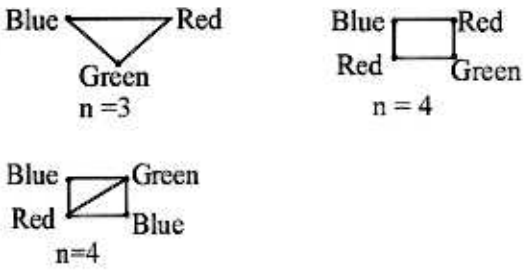
It there are more than n/5 elements in one set then other set will have less than 4n/5 elements and time complexity will be **less than T(n/5) + T(4n/5) + n** because recursion tree will be more balanced.

S.109 (b)

Given a set S of n positive integer and a positive integer W. And also an algorithm Q solves in O(nW) time.

S.110 (a)

Chromatic no. of n-vertex simple connected undirected graph which do not contain any odd length cycle will be $2(n \geq 2)$.



S.111 (a)

If there are n elements then in worst case, total swaps will be (n–1) in selection sort ∴ no of swaps = **θ(n).**

S.112 (b)

Avg. complexity of quick sort is **θ (n log n).**

Or

The recursion expression becomes

$T(n) = T(n/4) + T(3n/4) + Cn$

which results in **θ(n log n).**

S.113 (c)

Both P and Q are correct.

S.114 (c)

12 mod 10 = 2

18 mod 10 = 8

13 mod 10 = 3

2 mod 10 = 2 collision

(2+1) = 3 again collinsion (using linear probing)

(3+1) = 4 (using linear probing)

3 mod 10 = 3 collision

(3+1) = 4 collision (using linear probing)

(4+1) = 5 (using linear probing)

23 mod 10 = 3 collision

(3+1) = 4 collision (using linear probing)

(4+1) = 5 agian collision (using linear probing)

(5+1) = 6 (using linear probing)

5 mod 10 = 5 collision

(5+1) = 6 again collision (using linear probing)

(6+1) = 7 (using linear probing)

15 mod 10 = 5 collision

(5+1) = 6 collision(using linear probing)

(6+1) = 7 collision (using linear probing)

(7+1) = 8 collision (using linear probing)

(8+1) = 9 (using linear probing)

So resulting hash table.

0	
1	
2	12
3	13
4	2
5	3
6	23
7	5
8	18
9	15

S.115 (a)

Complexity is decided for large values of n only,

So, $T(n) = T(n/3) + cn$ for $n>3$

using master's theorem, case II

here $a = 1, b = 3, \log_b a = \log_3 1 = 0$

$f(n) = cn = \theta(n^1)$

since $n\log_b a = n^0$ is below $f(n) = \theta(n^1)$

this belongs to case III of master's theorem,

where the solution is $T(n) = \theta(f(n))$

$= \theta(n)$

5.4 STACK AND NP COMPLETE

LEVEL–1

Q.1 A problem in NP is NP complete if

(a) it can be reduced to the 3 SAT problem in polynomial time

(b) the 3–SAT problem can be reduced to it in polynomial time

(c) it can be reduced to any other problems NP in polynomial time

(d) some problem in NP can be reduced to it in polynomial time

Q.2 Which of the following applications may use a stack?

(a) A parentheses balancing program.

(b) Keeping track of local variables at run time.

(c) Syntax analyzer for a compiler.

(d) All of the above

Q.3 What is the maximum no. of parentheses that appears on the stack at any one time when the algorithm analyzes ((() (()) (()))? [HINT: consider the usual algo that determines balanced parenthesis]

(a) 1

(b) 2

(c) 3

(d) 4

Q.4 The operation

i = pop (s)

Push (s,i)

is equivalent to

(a) i = stacktop (s)

(b) empty (s)

(c) Remove (i)

(d) add (i)

Q.5 Evaluate the following postfix notation:

A:6,9,2,+,*,12,3,/,--

(a) 62

(b) 66

(c) 83

(d) None of these

Q.6 Choose the equivalent prefix form of the following expression [a+(b–c)]*[(d–e)/(f+g–h)]

(a) *+a–bc/–de–+fgh

(b) *+a–bc–/de–+fgh

(c) *+a–bc/–ed+–fgh

(d) *+a. b–c/–ed+–fgh

Q.7 Stacks are used in

(a) compilers in parsing an expression by recursion

(b) memory management in operating system

(c) both (a) and (b)

(d) none of the above

LEVEL–2

Common Data For Questions 8 & 9:

The subset-sum problem is defined as follows.

Given a set of n positive integers, $S = \{a_1, a_2, a_3, ..., a_n\}$ and positive integer W, is there a subset of S whose elements sum to W? A dynamic program for solving this problem uses a 2-dimensional Boolean array, X, with n rows and W + 1 columns,

X [i,j], $1 \le i \le n$, $0 \le j \le W$, is True if and only if there is a subset of $\{a_1, a_2, ... a_r\}$ whose element sum to j.

Q.8 If the following elements 5, 3, 4, 2, 6, 7 and 1, are inserted into a heap then what is printed for an in order traversal of the heap.

(a) 2, 5, 3, 7, 4, 6, 1

(b) 3, 5, 2, 7, 4, 6, 1

(c) 4, 6, 3, 7, 2, 5, 1

(d) 4, 6, 3, 7, 1, 5, 2

Q.9 Overflow–

(a) will not occur

(b) occurs when STACK contains 7 elements and there is a PUSH operation.

(c) occurs when STACK contain 9 elements and there is a PUSH operation.

(d) Data insufficient.

Q.10 If P is a pointer to a node, node (p) refers to the node pointed by p , info (p) refers to the information portion of that node, and next (p) refers to the next address portion. If next (p) is not null then info (next(p)) refers to–

(a) the information portion of the node that follows node (p) in the list.

(b) the address portion of the node that follows node (p) in the list.

(c) the information portion of the node that follows next (p) in the list

(d) the address portion of the node that follows next (p) in the list

Q.11 If each element on a stack, were a structure occupying ten words, the addition of an eleventh word to contain a pointer–.

(a) increases the space requirement by 20 percent

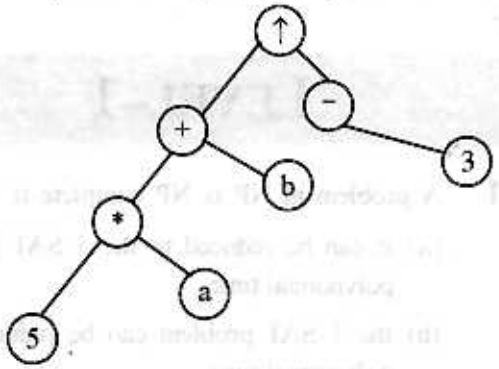
(b) increases the space requirement by 10 percent

(c) decreases the space requirement by 20 percent

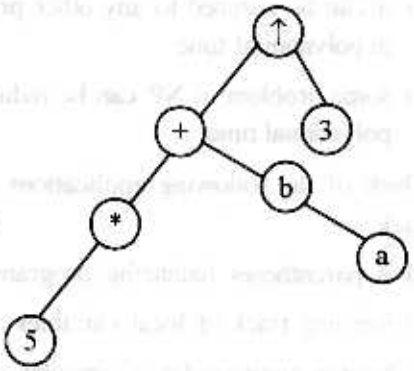
(d) decreases the space requirement by 10 percent.

Q.12 The scope of the exponential operation is –

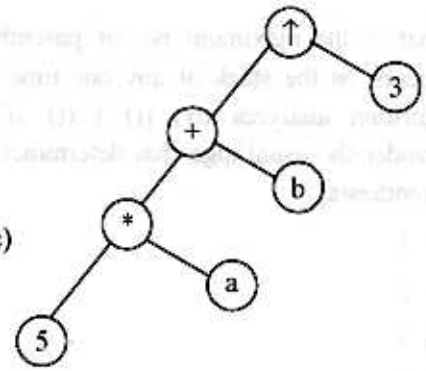
(a)



(b)



(c)



(d) Not possible

Q.13 If we find the prefix polish expression p which is equivalent to E and also find the preorder of the tree which is formed from the expression E then which of the following statement is correct?

Consider E = (2 * x + y)(5 * a – b) ↑ 3

(a) The prefix polish expression P and preorder of E is not equivalent.

(b) The prefix polish expression P and preorder to E is not equivalent to *+2* x y ↑–* 5 a 3 b

(c) The prefix polish expression P and preorder to E is not equivalent but is its postorder is equivalent.

(d) The prefix polish expression P and preorder of E is equivalent to * + * 2 x y ↑ – * 5 a b 3

Common Data For Questions 14 & 15:

Consider the following arithmetic expression P, written in postfix notation.

P: 12 , [7, 3], 2, 1, 5+, *+–/

Q.14 The equivalent infix expression is –

(a) 12/(7–3) + (2* 1) + 1

(b) 12/ (7–3) + 2 * (1+5)

(c) (12/7) –3 + (2*1) + 5

(d) None of these

Q.15 Evaluate the infix expression obtained in question 14:

(a) 10

(b) 15

(c) 5

(d) None of these

Q.16 The five items: U,V,W,X and Y are pushed onto a stack one after the other starting from U. The stack is popped four times and each element is inserted in a queue. Then two elements are deleted from the queue and pushed back on the stack and then one item is popped from the stack. Then the popped item is –

(a) U

(b) V

(c) W

(d) X

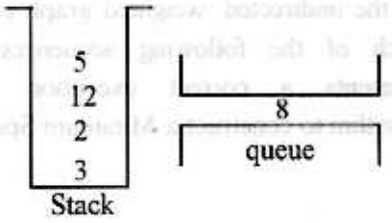
LEVEL–3

Linked Questions 17 & 18:

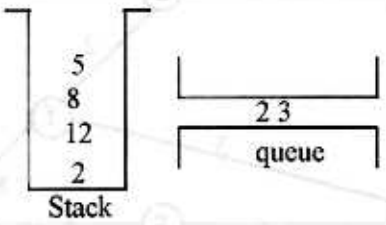
Q.17 We have a stack of integers s and a queue of integers q. Draw a picture of stack and Queue after the following operations:

pushstack	(s, 3)
pushstack	(s, 12)
enqueue	(q, 5)
enqueue	(q, 8)
popstack	(s, x)
pushstack	(s, 2)
enqueue	(q, x)
dequeue	(q, y)
pushstack	(s, x)
pushstack	(s, y)

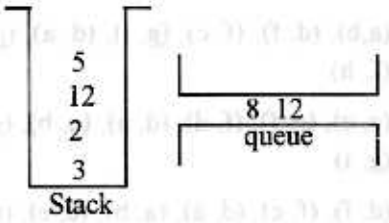
(a)



(b)



(c)



(d) None of these

Q.18 With reference to the correct option of above question (17), find the addition of all the elements from stack and queue.

(a) Stack = 22, queue =8

(b) Stack = 27, queue = 5

(c) Stack = 22, queue = 20

(d) None of these

Q.19 Find the number NUMB of element in a queue in terms of FRONT and REAR with FRONT ≤ REAR and REAR < FRONT.

(a) NUMB = N + REAR – FRONT = 1

NUMB = REAR + FRONT – 1

(b) NUMB = REAR + FRONT + 1

NUMB = N +REAR – FRONT + 1

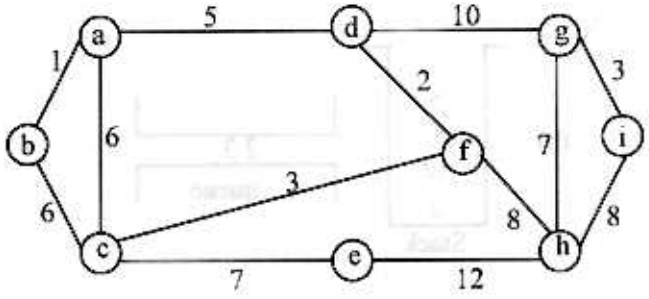
(c) NUMB = REAR – FRONT + 1

NUMB = N + REAR – FRONT + 1

(d) NUMB = N + REAR – FRONT – 1

NUMB = N + REAR + FRONT + 1

Q.20 For the undirected, weighted graph given below, which of the following sequences of edges represents a correct execution of Prim's algorithm to construct a Minimum Spanning Tree ?



(a) (a,b), (d, f), (f, c), (g, i), (d, a), (g, h), (c, e), (f, h)

(b) (c, e), (c, f), (f, d), (d, a), (a, b), (g, h), (h, f), (g, i)

(c) (d, f), (f, c), (d, a), (a, b), (c, e), (f, h), (g, h), (g, i)

(d) (h, g), (g, i), (h, f), (f, c), (f, d), (d, a), (a, b), (c, e)

Q.21 Assume that the operators +,–,* are left associative and ^ is right associative. The order of precedence (from highest to lowest) is ^,*,+,– . The postfix expression corresponding to the infix expression a + b * c – d^e^f is:

(a) abc * + def ^^ -

(b) abc * + de ^ f ^

(c) ab + c * d – e ^ f ^

(d) abc * + de ^ f ^

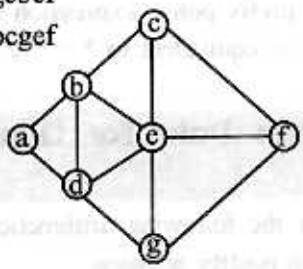
Q.22 Consider the following sequences of nodes for the undirected graph given below.

I. abefdgc

II. abefcgd

III. adgebcf

IV. adbcgef



A Depth First Search (DFS) is started at node a. The nodes are listed in the order they are first visited. Which of the above is (are) possible output(s) ?

(a) I and III only

(b) II and III only

(c) II, III and IV only

(d) I,II and III only

Q.23 Initially stack is empty. Stack is allocated N = 6 memory all. Find output

1. Set AAA: = 2 and BBB: = 5
2. Call PUSH (STACK, AAA)
   Call PUSH (STACK, 4)
   Call PUSH (STACK, BBB + 2)
   Call PUSH (STACK, 9)
   Call PUSH (STACK, AAA + BBB)
3. Repeat while TOP ≠ 0 ;
   Call POP (STACK, ITEM)
   Write: ITEM
   [End of loop.]
4. Return

(a) 2, 4, 7, 9, 7

(b) 7, 9, 7, 4, 2

(c) 2, 4, 9, 11, 9

(d) 9, 11, 9, 4, 2

GATE QUESTIONS

Q.24 The following sequence of operations is performed on a stack :

PUSH (10), PUSH (20), POP, PUSH (10), PUSH (20), POP, POP, POP, PUSH (20), POP

The sequence of value popped out is : **[GATE 1990]**

(a) 20,10,20,10,20,

(b) 10,20,10,10,20

(c) 10,20,20,10,20

(d) None of these

Q.25 The postfix expression for the infix expression A+B*(C+D)/F+D*E is **[GATE 1995] [2-Marks]**

(a) AB+CD+*F/D+E*

(b) ABCD+*F/DE*++

(c) A*B+CD/F*DE++

(d) A+*BCD/F*DE++

Q.26 A priority queue Q is used to implement a stack that stores characters. PUSH (C) is implemented as INSERT (Q,C,K) where K is an appropriate integer key chosen by the implementation. POP is implemented as DELETE MIN (Q). For a sequence of operations, the key chosen are in **[GATE 1997] [2-Marks]**

(a) non increasing order

(b) non decreasing order

(c) strictly increasing order

(d) strictly decreasing order

Q.27 Which of the following is essential for converting an infix expression to the postfix form efficiently? **[GATE 1997] [1-Mark]**

(a) An operator stack

(b) An operand stack

(c) An operand stack and an operator stack

(d) A parse tree

Q.28 Let f(n), g(n) and h(n) be functions defined for positive integers such that $f(n) = O(g(n), g(n)) \neq O(f(n))$, $g(n) = O(h(n))$ and $h(n) = O(g(n))$. Which one of the following statements is FALSE? **[IT-GATE 2004] [2-Marks]**

(a) $f(n) + g(n) = O(h(n) + h(n))$

(b) $f(n) = O(h(n))$

(c) $h(n) \neq O(f(n))$

(d) $f(n)h(n) \neq O(g(n)h(n))$

Q.29 A function f defined on stacks of integers satisfies the following properties. $f(\phi) = 0$ and $f(push(S,i)) = \max(f(S),0) + i$ for all stakes S and integers i.

If a stack S contains the integers 2, –3, 2, –1, 2 in order from bottom to top, what is f(S)? **[IT-GATE 2005] [1 Mark]**

(a) 6

(b) 4

(c) 3

(d) 2

Q.30 An Abstract Data Type (ADT) is **[GATE 2005] [1-Mark]**

(a) same as an abstract class

(b) a data type that cannot be instantiated

(c) a data type for which only the operations defined on it can be used, but none else

(d) all of the above

Q.31 The following postfix expression with single digit operands is evaluated using a stack:

8 2 3 ^ / 2 3 * + 5 1 * -

Note that ^ is the exponentiation operator. The top two elements of the stack after the first * is evaluated are: **[GATE 2007] [2-Marks]**

(a) 6, 1

(b) 5, 7

(c) 3, 2

(d) 1, 5

Q.32 Which of the following is valid for $2 \le i \le n$ and $a_i \le j \le W$? **[GATE 2008]** **[2-Marks]**

(a) $X[i,j] = X[i-1, j] \vee X[i, j-a_i]$

(b) $X[i,j] = X[i-1, j] \vee X[i-1, j-a_i]$

(c) $X[i,j] = X[i-1, j] \wedge X[i, j-a_i]$

(d) $X[i,j] = X[i-1, j] \wedge X[i-1, j-a_i]$

Q.33 Which entry of the array X, if TRUE, implies that there is a subset whose elements sum to W? **[GATE 2008]** **[2-Marks]**

(a) X [1, W]

(b) X [n, 0]

(c) X [n, W]

(d) X [n – 1, n]

ANSWER KEY

1	d	2	d	3	c	4	a	5	a
6	a	7	c	8	a	9	b	10	a
11	b	12	c	13	d	14	b	15	b
16	d	17	c	18	c	19	c	20	c
21	a	22	b	23	b	24	d	25	a
26	d	27	c	28	d	29	c	30	c
31	a	32	b	33	c				

SOLUTIONS

S.1 (d)

A problem in NP is complete if **some problem in NP can be reduced to it in polynomial time.**

Explanation:

We say that L is NP complete if the following statements are true about L:

(a) L is in NP

(b) For every language 'L' in NP there is a **polynomial time reduction of L'** to L.

S.3 (c)

Initially put the starting parentheses. Now the given expression has **3** consecutive parentheses which are appearing on the stack along with the one which is added.

S.4 (a)

The operation stack top (s) return the top element of stack s. The stacktop (s) operation can be decomposed into a pop and push operation.

i = stack top (s);

is equivalent to

i = pop (s);

push (s,i);

S.5 (a)

Symbol	STACK
6	6
9	6,9
2	6,9,2
+	6,11
*	66
12	66,12
3	66,12,3
/	66,4
–	62
)	

Final value of A is **62.**

S.6 (a)

The given expression is–

[a + (b–c)]*[(d – e)/(f + g – h)]

Let, the expression as- A * B ....(1)

where, A is [a + (b – c)]

B is [(d– e)/ (f + g – h)]

Now, prefix notation of A will be–

A = [a + (b – c)]

⇒ [a + (–bc)]

⇒ [+a – bc]

Now, prefix notation of B will be–

B = [d – e)/(f + g –h)]

⇒ [(– de)/(+ fg – h)]

⇒ [(– de)/(– + fgh)]

⇒ [/ – de – + fgh]

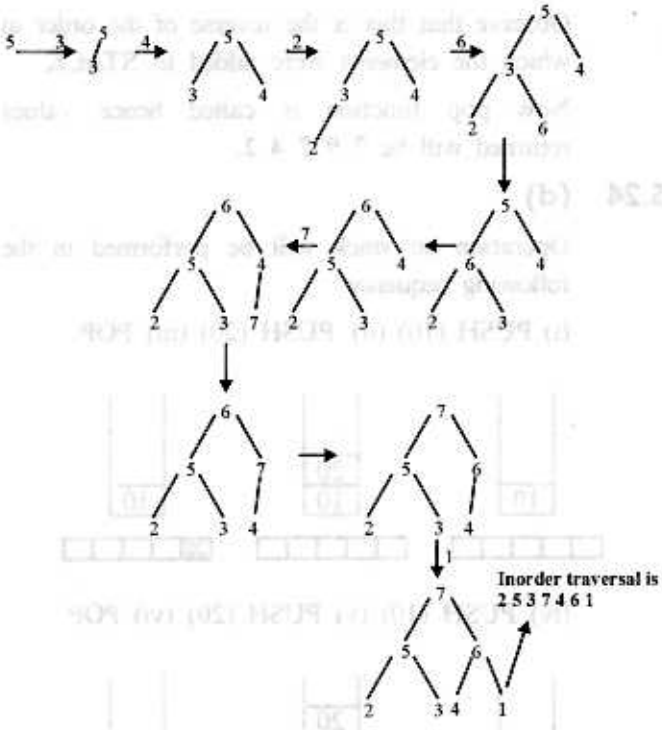
Now, coming back to equation (1)

[+a – bc] * [/ – de – +fgh]

⇒ *** + a – bc/ – de – + fgh**

S.8 (a)

Number 5,3,4,2,6,7,1 are to form a heap

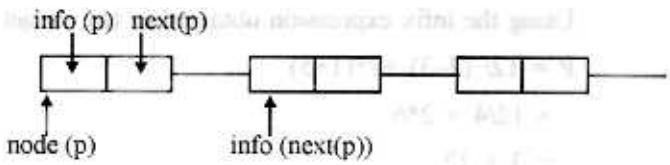


S.9 (b)

In this case, overflow will occur when STACK contains 7 elements and there is a PUSH operation to add another element to STACK.

S.10 (a)

If next (p) is not null, then info (next (p)) refers to the information portion of the node that follows node (p) in the list.



S.11 (b)

The space used for a list node is usually not twice the space used by an array element, since the elements in such a list usually consist of structures with many subfields. Also, it is sometimes possible to compress information and a pointer into a single word so that there is no space degradation.

S.12 (c)

The scope of the exponential operation is equivalent to finding the subtree rooted at the exponential operation. From the tree for the given expression, we can easily find out the subtree shown in option (c) and it corresponds to the subexpression $(5a+b)^3$.

S.13 (d)

There is no difference between the prefix polish expression P and the preorder of the tree formed by E and they are **both equivalent to * + * 2 x y ↑ – * 5 a b 3.**

Exp (2*x+y) (5*a–b)↑3

(*2x+y) *(*5a–b) ↑3

(+*2xy) * (–*5ab)↑3

(+*2xy) * ↑(–*5ab)3

*+*2xy↑–*5ab3

Now preorder of the tree is

*+*2xy↑–*5ab3.

Which is equivalent to above given exp.

S.14 (b)

Scanning from left to right, translate each operator from postfix to infix notation.

P = 12 , [7,3],2,1,5,+,*,+, –,/

= [12/(7–3)],2,1,5,+,*,+

= [12/(7–3)],2,[1+5],*,+

= [12/(7–3)], [2*1+5],+

= **12/ (7–3) + 2 * (1+5)**

S.15 (b)

Using the infix expression obtained in (a) we get

P = 12/ (7–3) +2*(1+5)

= 12/4 + 2*6

= 3 + 12

= **15.**

S.17 (c)

By basic operations of stack and queue we get (c).

S.19 (c)

If FRONT ≤ REAR then **NUMB = REAR – FRONT + 1**

IF REAR < FRONT, then FRONT – REAR – 1 is the number of empty cells, so

NUMB = N– (FRONT–REAR–1)

= **N + REAR –FRONT + 1**

S.20 (c)

Prim's algo will start with a tree that includes min. cost edge of graph G. Then edges are added to this tree one by one.

The next edge (i,j) to be added is such that i is a vertex already included in tree, j is a vertex not yet included and the cost of (i,j) is min. among all edges.

Here, when we start with min. cost edge (a,b) so after (a,b), (b,c) must be added to the tree, but in option (i) option (d,f) is added thats according to kruskal's and not prim.

so we take min. edge, i.e. (d,f)

In this way we move on selecting next min. cost edge at each node such that ckt. is not formed.

S.21 (a)

a + b * c – d ^ e ^ f

a + b * c – d ^ ef ^

a + b * c – def ^^

a + bc * – def ^^

abc * + – def ^^

abc * + def ^^ –

S.22 (b)

In I → a b e f d g c ; d can't be traversed after f and before g and c.

II → a b e f c g d

Here, all the nodes are traversed according to DFS.

III → a d g e d c f

Here, also all nodes are traversed acc. to DFS.

IV → a d b c g e f

It is BFS (Breadth first search)

S.23 (b)

Step1: Sets AAA = 2 and BBB = 5

Step 2: Pushes AAA = 2,4; BBB + 2 =7, 9 and AAA + BBB = 7 onto STACK, yielding

STACK: 2, 4, 7, 9, 7,_____

Step 3: Pops and prints the elements of STACK until STACK is empty. Since the top element is always popped, the output consists of the following sequence:

7, 9, 7, 4, 2

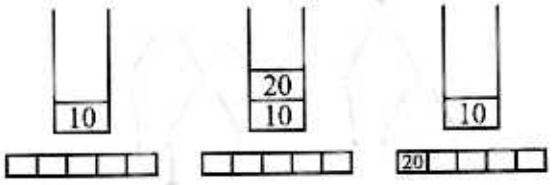
Observe that this is the reverse of the order in which the elements were added to STACK.

Now pop function is called hence values returned will be **7 9 7 4 2.**

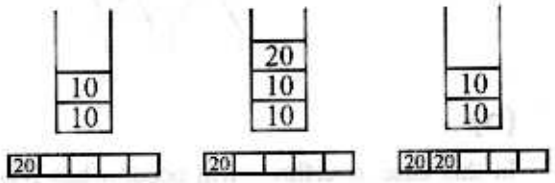
S.24 (d)

Operation on stack will be performed in the following sequence:

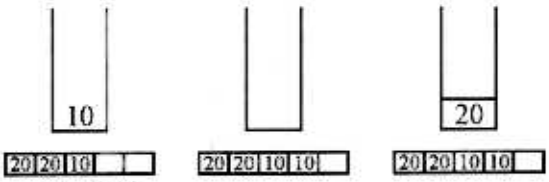
(i) PUSH (10) (ii) PUSH (20) (iii) POP



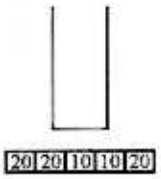
(iv) PUSH (10) (v) PUSH (20) (vi) POP



(vii) POP (viii) POP (ix) PUSH (20)



(x) POP



So the sequence of popped values is

20, 20, 10, 10, 20

S.25 (a)

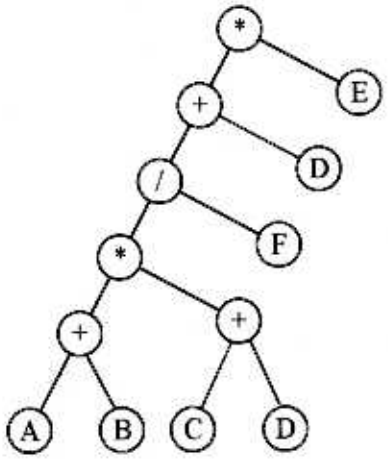
Given infix expression is

A + B * (C + D) / F + D * E

The expression can be written as

$$\Big(\big(\big(\big((A+B)*(C+D)\big)/F\big)+D\big)*E\Big)$$

Thus tree for the above expression is



Postfix traversal of above tree gives [LRV]:

AB + CD + * F/D + E*

S.26 (d)

PUSH (C) → C is inserted in stack

INSERT (Q, C, K) means C is inserted in queue at key K position.

POP is used for DELETE MIN (Q) from queue.

Sol. for a sequence of operations, the keys chosen are in **strictly decreasing order.**

S.27 (c)

For converting an infix expression to the postfix form efficiently, requirement of **operand stack and an operator stack** is must.

S.28 (d)

$$O(f(n))\ O(h(n)) = O(g(n)\ h(n))$$
$$f(n) = O(g(n))$$
$$h(n) = O(g(n))$$
$$f(n)\ h(n) = (O(g(n)))\ (O(g(n)))$$
$$= O(g(n))\ O(h(n))$$

from given assumption.

∴ **(d) is false statement.**

S.29 (c)

∵ $f(\phi) = 0$ ← Top of the stack.

f(push (s, i)) = max (f(s), 0) + i

where i = 2, –3, 2, –1, 2

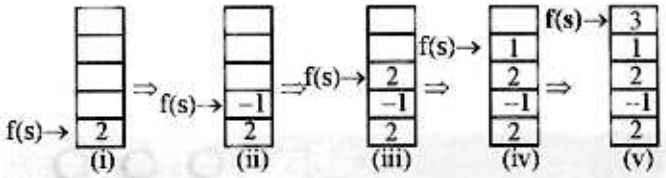
(i) f(push(s, 2)) = max(f(ϕ), 0) + 2 = 0 + 2 = 2

(ii) f(push(s, –3)) = max(f(s), 0) + (–3) = 2 + (–3) = –1

(iii) f(push(s, 2)) = max(f(s), 0) + 2 = 0 + 2 = 2

(iv) f(push(s, –1)) = max(f(s), 0) + –(1) = 2 + (–1) = 1

(v) f(push (s, 2)) = max(f(s), 0) + 2 = 1 + 2 = 3



S.30 (c)

ADT refers to a programmer defined data type together with a set of operations that can be performed on that data.

S.31 (a)

Exp./symbol	Op 1	Op 2	Value	Top s (RL)
8				8
2				8,2
3				8,2,3
^	2	3	8	8,8
/	8	8	1	1
2				1,2
3				1,2,3
*	2	3	6	1,6

S.32 (b)

The subset-sum problem is NP-complete problem. For example given a set S = {–7, –3, –2, 5, 8,} the algorithm determine whether the sum of the element of S' is zero such that S' ⊆ S. For example if S' = {–3, –2, 5} then return yes. In given dynamic algorithm x is a two dimensional Boolean array containing n rows and W + 1 columns where W is a positive weight or sum.

$$x[i,j]\ 1 \le i \le n,\ 0 \le j \le W$$

The i^{th} row determines the element of subset of S, and j^{th} column determine the corresponding weight.

If $2 \le i \le n$ and $a \le j \le W$

Let subset S' = $\{a_i, a_2, ..., a_i\}$ the weight as a_1, ...,a_2 is W. So

$$\mathbf{X[i,j] = X[i - 1, j] \vee X[i - 1, j - a_i]}$$

The now starts from 2^{nd} position so i = i – 1. The column position starts from but we will add upto a_i^{th} position so $j = j - a_i$.

S.33 (c)

The entry **X[n, W]** is true, because we find the weight W of subset of S which contains n elements.

UNIT–6

COMPILER DESIGN

UNIT-6

COMPILER DESIGN

6.1 INTRODUCTION TO COMPILER

LEVEL–1

Q.1 If xy translator is used for translation of COBOL into assembly language then in this case xy is called as _____

(a) Assembler

(b) Compiler

(c) Interpreter

(d) None of these

Q.2 During which type of analysis the compiler tries to detect construction that have the right syntactic structure but no meaning to the operation involved.

(a) Syntax analysis

(b) Semantic analysis

(c) Both (a) and (b)

(d) Neither (a) nor (b)

Q.3 Which of the following is a semantic action?

(a) Checking semantic validity of constructs in synthesis phase

(b) Determining the meaning of synthesis phase

(c) Constructing an intermediate representation

(d) all of the above

Q.4 Consider the following situation:

Many programming language definitions require a complier to report an error every time a real number is used to index an array. Some language specification may permit some operand corrections.

The given situation is handled in _____

(a) Syntax analysis

(b) Semantic analysis

(c) Linear analysis

(d) Hierarchical analysis

Q.5 Which of the following statement is true?

(i) A symbol table is a data structure containing a record for each identifier, with fields for the attributes of the identifier.

(ii) When an identifier in the source program is detected by the lexical analyzer, the identifier is entered into the symbol table with its corresponding attributes.

(a) (i)

(b) (ii)

(c) Both (i) and (ii)

(d) Neither (i) nor (ii)

Q.6 Which type of compilers are designed to combine the main advantage of interpreters and compilers.

(a) Optimizing compilers

(b) Incremental compilers

(c) Single pass compilers

(d) None of these

Q.7 Which of the following(s) is(are) function(s) of preprocessors

(i) produce input to compliers

(ii) file inclusion

(iii) Language extensions

(iv) Macro processing

(a) (ii),(iv)

(b) (i),(iv),(iii)

(c) (ii),(iii),(iv)

(d) (i), (ii), (iii), (iv)

Q.8 In implementation point of view, front end normally consists of _____

(i) Lexical analysis

(ii) Syntax analysis

(iii) Semantic analysis

(iv) Intermediate code generation

(a) (i),(ii)

(b) (i)

(c) (i),(iii)

(d) (i), (ii), (iii),(iv)

Q.9 Which of the following is incorrect?

(i) Lexical constructs do require recursion

(ii) Cortext free grammar are a formalization of recursive rules that can be used to guide syntactic analysis

(a) (i)

(b) (ii),(iii)

(c) (ii)

(d) Neither (i) nor (ii)

Q.10 Which uses the Hierarchical structure determined by the syntax analysis phase to identify the operators and operands of expressions and statements.

(a) Hierarchical analysis

(b) Semantic analysis

(c) Linear analysis

(d) None of these

Q.11 For achieving aim of optimizing compilers requires

(i) additional analysis of the program structure and the definition of program variable.

(ii) detect the possibilities of removing loop invariant code out of a loop

(iii) eliminating redundant expression evaluation

(a) (i),(iii)

(b) (i),(ii)

(c) (ii),(ii)

(d) (i),(ii),(iii)

Q.12 Which of the following statement is correct?

(a) A cross complier which runs on one machine and generates code for another machine.

(b) The difference between a cross complier and a normal compiler is in terms of the code generation only.

(c) Cross compilers are widely used for mini and micro compilers

(d) All of above

Q.13 Which can detect errors where the characters remaining in the input do not form any token of the language.

(a) Syntax analysis phase

(b) Semantic analysis phase

(c) Lexical analysis phase

(d) all of above

Q.14 In implementation point of view, back end normally consists of _____

(i) Code generation

(ii) Code optimization

(iii) Error handling

(iv) Symbol table operations

(a) (ii),(iv)

(b) (ii),(iii),,(iv)

(c) (i),(ii),(iii)

(d) (i),(ii),(iii),(iv)

Q.15 Assembler is a program that

(a) places programs into memory and prepares them for execution

(b) automates the translation of assembly language into machine language

(c) accepts a program written in a high level language and produces an object program.

(d) appears to execute a resource as if it were machine language

Q.16 An interpreter is a program that

(a) places programs into memory and prepares them for execution.

(b) automates the translation of assembly language into machine language.

(c) accepts a program written in a high level language and produces an object program.

(d) appears to execute a source program as if it were machine language.

Q.17 A compiler is a program that

(a) places programs into memory and prepares them for execution.

(b) automates the transition of assembly language into machine language.

(c) accepts a program written in a high level language and produces an object program.

(d) appears to execute a source program as if it were machine language.

Q.18 Compiler can diagnose

(a) grammatical errors only

(b) logical errors only

(c) grammatical as well as logical errors

(d) neither grammatical nor logical errors.

Q.19 A programmer, by mistake, writes an instruction to divide, instead of a multiply, such errors can be detected by a/an

(a) compiler

(b) interpreter

(c) compiler or interpreter test

(d) none of these

Q.20 What is true about machine language?

(a) It is understood by the computer

(b) It varies from one model of computer to another

(c) It may always be represented by binary numbers

(d) All of these

Q.21 A computer cannot " boot" if it does not have the

(a) Compiler

(b) Loader

(c) Operating System

(d) Assembler

Q.22 Arrange the following configuration for CPU in decreasing order of operating speed?

(a) Hardware Control, Vertical micro programming, Horizontal micro programming.

(b) Hardware Control, Horizontal micro programming, vertical micro programming.

(c) Horizontal micro programming, Vertical micro programming, Hardware Control.

(d) Vertical micro programming, Horizontal micro programming, Hardware control

Q.23 The most common system security method is

(a) password

(b) encryption

(c) firewall

(d) all of these

Q.24 Which of the following do not translate a source program into a machine language program?

(a) Single pass compiler

(b) Interpreter

(c) Assembler

(d) None of these

Q.25 Which of the following statements is incorrect?

(a) Executing a program written in a high– level programming languages is basically a two steps process i.e. the source program must first be compiled, i.e., translated into the object program. Then the resulting object program is loaded into memory and executed.

(b) Interpreter are often smaller than compilers and facilitate the implementation of complex programming language constructs.

(c) In most of the code languages, one communicates directly with the operating system with no prior translation at all.

(d) None of these

Q.26 Which of the following are the properties of intermediate code generation?

(i) Intermediate code should be easy to produce

(ii) Intermediate code should be easy to translate into the target program

(iii) Each instruction has at most one operator in addition to the assignment.

(a) (i).(ii)

(b) (i)

(c) (i).(iii)

(d) (i). (ii). (iii). (iv)

Q.27 Diagnostic Compilers and Correcting compilers are

(a) same

(b) different

(c) both useless

(d) None of these

Q.28 Which provides general purpose facilities required in most application domains, such a language is independent of specific application domains and results in a large specification gap which has to be bridged by an application designer.

(a) Problem oriented language

(b) procedure oriented language

(c) object oriented language

(d) None of these

Q.29 Which of the following statement is true ?

(a) Macro definitions cannot appear within other macro definitions in assembly language programs.

(b) Overlaying is used to run a program which is longer than the address space of computer.

(c) Mutual memory can be used to accommodate a program which is longer than the address space of computer

(d) It is not possible to write interrupt service routines in a high level language.

Q.30 A programming language is to be designed to run on a machine that does not have big memory. The language should

(a) prefer an two-pass compiler to an one-pass compiler

(b) prefer an one-pass compiler to a two-pass compiler

(c) prefer an interpreter to a compiler

(d) not support recursion

Q.31 Two procedures both of which treat the other as a called procedure and itself the callee, are called as

(a) master-slave routines

(b) sub-sub-routines

(c) co-routines

(d) ambiguous master-slave routines

Q.32 A compiler-compiler is a/an

(a) compiler which compiles a compiler program

(b) software tool used in automatic generation of a compiler

(c) compiler written in the same language it compiles

(d) another name for cross-compiler

Q.33 A compiler which allows only the modified section of the source code to be recompiled is called as

(a) incremental compiler

(b) reconfigurable compiler

(c) dynamic compiler

(d) selective compiler

Q.34 Choose the correct statement.

(a) Any software can be simulated by hardware.

(b) Any hardware can be simulated by software

(c) Firmware is nothing but hardware implementation of software.

(d) Firmware is nothing but software implementation of hardware.

Q.35 A _____ grammar has no derivation of the form _____ for any non terminal A:

(a) Context free, $S \rightarrow \varepsilon$

(b) Cycle free, $A \overset{+}{\Rightarrow} A$

(c) Ambiguous, $A \overset{*}{\Rightarrow} A$

(d) Unambiguous, $S \overset{*}{\Rightarrow} wxy$

LEVEL–2

Q.36 Match the following:

	Group I		Group II
1.	A language translator	(i)	bridges the specification gap between two PLs.
2.	A preprocessor	(ii)	bridges an execution gap to the machine language of a computer system.
3.	A language migrator	(iii)	language processor bridges an execution gap.

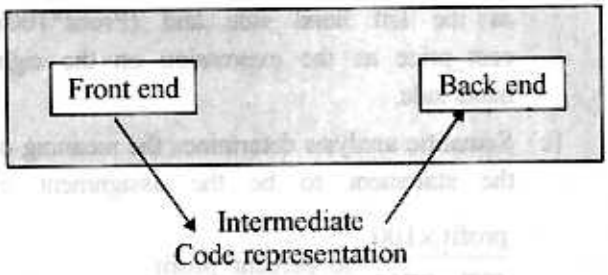
(a) (1)-iii (2)-ii (3)-i

(b) (1)-ii (2)-iii (3)-i

(c) (1)-i (2)-ii (3)-ii

(d) None of these

Q.37 Consider the following, two pass schematic for language processing.



The desirable properties of intermediate code are

(i) ease of use

(ii) efficient processing algorithm for construction and analyzing the intermediate code

(iii) intermediate code must be compact

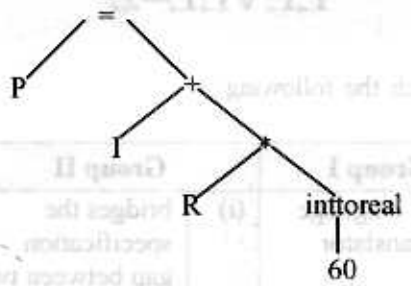
(a) (i),(ii)

(b) (ii),(iii)

(c) (i),(ii),(iii)

(d) (i),(iv)

Q.38 Consider the following declaration.



The above declaration represents:

(a) Hierarchical analysis but it is not valid one

(b) Syntax analysis but is not valid one

(c) Semantic analysis with valid conversion from integer to real

(d) None of these

Q.39 Consider the statement

percent_profit := (profit*100)/cost_price

in some programming language.

Then which of the following statement is incorrect?

(a) Lexical analysis identifies:=*and/ as operation, 100 as a constant and the remaining strings as identifiers.

(b) Syntax analysis identifies the statement as an assignment statement with percent_profit as the left hand side and (Profit*100)/ cost_price as the expression on the right hand side.

(c) Semantic analysis determines the meaning of the statement to be the assignment of $\frac{\text{profit} \times 100}{\text{cost\_price}}$ to percent_profit.

(d) None of these

Q.40 In some programming languages, an identifier is permitted to be a letter followed by any number of letters or digits. If L and D denotes the sets of letters and digits respectively, which of the following expression define an identifier?

(a) $(L \cup D)$

(b) $L(L \cup D)$

(c) (L.D)

(d) L.(L D)

Q.41 Consider the following translation statement

X:=Y + Z * 60

The which of the following statement is correct about lexical analysis phase?

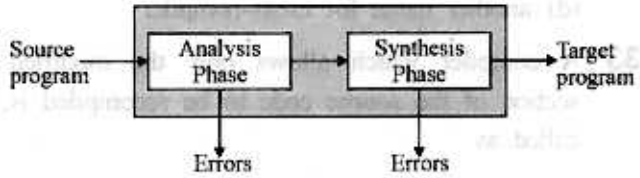
(a) When an identifier Z is found, then lexical analyzer generates a token say id

(b) When an identifier Z is found, then lexical analyzer enters the lexeme Z into the symbol table if is not already there.

(c) The lexical value associated with occurrence of id (token generated due to Z) points to the symbol table entry for Z.

(d) all of the above

Q.42 Consider the following language processor.



The language processing can be performed on a statement by statement basis that is, analysis of a source can be immediately followed by synthesis of equivalent target statements.

This may not be feasible due to

(i) Forward references

(ii) Issues concerning memory requirements and organization of a language processor

(a) (i)

(b) (ii)

(c) Both (i) and (ii)

(d) Neither (i) nor (ii)

Q.43 Consider the following Intermediate code in general form

x = y op z

then with x, y, z we can represent _____

(i) names

(ii) constants

(iii) compiler generated temperaries.

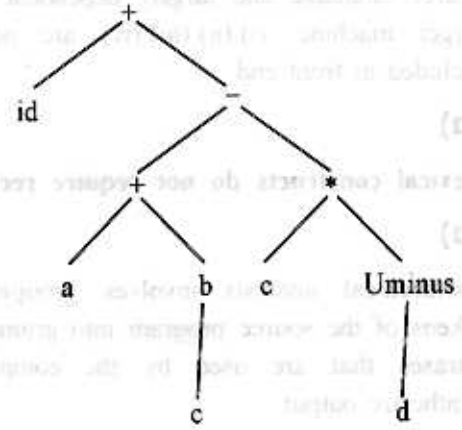
(a) (i),(ii)

(b) (iii),(i)

(c) (i),(ii),(iii)

(d) (i),(iii)

Q.44 The intermediate code generated for the following syntax tree in Postfix notation is:



(a) id, acb + cd * – +

(b) id, acb + cd Uminus * – +

(c) id, ac + cd b + – +

(d) none of these

Q.45 Consider the following declaration

```
for (i=0; i<=5; i++)
{   c=c+i;
}
```

Then certain process is applied to above code, and due to it we get output as follows

for (i=0; i<=5++) {c}

Then, that process may be_____

(a) Linear analysis

(b) Hierarchical analysis

(c) Semantic analysis

(d) None of these

ANSWER KEY

1	b	2	b	3	d	4	b	5	a
6	b	7	d	8	d	9	a	10	a
11	d	12	d	13	c	14	d	15	b
16	d	17	c	18	a	19	d	20	d
21	b	22	b	23	a	24	b	25	d
26	a	27	b	28	b	29	b	30	a,c,d
31	c	32	b	33	a	34	a,b,c	35	a
36	b	37	c	38	c	39	d	40	b
41	d	42	c	43	c	44	b	45	a

SOLUTIONS

S.1 (b)

A Translator is a program that takes as input a program written in one programming language (the source language) and produces as output a program in another language (the object language) if the source language is a high level language such as COBOL and the object language is an assembly language then in the case **Translator is called as compiler.**

S.2 (b)

During **semantic analysis** the compiler tries to detect constructs that have the right syntactic structure but no meaning to the operation involved.

e.g., if we try to add two identifier, one of which is the name of an array, and the other is the name of a procedure.

S.3 (d)

(a), (b), (c) are all semantic actions.

S.4 (b)

Type checking is an important component of **semantic analysis.**

S.5 (a)

Normally the attributes of an identifier cannot normally be determined during lexical analysis.

S.6 (b)

Incremental compilers are designed to combine the main advantages of interpreters and compilers. This is achieved by complying the source program in such a manner that easy, low cost program modification is feasible

S.7 (d)

(i),(ii),(iii),(iv) are functions of preprocessors.

S.8 (d)

In an implementation, activities from more than one phase are often grouped together. Often the phases are collected into a front end and a back end. The front end consists of those phases, as parts of phases, that depend primarily on the source language and largely dependent of the target machine. (i),(ii),(iii),(iv) are normally included in front end.

S.9 (a)

Lexical constructs do not require recursion.

S.10 (a)

Hierarchical analysis involves grouping the tokens of the source program into grammatical phrases that are used by the compiler to synthesize output.

Example: position:= initial + rate * 60.

S.11 (d)

The aim of optimizing compilers is to produce efficient target code. To achieve that, it requires all (i), (ii), (iii).

S.12 (d)

(a), (b), (c) ⇒ are true statement about cross compiler.

(a) Definition of cross compilers

(b) Difference between cross compiler and a normal compiler.

(c) Cross compilers are widely used for mini and micro computer.

S.13 (c)

Lexical analysis phase can detect errors where the characters remain in the input.

S.14 (d)

The back end includes those portions of the compiler that depend on the target machine and generalizing these portions do not depend on the source language in the back end. Back end define code generation, along with error handling and symbol table operations.

S.15 (b)

A loader is a program that places programs into memory and prepares them for execution.

An assembler is a program that translates assembly language into machine language.

S.16 (d)

An interpreter is a program that appears to execute a source program as if it were machine language.

S.17 (c)

A compiler is a program that **accepts a program written in a high level language and produces an object program.**

S.18 (a)

Compilers cannot diagnose errors in logic. They can **diagnose only grammatical errors in writing the source program.**

S.19 (d)

Program containing such errors will be successfully executed and give incorrect answer. To avoid such errors it is necessary to use test cases which are manually computed and compared with computer solutions.

S.22 (b)

Hardware is the fastest, since horizontal micro programming involves lesser number of instructions, it is faster than vertical microprogramming.

S.23 (a)

Password is the most commonly used system security method. Encryption is the process of transforming information using an algorithm to make it unreadable to anyone. Firewall is basically used for network security as it prevents unauthorized is unwanted communication.

S.24 (b)

Single pass compiler is a compiler that passes through the source code of each compilation only once.

An interpreter may be a program that either

1. executes the source code directly.
2. translates source code into some efficient intermediate representation (code) and immediately executes this.
3. explicitly executes stored precompiled code made by a compiler which is part of the interpreter system.

S.25 (d)

Statements (a), (b), (c) are correct statement related to interpreter and complier.

S.26 (a)

(i) and (ii) are properties of intermediate code generation and (iii) is related to three address code which is important form of the intermediate code generation.

S.28 (b)

Definition and function of procedure oriented language.

S.29 (b)

All (a), (c) and (d) are false. The difference between (b) and (c) is vital.

8086 processor has 32 bit address capacity (32 pins for address) hence it is capable of accessing $2^{32}=2^2\times2^{30}=4GB=$ 4096 MB of main memory.

At present day compute is have at most 32 MB RAM. The address space of 8086 processor is 4 GB and hence virtual memory can be at most 4 GB. If a program is of 5 GB, we have to use only **overlaying technique.**

S.30 (a, c, d)

In a **two-pass compiler,** the space occupied by the first pass can be used by the second pass. Since **an intepreter** translates and executes one line at a time, it is less complex than a compiler. So, it should occupy less space. **Recursion** needs run-time stack whose size changes dynamically and often increases exponentially with respect to the associated parameter.

S.31 (c)

Co-routines are program components that generalize sub-routines to allow multiple entry points for suspending and resuming execution at certain locations.

S.32 (b)

These are **tools that can be used to automatically generate modules of a compiler.** e.g., YACC (Yet Another Compiler Compiler) generates the parser when the syntax of the language, in the form of CFG is given as input.

S.34 (a, b, c)

In fact software and hardware are logically equivalent. The function of the hardware can be simulated by the software and the function of the software can be simulated by the hardware.

Firmware is a software that is embedded onto a piece of hardware in order to control that hardware.

S.37 (c)

Desirable properties of Intermediate code

- Ease of use : It should be easy to contruct and analyze.
- Processing efficiency : efficient algorithm must be created for constructing and analyzing the intermediate code
- Memory efficiency : Intermediate code must be compact.

S.38 (c)

In **semantic analysis** certain checks are performed to ensure that the components of a program fit together meaningfully

S.39 (d)

(a), (b), (c) are correct about Lexical analysis, syntax analysis and semantic analysis.

S.40 (b)

L(LUD) is the set of all strings of letters and digits beginning with a letter.

S.41 (d)

(a), (b), (c) are true statements about lexical analysis.

S.42 (c)

The situation given in the question may not be feasible due to

(i) Forward references : A forward reference of a program entity is reference to the entity which proceeds its definition in the program.

(ii) Issues concerning memory requirements and organization of a language processor.

S.43 (c)

There address code is one of the form of Intermediate code.

x: = y op z

Where **x, y ad z can be represent name, constants or compiler generated temporaries** and OP stands for any opertor.

S.45 (a)

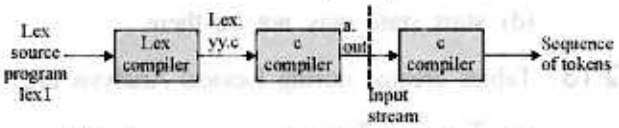
In linear analysis, the stream of characters making up the source program is read from left to right and grouped into tokens that are sequences of characters having a collective meaning.

6.2

LEXICAL ANALYSIS

LEVEL–1

Q.1 Consider the following diagram



The lex.yy.c consists of a_____

(a) regular expression of lex.1 with standard lexemes

(b) tabular representation of a transition diagram constructed from the regular expressions of lex.1 together with a standard routine that uses the table to recognize lexemes.

(c) regular expression with standard lexemes and object program

(d) None of these

Q.2 ..... determines the syntactic structure of the input statement

(a) Semantic Analysis

(b) Pragmatics

(c) Parsing

(d) None of these

Q.3 Semantic Analysis processes

(a) declarative statements

(b) imperative statements

(c) both (a) and (b) above

(d) None of these

Q.4 What is Kleen closure of L?

(a) One and only one concatenations of L

(b) One or more concatenations of L

(c) Zero or more concatenations of L

(d) Zero or one concatenations of L

Q.5 Which of the following is not a rule, for a language denoted by the regular expression?

(a) A language denoted by a regular expression is said to be a regular expression.

(b) If a is a symbol in S (alphabet), then a is a regular set.

(c) If r is a regular expression denoting the language L(r), then (r)* is a regular expression denoting (L(r))*

(d) None of these

Q.6 The declaration section of every lex program consists of _____

(i) Variables

(ii) manifest constants

(iii) regular definition

(a) (i),(ii),(iii)

(b) (ii)

(c) (i),(ii)

(d) (ii),(iii)

Q.7 Given a regular expression r and an input string x, then DFA formed from it has space complexity as _____

(a) $O(2^{|r|})$

(b) $O(2|r|)$

(c) $O(|r| \times |x|)$

(d) $O(|r|)$

Q.8 Which one of the following is a function of Lexical analyzer?

(i) read the input characters and produce as output a sequence of token

(ii) stripping out from the source comments and white space in the form of blank, tab and new line characters

(iii) Correlating error message from the compiler with the source program

(a) (i)

(b) (i),(ii)

(c) (i),(iii)

(d) (i),(ii),(iii)

Q.9 Which one is an implementation approach of a lexical analyzer?

(a) For implementation of lexical analyzer, use a lexical analyzer generator, to produce the lexical analyzer from a regular expression based specification.

(b) For implementation of lexical analyzer, write the lexical analyzer in a conventional system programming language using the i/o facilities of that language to read the input.

(c) For implementation of lexical analyzer, write the lexical analyzer in assembly language and explicitly manage the reading of input.

(d) All of the above

Q.10 Which of the following statement is correct?

(a) The set of string of balanced parenthesized cannot be described by a regular expression, but, this set can be specified by a context free grammar.

(b) The set {wcw | w is a string of a's and b's} cannot be denoted by any regular expression, nor it can be described by a context free grammar.

(c) both (a) and (b)

(d) Neither (a) nor (b)

Q.11 In analyzing the compilation of PL/I program, the term "Machine independent optimization" is associated with

(a) recognition of basic analytic construction through reductions.

(b) recognition of basic elements and creation of uniform symbols.

(c) creation of more optimal matrix.

(d) use of macro processor to produce more optimized assembly code.

Q.12 In an incompletely specified automata

(a) no edge should be labelled epsilon (ε)

(b) from any given state, there can, be any token leading to two different states

(c) some states have no transition on some tokens

(d) start state may not be there

Q.13 Tables created during Lexical Analysis are:

(a) Terminal Table

(b) Identifier Table

(c) Literal Table

(d) Uniform Symbol Table

(e) All the above

Q.14 During lexical analysis source program is scanned:

(a) Serially

(b) Sequentially

(c) By Index

(d) None of the above

Q.15 In a two-pass assembler the pseudo-code EQU is to be evaluated during

(a) pass 1

(b) pass 2

(c) not evaluated by the assembler

(d) none of the above

LEVEL–2

Q.16 Consider the following Fortran statement

E = M*C*4

then for E, the attribute value is

(a) <id,E>

(b) <id, symbol table entry for E>

(c) <id, pointer to symbol table entry for E>

(d) none of these

Q.17 Consider the following C–declaration

int a;

fi(a==f(x)).....

and if the string fi is encountered in a C, n for first time in the context then which of the following statement is correct about it

(a) A lexical analyzer cannot tell whether fi, is a misspelling of the keyword if or an undeclared function identifier

(b) Since fi is valid identifier. The lexical analyzer must return the token for an identifier and let some other phase of the compiler handle any error.

(c) Neither (a) and (b)

(d) Both (a) and (b)

Q.18 The string 1101 does not belong to the set represented by

(a) 110*(0+1)

(b) 1(0+1)*101

(c) (10)*(01)*(00+11)*

(d) (00+(11)*01)*

Q.19 Which is not true about syntax and semantic parts of a computer language

(a) Syntax is generally checked by the programmer.

(b) Semantic is the responsibility of the programmer.

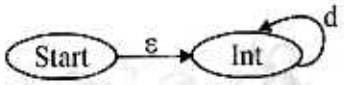
(c) Semantic is checked mechanically by a computer.

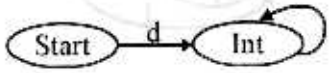
(d) Both (b) and (c)

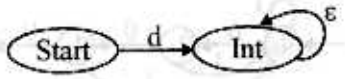
LEVEL–3

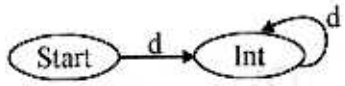
Q.20 For following expression the corresponding DFA is _____

<integer> ::d|<integer>d

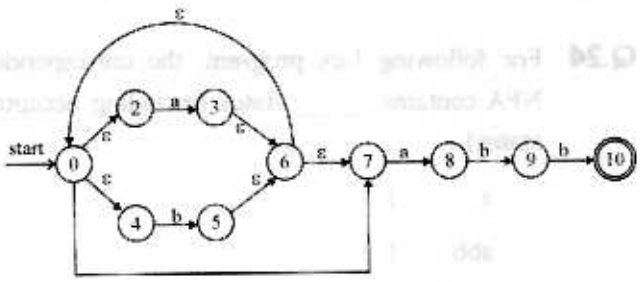
(a) 

(b) 

(c) 

(d) 

Q.21 Consider the following NFA,



Let x be the string consisting of the single character a. Then what will be the final result?

(a) NFA will accept x

(b) NFA will reject x

(c) NFA is not in position to accept or reject

(d) None of these

Linked Questions 22 & 23:

Q.22 A regular expression that represents a language accepting all string containing 3 consecutive zeros is

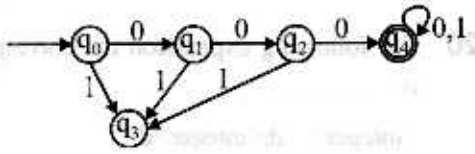
(a) 000 (1+0)*

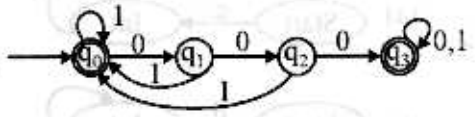
(b) (1+0)*00(0,ε)

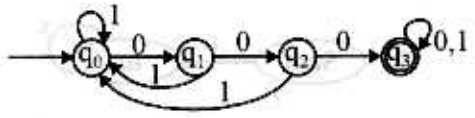
(c) (0*1*)000(0+1)*

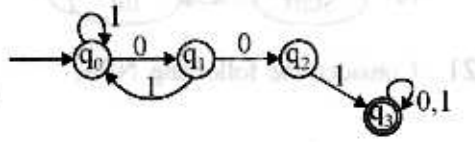
(d) (10+10)*(000+11)*

Q.23 Choose the DFA that most approximately describes the resulting regular expression.

(a)


(b)


(c)


(d)


Q.24 For following Lex program, the corresponding NFA contains _____ states (including accepting states)

a { }

abb { }

a^*b^+ { }

(a) 8

(b) 7

(c) 5

(d) none of the above

Q.25 If the regular set A is represented by A = (01 + 1)* and the regular set 'B' is represented by B = ((01)*1*)* then

(a) A = B

(b) A and B are uncomparable

(c) B ⊂ A

(d) A ⊂ B

Q.26 For an unsigned number such as 5292, 49.37, 6.336 E4 or 1.894E–4, then which of the following is valid regular definition.

(a) digit → 0|1|.....|9

digits → digit digit*

optional fraction → · digits|ε

optional–exponent → (E(+|–|ε) digits)|ε

num → digits optional – fraction optional–exponent

(b) digit → 0|1|.....|9

digits → digit digit*

optional fraction → · digits|ε

optional–exponent → (E(+|–|ε) digits)|ε

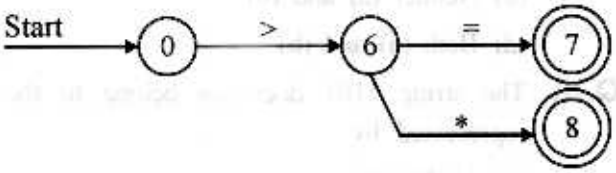
(c) digit → 0|1|.....|9

optional fraction → · digits|ε

optional–exponent → (E(+|–|ε) digits)|ε

(d) none of the above

Q.27 The following transition diagram is for:



(i) >

(ii) >=

(a) Only (i)

(b) Only (ii)

(c) Neither (i) nor (ii)

(d) Both (i) and (ii)

GATE QUESTIONS

Q.28 Which of the following strings can definitely be said to be token without looking at the next input character while compiling a Pascal program?

[GATE 1995]

[1-Mark]

I. begin II. Program III. < >

(a) I

(b) II

(c) III

(d) All of the above

Q.29 Type checking is normally done during

[GATE 1998]

[1-Mark]

(a) lexical analysis

(b) syntax analysis

(c) syntax directed translation

(d) code optimization

Q.30 The number of tokens in the Fortran statement DO 10 I =1.25 is **[GATE 1999]**

[2-Marks]

(a) 4

(b) 3

(c) 5

(d) None of the above

Q.31 The number of tokens in the following C statement Print f ("i=%d,&i=%x",i,&i); is

[GATE 2000]

[1-Mark]

(a) 3

(b) 26

(c) 10

(d) 21

Q.32 Which of the following are true?

[GATE 2008]

[2-Marks]

(i) A programming language which does not permit global variables of any kind and has no nesting of procedures/functions, but permits recursion can be implemented with static storage allocation.

(ii) Multi level access link (or display) arrangement is needed to arrange activation records only if the programming language being implemented has nesting of procedures/ function

(iii) Recursion in programming languages cannot be implemented with dynamic storage allocation.

(iv) Nesting of procedures/functions and recursion require a dynamic heap allocation scheme and cannot be implemented with a stack-based allocation scheme for activation records.

(v) Programming languages which permit a function to return a function as its result cannot be implemented with a stack based storage allocation scheme for activation records.

(a) (ii) and (v) only

(b) (i), (iii) and (iv) only

(c) (i), (ii) and (v)

(d) (ii),(iii) and (v) only

ANSWER KEY

1	b	2	c	3	c	4	c	5	d
6	a	7	a	8	d	9	d	10	c
11	c	12	d	13	e	14	b	15	a
16	c	17	d	18	c	19	d	20	d
21	b	22	c	23	c	24	a	25	a
26	a	27	d	28	c	29	b	30	c
31	d	32	a						

SOLUTIONS

S.1 (b)

When lex.l is run through lex compiler to produce lex. yy.c. The program lex.yy.c consist of a tabular representation of a transition diagram constructed from the regular expressions of lex.l together with a standard routine that uses the table of recognize lexemes.

S.2 (c)

Parsing determines the syntactic structure of a string.

S.3 (c)

During sematic analysis, a compiler builds and examines as intermediate representation of the source program and checks it for consistency.

S.4 (c)

Kleen closure of $L = L^* \bigcup_{i=0}^{\infty} L_i$ which indicates that **"zero or more concatenations of L".**

S.5 (d)

(a) $\Rightarrow$ A language denoted by a regular expression is said to be a regular set.

(b) $\Rightarrow$ If a is a symbol is S, then a is a regular expression that denotes {a} i.e. the set containing the string a.

(c) $\Rightarrow$(r)* is a regular expression denoting (L(r))*

S.6 (a)

A Lex program mainly consists of three parts: declarations, translation rules and auxiliary procedures, out of which the declaration section includes **variables, manifest constants and regular definition.**

S.7 (a)

The DFA constructed from a regular expression r has space complexity = $O(2^{|r|})$.

S.8 (d)

(i),(ii),(iii) are functions of Lexical analyzer.

(i) $\Rightarrow$primary function of lexical analyzer, whereas (ii),(iii) are secondary functions.

S.9 (d)

(a), (b), (c) are approaches used for lexical analyzer.

S.10 (c)

(a) Regular expressions cannot be used to describe balanced or nested constructs.

(b) Repeating string cannot be described by regular expression and context free grammar.

S.12 (d)

Generate some of the strings that can be derived from the start state and verify they fall into the category covered by option (d).

S.13 (e)

Terminal Table, Literal Table, Identifier Table and Uniform symbol table are the databases associated with lexical analysis.

S.14 (b)

It is the primay task of a lexical analyzer to **scan the whole input program sequentially** and then report errors.

S.18 (c)

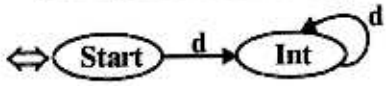
The string 1101 does not belong to the set

(10)*(01)*(00+11)*

S.20 (d)

The given expression is to recognize strings.

State	(Next symbol)/d
Start	Int
Int	Int



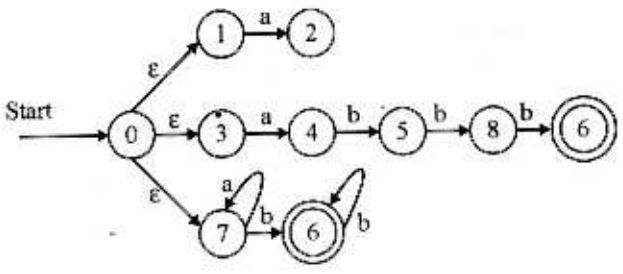
Where d represents a digit.

S.21 (b)

When the string consisting of the single character a is input to the NFA, the start is ε closure ({0}) = {0,2,4,6,7}. On input symbol a there is a transition from 2 to 3 and from 7 to 8. Thus, T is {3,8}. Taking the ε closure of T gives us the next state {0,2,3,4,6,7,8} since none of these nondeterministic states is accepting the algorithm i.e., **x is rejected.**

S.24 (a)

NFA for given Lex program is as follows:



S.25 (a)

Given that, A = (01 + 1)* and B = ((01)*1*)*

$(R_1 + R_2)^* = R_1^* R_2^*$

$(R_1^*)^* = R_1^*$

$\therefore$ A = (01)*(1)*

B = (01)*(1)* $\Rightarrow$ we get **A = B.**

S.26 (a)

If Σ is an alphabet of basic symbol, then a regular definition is a sequence of definition of the form

and for an unsigned number, the valid regular definition is (a).

This definition says an optional – fraction is either a decimal point followed by one or more digits, or it missing (the empty string). An optional – exponent, if it is not missing, is an E followed by an optional + or – sign followed by one or more digits.

S.27 (d)

The given transition diagram is for the pattern >= and >. Its start state is state 0. In state 0 we read the next input character. The edge labeled > from state 0 is to be followed to state 6 if this input character is >. Otherwise, we have failed to recognize either > or <=.

On reaching state 6, we read the next input character. The edge labeled = from state 6 is to be followed to state 7 if this input character is an = otherwise, the edge labeled other indicates that go to state 8. The double circle on state 7 indicates that it is accepting state, a state in which the **token >=** has been found.

S.28 (c)

< > symbol is said to be token without looking at the next input character while compiling a Pascal program.

S.29 (b)

Type checking is normally done during **syntax analysis.**

In practice, there are a number of tasks that might be conducted during parsing such as collecting information about various tokens into the symbol table, performing type checking and other kinds of semantic analysis and generating intermediate code.

S.30 (c)

No. of tokens

$$\underbrace{D0}_{1}\ \underbrace{10}_{2}\ \underbrace{I}_{3}\ \underbrace{=}_{4}\ \underbrace{1.25}_{5}$$

$\therefore$ Numbers of tokens = **5**

S.31 (d)

printf is taken as a single token and rest can be counted. **The result is 21.**

S.32 (a)

Consider each statement separately.

I. If a programming language doesn't permit global variables and no nesting procedures then we can't use the recursion with static storage because there is no stack support for example FORTRAN doesn't support the recursion. So statement is false

II. Statement is true.

III. Recursion in programming languages will be implemented with dynamic storage for example, C language implement recursion with the help of heap, its size also increases because heap stores all the activation record of a recursion. So statement is false.

IV. Statement is false because we can implement recursion either with the help of heap or a stack.

V. Statement is true.

6.3 PARSING TECHNIQUES

LEVEL–1

Q.1 _____ is most useful in describing nested structures such as balanced parentheses, matching begin–end's corresponding if then else's

(a) CFL

(b) Regular Expression

(c) CFG

(d) Both (a) and (c)

Q.2 Consider the following grammar

$S \rightarrow X\ a \mid c$

$X \rightarrow X\ d \mid Se \mid \varepsilon$

the nonterminal S is _____

(a) Immediate left recursive

(b) Left recursive

(c) Both (a) and (b)

(d) Neither (a) nor (b)

Q.3 Elimination of left recursion can be done systematically if the grammar has _____

(a) cycles or ε – productions

(b) no cycles but ε – productions

(c) no cycle or no ε – productions

(d) cycles and no ε – productions

Q.4 Left factoring is _____

(i) a grammar transformation

(ii) useful for producing a grammar suitable for predictive parsing.

(a) (ii)

(b) (i)

(c) neither (i) nor (ii)

(d) both (i) and (ii)

Q.5 Variable prefix property is supported by _____

(i) LR parsing

(ii) LL parsing

(a) (ii)

(b) (i)

(c) neither (i) nor (ii)

(d) both (i) and (ii)

Q.6 Consider the following grammar for expressions

$E \rightarrow EA \mid (E) \mid -E \mid id$

$A \rightarrow + \mid - \mid * \mid / \mid \uparrow$

Then which of the following statement is correct?

(a) The above grammar representation can be converted into operator grammar by replacing terminal values.

(b) The above grammar expression is for an operator grammar

(c) The above grammar expression is not an operator grammar

(d) (a) and (c)

Q.7 Which of the following statement is incorrect about parsing?

(i) Top–down parsing expands the start symbol to the required string needed whereas in Bottom–up parsing reduces the syntax to the start symbol.

(ii) Top–down parsing is implemented using the set of recursive procedures and implementation of bottom–up parsing is done using the stacks and input buffer.

(iii) Backtracking is required for the both bottom–up and top–down parsing implementation.

(a) (i), (ii)

(b) (iii)

(c) (i), (iii)

(d) none of these

Q.8 Consider the language L_1 = {wcw|w is in (a|b)*}, then which of the following statement is correct about L_1?

(i) L_1 is not context free language.

(ii) L_1 consist of all words composed of a repeated string of a's and b's separated by c.

(iii) identifiers recognizing L_1 can be embedded in the problem of checking that identifiers are declared before use.

(a) (i),(ii),(iii)

(b) (i),(iii)

(c) (i),(ii)

(d) (ii),(iii)

Q.9 Which of the following is the disadvantages of an operator precedence parsing?

(i) Operator precedence parsing is hard to handle tokens like the minus sign.

(ii) One cannot always be sure that the parser accepts exactly the desired language.

(iii) Only a small class of grammars can be parsed using operator precedence techniques.

(a) (i), (ii), (iii)

(b) (i), (iii)

(c) (ii), (iii)

(d) (i), (iii)

Q.10 The leaves of the parse tree are labeled by nonterminal or terminals and read from left to right, they are concisely called as _____ of the tree

(a) yield

(b) frontier

(c) Both (a) and (b)

(d) Neither (a) nor (b)

Q.11 A grammar is left recursive if it has a non terminal Z such that there is a derivation _____

(a) $Z \overset{+}{\Rightarrow} aZ$ for some string a

(b) $Z \overset{+}{\Rightarrow} Za$ for some string a

(c) $Z \overset{*}{\Rightarrow} Za$ for some string a

(d) $Z \overset{*}{\Rightarrow} aZ$ for some string a

Q.12 Which of the following is incorrect?

(a) Syntax directed definitions are high level specifications for translations, they hide many implementation details and free the user from having to specify explicitly the order in which translations takes place.

(b) Translate in schemes indicate the order in which semantic rules are to be evaluated, so they allow some implementations details to be shown.

(c) Neither (a) nor (b)

(d) Both (a) and (b)

Q.13 Language which have many types, but the type of every name and expression must be calculated at compile time are

(a) Strongly typed languages

(b) weakly typed language

(c) loosely typed languages

(d) none of these

Q.14 The function of the syntax phase is

(a) to recognize the major constructs of the language and to call the appropriate action routines that will generate the intermediate form or matrix for these constructs.

(b) to build a literal table and an identifier table

(c) to build an uniform symbol table.

(d) to parse the source program into the basic element or tokens of the language.

Q.15 A hardware device that is capable of executing a sequence of instruction is known as
(a) CPU
(b) ALU
(c) CU
(d) processor

Q.16 Dividing a project into segments and smaller units in order to simplify the analysis, design and programming efforts is known as
(a) Modular approach
(b) Top down approach
(c) Bottom up approach
(d) Left right approach

Q.17 Which of the following derivations do a top down parser use while parsing an input string? The input is assumed to be scanned in left to right order.
(a) Left most derivation
(b) Left most derivation traced out in reverse
(c) Right most derivation
(d) Right most derivation traced out in reverse

Q.18 The most powerful parsing method is
(a) LL(1)
(b) Canonical LR
(c) SLR
(d) LALR

Q.19 Which of the following features cannot be captured by CFG
(a) Syntax of if then else statements
(b) syntax of recursive procedures
(c) Whether a variable is declared before its use
(d) Matching nested parenthesis

Q.20 Which one of the following statements is true?
(a) Macro definitions cannot appear within other macro definitions in assembly languages programs.
(b) Overlaying is used to run a program which is longer than the address space of computer
(c) Virtual memory can be used to accommodate a program which is longer than the address space of a computer
(d) It is not possible to write interrupt service routines in a high level language

Q.21 Which of the following statements is true?
(a) SLR parser is more powerful than LALR
(b) LALR parser in more powerful than canonical LR parser.
(c) Canonical LR parser is more powerful than LALR parser.
(d) The parsers SLR, canonical LR, and LALR have the same power

Q.22 A top down parser generates
(a) left most derivation
(b) right most derivation
(c) right most derivation in reverse
(d) left most derivation in reverse

Q.23 A bottom up parser generates
(a) left most derivation
(b) right most derivation
(c) right most derivation in reverse
(d) left most derivations in reverse

Q.24 Shift reduce parsers are
(a) top down parsers
(b) bottom up parsers
(c) may be top down or bottom up parsers
(d) none of the above

LEVEL–2

Q.25 The prefix expression for
A+B*C+C*D**E is
(a) ++A*BC*C**DE
(b) ++A*BC**C*DE
(c) ++A**BCC**DE
(d) None of these

Q.26 On grammar G we eliminate left recursion from it, and then left factoring the resulting grammar that can be parsed by a recursive descent parser then such a parser is called as:
(a) Predictive parser
(b) S–R parser
(c) LL(1) parser
(d) Table driven parser

Q.27 Consider the following step of shift reduce parser

Stack	Input
$$	$

Then, in this configuration the parser

(a) halts

(b) announces successful completion of parsing

(c) Both (a) and (b)

(d) Neither (a) nor (b)

Q.28 Given a context free grammar G, with start symbol S and α is the arbitrary string of grammar symbol which may contain non terminals such that

$S \Rightarrow \alpha, \alpha$ is

(a) sentence of G

(b) CFL of G

(c) sentential form of G

(d) none of these

Q.29 Consider the following grammar for arithmetic expression involving +,−,*,/ and ↑

$E \rightarrow E + E \mid E - E \mid E * E \mid E / E \mid E \uparrow E \mid (E) \mid - E \mid id$

Then which of the following is correct?

(a) The grammar is ambiguous and we cannot disambiguate it

(b) Syntactic variable and syntactic category are two different terminology related to non terminals.

(c) The productions define the ways in which the syntactic categories may be built up from one another and from the terminals.

(d) The grammar is ambiguous and we can disambiguous it by specifying the associativity and precedence of arithmetic operators

Q.30 Consider the following grammar for arithmetic expressions.

$E \rightarrow E + E \mid E * E \mid (E) \mid - E \mid id$

Then −(id+id) is _____

(a) invalid sentence of a grammar

(b) valid sentential form of grammar

(c) valid sentence of grammar

(d) invalid sentential form of a grammar

Q.31 Consider the following language declaration

$L_1 = \{xzx \mid x \text{ is in}(a|b)\}$

$L_2 = \{a^n b^n c^n d^m \mid n \geq 1 \text{ and } m \geq 1\}$

Then which of the statement is correct about L_1 and L_2?

(a) L_1 is not context free but L_2 is context free language

(b) L_1 is context free but L_2 is not context free language

(c) L_1 and L_2 both are not context free language

(d) L_1 and L_2 both are context free language

Q.32 The language $L = \{a^n b^n c^n \mid n \geq 0\}$, is used for an application of typeset text. Typeset text use italics where ordinary typed text uses underlining. In convenient a file of a text destined to be printed on a line printer to text suitable for a photo typesetter, one has to replace underlined works by italics. An underline word is a string of letters followed by an equal number of backspaces and an equal number of underscores. If we regard a as any letter, b as backspace and c as underscore then which of the following statement is correct?

(i) L represents underlined words.

(ii) We cannot use a grammar to describe underlined words.

(iii) We cannot use a parser generating tool based solely on CFG to create a program to convert underlined text.

(a) (i),(ii)

(b) (ii),(iii)

(c) (i),(iii)

(d) (i),(ii),(iii)

Q.33 In a syntax directed definition terminals are assumed to have synthesized attributes only because_____

(a) the definition does not provide any semantic rules for terminals

(b) the definition and declaration does not provide any expression for non terminals.

(c) values for attributes of terminals are usually supplied by the lexical analyzer

(d) None of these

Q.34 Let E be a shifting operation applied to a function f, such that $E(f)=f(x+\beta)$. Then

(a) $E(\alpha f+\beta g) = \alpha E(f) + \beta E(g)$

(b) $E(\alpha f+\beta g) = (\alpha+\beta) E (f+g)$

(c) $E(\alpha f+\beta g) = \alpha E(f+g)$

(d) $E(\alpha f+\beta g) = \alpha \beta (f+b)$

Q.35 If L1 and L2 are context free language and R a regular set, which one of the languages below is not necessarily a context free language

(a) L_1L_2

(b) $L_1 \cap L_2$

(c) $L_1 \cap R$

(d) $L_1 \cup L_2$

Q.36 Given a grammar with rules

$S \rightarrow AaB$, $A \rightarrow a$, $B \rightarrow b$

which of following don't have any syntax tree?

(a) Aab

(b) aAb

(c) aaB

(d) None of the above

Q.37 A context free grammar has rules of the form

$\alpha ::= \beta$, where α is:

(a) a string of non-terminals of length 1

(b) a string of non-terminals of length greater than 1

(c) a string of terminals and non-terminals of any length

(d) none of the above

Q.38 This grammar is

(a) unambiguous

(b) ambiguous

(c) context free

(d) none of the above

Q.39 The above grammar is used to generate all valid arithmetic expressions in a hypothetical language in which.

(a) / associates from the left

(b) – associates from the left

(c) + associates from the left

(d) * associates from the left

Q.40 The above grammar is used to generate all valid arithmetic expressions in a hypothetical language in which

(a) + has the highest precedence

(b) * has the highest precedence

(c) – has the highest precedence

(d) / has the highest precedence

Common Data For Questions 41 to 43:

Let x be a string and let A be a non-terminal. $FIRST_k$ (x) is the set of all leading terminal strings of length k or less, in the strings derivable from x.

$FOLLOW_k$ (A) is the set of all derviable terminal strings of length k or less, that can follow A in some left-most sentential form.

Q.41 Consider the grammar

$E \rightarrow TE'$

$E' \rightarrow +TE' \mid \varepsilon$

$T \rightarrow FT'$

$T' \rightarrow *FT' \mid \varepsilon$

$F \rightarrow (E) \mid id$

$FIRST_1(E)$ will be same as that of

(a) $FIRST_1(T)$

(b) $FIRST_1(F)$

(c) $FIRST_1(T')$

(d) all of the above

Q.42 $FOLLOW_1$ (F) IS

(a) {+, *,), $ }

(b) {+,), $ }

(c) {*,), $ }

(d) {+, (,), * }

Q.43 Which of the following remarks logically follows?

(a) $FIRST_k(\varepsilon) = (\varepsilon)$

(b) If $FOLLOW_k$ (A) contains ε, then A is the start symbol.

(c) If $A \rightarrow w$, is a production in the given grammar G, then $FIRST_k(A)$ contains $FIRST_k(w)$

(d) If $A \rightarrow w$, is a production in the given grammar G, then $FIRST_K(w)$ contains $FIRST_k(A)$

LEVEL–3

Q.44 Consider the following grammar for arithmetic expressions.

$E \rightarrow E + E \mid E * E \mid (E) \mid - E \mid id$

We form a parse tree to derive id + id * id in two distinct ways. Then which of the following statements is correct about it?

(a) A parse tree form is first ways is correct in same sense, in that it reflects the commonly assumed "precedence" of + and*

(b) A parse tree form in second ways is not correct in the precedence of operator + and *

(c) Both (a) and (b)

(d) Neither (a) nor (b)

Q.45 Consider the following grammar for arithmetic expressions

$X \rightarrow X+Z|Z$

$Z \rightarrow Z*Y|Y$

$Y \rightarrow (X)id$

Then after eliminating the immediate left recursions we get

(a) $X \rightarrow ZX'$

$X+ZX'|*YZ'$

$F \rightarrow (X) \mid id$

(b) $X \rightarrow ZX'$

$X' \rightarrow ZX'$

$Z \rightarrow YZ'$

$Z'*YZ'$

$Y \rightarrow id$

(c) $X \rightarrow ZX'$

$X' \rightarrow +ZX|\varepsilon$

$Z \rightarrow YZ'$

$Z' \rightarrow *YZ|\varepsilon$

$Y \rightarrow (X) \mid id$

(d) None of these

Q.46 If $A \rightarrow \alpha\beta_1 \mid \alpha\beta_2$ are two A–productions and the input begins with a non empty string derived from a. Then in such a situation _____ is preferable

(a) Left factoring

(b) Elimination of left recursion

(c) Elimination of ambiguity

(d) None of the above

Q.47 Considering the following grammar

$S \rightarrow iEtS \mid iEtSeS|a$

$E \rightarrow b$

Then after left factoring we get_____

(a) $S \rightarrow iEtS| iEtSeS$

$E \rightarrow b|a$

(b) $S \rightarrow iEtSS'$

$S \rightarrow iEtSeS|a$

$E \rightarrow d$

(c) $S \rightarrow iEtSS'|a$

$S' \rightarrow eS|\varepsilon$

$E \rightarrow b$

(d) None of these

Linked Questions 48 & 49:

Consider the grammar

$S \rightarrow ABSc|Abc$

$BA \rightarrow AB$

$Bb \rightarrow bb$

$Ab \rightarrow ab$

$Aa \rightarrow aa$

Q.48 Which of the following sentences can be derived by this grammar?

(a) abc

(b) aab

(c) abcc

(d) abbc

Q.49 The language generated by above grammar is the set of all strings made up of a,b,c such that

(a) The number of a's, b's and c's will be equal

(b) a's always precede b's

(c) b's always precede c's

(d) the number of a's, b's and c's are same and the a's precede b's which precede c's

GATE QUESTIONS

Q.50 Consider the SLR(1) and LALR(1) parsing tables for a context free grammar. Which of the following statements is/are true? **[GATE 1992]**

(a) The go to part of both tables may be different

(b) The shift entries are identical in both the tables

(c) The reduce entries in the tables may be different

(d) The error entries in the table may be different

Q.51 A shift reduce parser carriers out the actions specified within braces immediately after reducing with the corresponding rule of grammar.

$S \rightarrow xxW\{print\ "1"\}$

$S \rightarrow y\{print\ "2"\}$

$W \rightarrow Sz\{print\ "3"\}$

What is the translation of xxxxyzz using the syntax directed translation scheme described by the above rules? **[GATE 1995]**

[2-Marks]

(a) 23131

(b) 11233

(c) 11231

(d) 33211

Q.52 The pass numbers for each of the following activities

(i) object code generation

(ii) literals added to literal table

(iii) listing printed

(iv) address resolution of local symbols that occur in a two pass assembler respectively are **[GATE 1996]**

[1-Mark]

(a) 1,2,1,2

(b) 2,1,2,1

(c) 2,1,1,2

(d) 1,2,2,2

Q.53 Which of the following statements is true? **[GATE 1998]**

[1-Mark]

(a) SLR parser is more powerful than LALR.

(b) LALR parser is more powerful than Canonical LR parser.

(c) Canonical LR parser is more powerful than LALR parser.

(d) The parsers SLR, Canonical LR, and LALR have the same power

Q.54 The process of assigning load address to the various parts of the program and adjusting the code and data in the program to reflect the assigned addresses is called **[GATE 2001]**

[1-Mark]

(a) Assembly

(b) Parsing

(c) Relocation

(d) Symbol resolution

Q.55 Which of the following statements is false? **[GATE 2001]**

[1-Mark]

(a) An unambiguous grammar has same leftmost and rightmost derivation

(b) An LL (1) parser is top–down parser

(c) LALR is more powerful than SLR (Simple LR)

(d) An ambiguous grammar can never be LR(k) for any k

Q.56 Consider the grammar shown below

$S \rightarrow i\ E\ t\ S\ S' \mid a$

$S \rightarrow eS \mid \varepsilon$

$E \rightarrow b$

In the predictive parse table. M, of this grammar, the entries M[S',ε] and M [S',$] respectively are **[GATE 2003]**

[2-Marks]

(a) $\{S' \rightarrow eS\}$ and $\{S' \rightarrow \varepsilon\}$

(b) $\{S \rightarrow eS\}$ and { }

(c) $\{S' \rightarrow \varepsilon\}$ and $\{S' \rightarrow \varepsilon\}$

(d) $\{S' \rightarrow eS, S' \rightarrow \varepsilon\}$ and $\{S' \rightarrow \varepsilon\}$

Q.57 Consider the translation scheme shown below

$S \rightarrow TR$

$R \rightarrow +T$ {print ('+');}$R \mid \varepsilon$

$T \rightarrow$ **num** {print (num. val);}

Here **num** is a token that represents an integer and **num.val** represents the corresponding integer value. For an input string '9+5+2', this translation scheme will print **[GATE 2003]**

[2-Marks]

(a) 9 + 5 + 2

(b) 9 5 + 2 +

(c) 9 5 2 + +

(d) + + 9 5 2

Q.58 Which of the following suffices to convert an arbitrary CFG to an LL(1) grammar?

[GATE 2003]

[1-Mark]

(a) Removing left recursion alone

(b) Factoring the grammar alone

(c) Removing left recursion and factoring the grammar

(d) None of the above

Q.59 In a bottom-up evaluation of a syntax directed definition, inherited attributes can**[GATE 2003]**

[1-Mark]

(a) always be evaluated

(b) be evaluated only if the definition is L-attributed

(c) be evaluated only if the definition has synthesized attributes

(d) never be evaluated

Q.60 Assume that the SLR parser for a grammar G has n_1 states and the LALR parser for G has n_2 states. The relationship between n_1 and n_2 is

[GATE 2003]

[1-Mark]

(a) n_1 is necessarily less than n_2

(b) n_1 is necessarily equal to n_2

(c) n_1 is necessarily greater than n_2

(d) None of the above

Q.61 Consider the grammar shown below:

$S \rightarrow CC$

$C \rightarrow cC \mid d$

The grammar is **[GATE 2003]**

[2-Marks]

(a) LL (1)

(b) SLR (1) but not LL (1)

(c) LALR (1) but not SLR (1)

(d) LR (1) but not LALR (1)

Q.62 Which of the following grammar rules violate the requirements of an operator grammar? P, Q, R are non terminals and r, s, t are terminals

[GATE 2004]

[1-Mark]

(i) $P \rightarrow QR$ (ii) $P \rightarrow QsR$

(iii) $P \rightarrow \varepsilon$ (iv) $P \rightarrow QtRr$

(a) (i) only

(b) (i) and (iii) only

(c) (ii) and (iii) only

(d) (iii) and (iv) only

Q.63 Consider the grammar with the following translation rules and E as the start symbol

$E \rightarrow E_1$ # T	{E.value= E_1. value * T. value}
| T	{E.value = T.value}
$T \rightarrow T_1$ & F	{T.value=T_1.value*F.value}
| F	{T.value = F.Value}
F → num	{F. value = num. value}

compute E. value for the root of the parse tree for the expression.

2#3&5#6&4. **[GATE 2004]**

[2-Marks]

(a) 200

(b) 180

(c) 160

(c) 40

Q.64 The grammar A→AA|A|ε is not suitable for predictive parsing because the grammar is

[GATE 2005]

[1-Mark]

(a) ambiguous

(b) Left recursive

(c) right recursive

(d) an operator grammar

Q.65 Consider the grammar E → E+n|E×n|n

For a sentence n+n×n, the handles in the right-sentential form of the reduction are

[GATE 2005]

[2-Marks]

(a) n, E + n and E + n × n

(b) n, E + n and E + E × n

(c) n, n + n and n + n × n

(d) n, E + n and E × n

Q.66 Consider the grammar

S→(S) | a

Let the number states in SLR(1), LR(1) and LALR(1) parsers for the grammar be n_1, n_2 and n_3 respectively. The following relationship holds good

[GATE 2005]

[2-Marks]

(a) $n_1 < n_2 < n_3$

(b) $n_1 = n_3 < n_2$

(c) $n_1 = n_2 = n_3$

(d) $n_1 \geq n_3 \geq n_2$

Common Data For Questions 67 & 68:

Consider the following expression grammar. The semantic rules for expression evaluation are stated next to each grammar production.

E→number	E.val = number.val
|E'+'E	$E^{(1)}.val = E^{(2)}.val + E^{(3)}.val$
|E'×'E	$E^{(1)}.val = E^{(2)}.val \times E^{(3)}.val$

Q.67 The above grammar and the semantic rules are fed to a *yacc* tool (which is an LALR(1) parser generator) for parsing and evaluating arithmetic expressions. Which one of the following is true about the action of *yacc* for the given grammar?

[GATE 2005]

[2-Marks]

(a) It detects recursion and eliminates recursion.

(b) It detects, reduce-reduce conflicts, and resolve.

(c) It detects shift reduce conflict, and resolves the conflict in favor of a shift over a reduce action.

(d) It detects shift reduce conflict, and resolves the conflict in favor of a reduce over a shift action.

Q.68 Assume the conflicts in question (67) are resolved and an LALR(1) parser is generated for parsing arithmetic expression as per the given grammar. Consider an expression 3 × 2 + 1. What precedence and associativity properties does the generated parser realize?

[GATE 2005]

[2-Marks]

(a) Equal precedence and left associativity; expression is evaluated to 7.

(b) Equal precedence and right associativity; expression is evaluated to 9.

(c) Precedence of '×' is higher than that of '+'; and both operators are left associative; expression is evaluated to 7.

(d) precedence of '+' is higher than that of '×' and both operators are left associative; expression is evaluated to 9.

Q.69 Consider the following grammar.

S→S*E

S→E

E→F + E

E→F

F→ id

Consider the following LR(0) items corresponding to the grammar above.

[GATE 2006]

[1-Mark]

(i) S→S* E

(ii) E → F. + E

(iii) E→F+.E

Given the items above, which two of them will appear in the same set in the canonical sets of items for the grammar?

(a) (i) and (ii)

(b) (ii) and (iii)

(c) (i) and (iii)

(d) None of these

Linked Questions 70 & 71:

Q.70 Which one of the following grammars generates the language $L = (a^i b^j | i \neq j)$? **[GATE 2006]**

[2-Marks]

(a) S→AC|CB

C→aCb|a|b

A→aA|ε

B→Bb|ε

(b) S→aS|Sb|a|b

(c) S→AC|CB

C→aCb|ε

A→aA|ε

B→Bb|ε

(d) S→AC|CB

C→aCb|ε

A→aA|a

B→ Bb|b

Q.71 In the correct grammar above, what is the length of the derivation (number of steps starting from S) to generate the string $a^l b^m$ with l≠m?

[GATE 2006]

[2-Marks]

(a) max(l, m)+2

(b) l + m + 2

(c) l + m + 3

(d) max (l,m) + 3

Q.72 Consider the following grammar

S→FR

R→*S|ε

F→id

In the predictive parser table, M, of the grammar the entries M [S,id] and M[R,$] respectively. **[GATE 2006]**

[2-Marks]

(a) {S→FR} and {R→ε}

(b) {S→FR} and {}

(c) {S→FR} and {R→*S}

(d) {F→id} and {R→ε}

Q.73 Which one of the following is a top down parser? **[GATE 2007]**

[1-Mark]

(a) Recursive descent parser

(b) Operator precedence parser

(c) An LR(k) parser

(d) An LALR(k) parser

Q.74 Consider the following two statements:

P: Every regular grammar is LL(1)

Q: Every regular set has LR(1) grammar

Which of the following TRUE?

[GATE 2007]

[2-Marks]

(a) Both P and Q are true

(b) P is true and Q is false

(c) P is false and Q is true

(d) Both P and Q are false

Q.75 Consider the grammar with non terminals $N = \{S,C,S_1\}$ terminals $T = \{a,b,i,t,e\}$, with S as the start symbol, and the following set of rules

$S \rightarrow iCtSS_1|a$

$S_1 \rightarrow eS|\varepsilon$

$C \rightarrow b$

The grammar is Not LL(1) because:

[GATE 2007]

[2-Marks]

(a) It is left recursive

(b) It is right recursive

(c) It is ambiguous

(d) It is not context free

Common Data For Questions 76 & 77:

Consider the CFG with {S,A,B} as the non-terminal alphabet, {a,b} as the terminal alphabet, S as the start symbol and the following set of production rules

$S \rightarrow bA$	$S \rightarrow aB$
$A \rightarrow a$	$B \rightarrow b$
$A \rightarrow aS$	$B \rightarrow bS$
$S \rightarrow bAA$	$B \rightarrow aBB$

Q.76 Which of the following strings is generated by the grammar? **[GATE 2007]**

[2-Marks]

(a) aaaabb

(b) aabbbb

(c) aabbab

(d) abbbba

Q.77 How many derivation trees are there?

[GATE 2007]

[2-Marks]

(a) 1

(b) 2

(c) 3

(d) 4

Q.78 Which of the following describes a handle (as applicable to LR parsing) appropriately?

[GATE 2008]

[1-Mark]

(a) It is the position in a sentential form where the next shift or reduce operation will occur

(b) It is a non terminal whose production will be used for reduction in the next step

(c) It is production that may be used for reduction in a future step along with a position in the sentential form where the next shift or reduce operation will occur

(d) It is the production p that will be used for reduction in the next step along with a position in the sentential form where the right hand side of the production may be found.

Q.79 An LALR (1) parser for a grammar G can have shift reduce (S–R) conflicts if and only if

[GATE 2008]

[2-Marks]

(a) the SLR(1) parser for G has S–R conflicts

(b) the LR(1) parser for G has S–R conflicts

(c) the LR(0) parser for G has S–R conflicts

(d) the LALR(1) pafser for G has reduce-reduce conflicts.

ANSWER KEY

1	c	2	b	3	c	4	d	5	d
6	d	7	b	8	a	9	a	10	c
11	b	12	d	13	a	14	c	15	d
16	a	17	a	18	d	19	d	20	b
21	c	22	a	23	b	24	b	25	a
26	a	27	a	28	c	29	d	30	c
31	c	32	d	33	a	34	d	35	a
36	b	37	a	38	a,c	39	a,c,d	40	a
41	a,b	42	a	43	a,b,c	44	c	45	c
46	a	47	c	48	a	49	d	50	c
51	a	52	b	53	c	54	c	55	a
56	d	57	b	58	c	59	c	60	b
61	d	62	b	63	c	64	a	65	d
66	b	67	c	68	c	69	c	70	d
71	a	72	a	73	a	74	a	75	a
76	c	77	b	78	a	79	b		

SOLUTIONS

S.1 (c)

Context free grammar (CFG) is most useful in describing nested structures. We cannot use regular expression for it.

S.2 (b)

The non terminal S is left recursive because $S \Rightarrow Xa \Rightarrow Sea$, but it is not immediate left recursive.

S.3 (c)

We eliminate left-recursion in three steps.

(a) eliminate $\in$-productions (impossible to generate $\in$!)

(b) eliminate cycles ($A^+ \Rightarrow A$)

(c) eliminate left-recursion.

S.4 (d)

Left factoring is **a grammar transformation and useful for producing a grammar suitable for predictive parsing.**

S.5 (d)

LL and LR parsing methods, detect an error as soon as possible. They have the variable prefix property, meaning that they detect that an error has occurred as soon as they see a prefix of the input that is not a prefix of any string in the language.

S.6 (d)

(a) $\Rightarrow$ By replacing the terminals value, the grammar expression can be converted into operator grammar

(c) $\Rightarrow$ The given representation is not an operator grammar (by definition)

S.7 (b)

(i) and (ii) are correct statement about top–down and bottom–up parsing techniques whereas (iii) no backtracking is required for top–down parsing.

S.8 (a)

(i),(ii),(iii) are correct statement about L_1 which is not context free

S.9 (a)

Disadvantages of operator Precedence Parsing

(i) It cannot handle the unary minus (the lexical analyzer should handle the unary minus).

(ii) It has small class of grammars.

(iii) Difficult to decide which language is recognized by the grammar

S.19 (d)

It is because, it is equivalent to recognizing wcw, where the first w is the declaration and the second is its use. wcw is not a CFG.

S.20 (b)

All (a),(c) and (d) are false. The difference between (b) and (c) is virtual.

For example 8086 processor has 32 bits address capacity (32 pins for address), hence it is capable of accessing $2^{32} = 2^2 \times 2^{30}$ =4096 MB of main memory. But present day computer have at most 32 MB of RAM. The address space of 8086 processor is 4 GB. And hence virtual memory can be at most 4 GB. If a program is of 5GB, we have to use only overlaying techniques.

S.21 (c)

LR is powerful than LALR parsers and LALR are powerful than SLR parsers.

S.24 (b)

Any shift-reduce parser typically works by shifting entries onto the stack. If a handle is found on the top of the stack, it is popped and replaced by the corresponding left hand side of the production. If ultimately we have only the starting non-terminal on the stack, when there are no more tokens to be scanned the parsing will be successful. So, it is **bottom up.**

S.26 (a)

By carefully writing a grammar, eliminating left recursion from it and left factoring the resulting grammar, we can obtain a grammar that can be parsed by a recursive descent parser that needs no backtracking. **This is called predictive parser.**

S.27 (a)

The shift reduce parser operates by shifting zero or more input symbols onto the stack until handle is on top of the stack. The parser then reduces this handle to the left side of the appropriate production. The parser repeats this cycle until it has detected an error or until the stack contains the start symbol and the input is empty. For this configuration **the parser halts and announces successful completion of parsing.**

S.28 (c)

With $S \Rightarrow \alpha$, where α is non terminal then we say α is **a sentential form of G.**

S.29 (d)

The grammar,

$E \rightarrow E + E \mid E - E \mid E * E \mid E/ E \mid E \uparrow E \mid (E) \mid - E \mid id$

is an ambiguous grammar, we can eliminate ambiguity by specifying the associativity and precedence of arithmetic operators.

S.30 (c)

$E \rightarrow E + E \mid E * E \mid (E) \mid - E \mid id$

$E \Rightarrow -E \Rightarrow -(E) \Rightarrow -(E+E)$

$\Rightarrow -(id+E) \Rightarrow -(id+id)$

The string $-(id+E) \Rightarrow -(id+id)$

The string – (id+id) is a valid sentence of the given grammar.

S.31 (c)

L_1= {xzx|x is in (a|b)*} is not a context free language i.e. L_1 consists of all words composed of a repeated strings of a' s and b's separated by z.

L_2 = {$a^n b^m c^n d^m$|n ≥1 and m ≥1} is not context free i.e. L_2 consists of words in a*b*c*d* such that the number of a 's and c's are equal and the number of b's and d's are equal.

S.32 (d)

$L=\{a^n b^n c^n | n \geq 0\}$ is a language i.e. strings in a*b*c with equal number of a's,b's and c's is not context free. The typeset problem given in the question follows (i), (ii) and (iii).

S.33 (a)

In a syntax directed definition, terminals are assumed to have synthesized attributes only

because the definition does not provide any semantic rules for terminals.

S.36 (b)

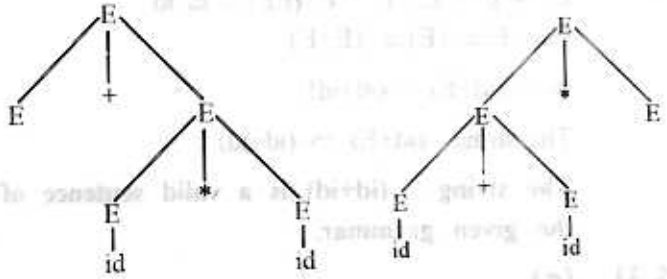
aAb is not a sentential term of grammar.

S.37 (a)

In CFG, all rules must have **exactly one non-terminal on left hand side of the rule.**

S.44 (c)

First form	second form
$E \Rightarrow E + E$	$E \Rightarrow E * E$
$\Rightarrow id + E$	$\Rightarrow E + E * E$
$\Rightarrow id + E * E$	$\Rightarrow id + E * E$
$\Rightarrow id + id * E$	$\Rightarrow id + id * E$
$\Rightarrow id + id * id$	$\Rightarrow id + id * id$



S.45 (c)

A grammar is left recursive if it has nonterminal A such that there is a derivation $A \overset{+}{\Rightarrow} A\alpha$ for some string α. Applying rules to eliminate left recursive we get (C).

$X \rightarrow ZX'$

$X' \rightarrow +ZX|\varepsilon$

$Z \rightarrow YZ'$

$Z' \rightarrow *YZ|\varepsilon$

$Y \rightarrow (X) \mid id$

S.46 (a)

Left factoring is a grammar transformation that is useful for production of grammar suitable for predictive parsing. In given situation, we do not know whether to expand A to $\alpha\beta_1$ or $\alpha\beta_2$. However, we may defer the decision by expanding A to $\alpha A'$. Then after seeing the input derived form α, we expand A' to β_1 or β_2 i.e. left factored, the original productions becomes

$A \rightarrow \alpha A'$

$A \rightarrow \beta_1|\beta_2$

S.47 (c)

Given grammar is,

$S \rightarrow iEtS|iEtSeS|a$

$E \rightarrow b$

After applying algorithm for left factoring a grammar we get

$S \rightarrow iEtSS'|a$

$S \rightarrow eS|\varepsilon$

$E \rightarrow b$

S.48 (a)

Abc can be derived as,

$S \rightarrow \underline{Ab}c$

$\rightarrow abc$ using $Ab \rightarrow ab$

As we see, any production from the start state has to end in C. So option (b) is impossible (c) and (d) are also not possible with given rules

S.49 (d)

Generate some of the strings that can be derived from start state and verify they fall into the category covered by option (d).

S.51 (a)

x x x x y z z		
x x x x S z z	S→y	2
x x x x W z	W→Sz	2 3
x x S z	S → x x W	2 3 1
x x W	W → Sz	2 3 1 3
S	S → x x W	2 3 1 3 1

S.52 (b)

The pass numbers for each of the following activities are as follows:

(i) Object code generation – **Pass 2**

(ii) literals added to literal table – **Pass 1**

(iii) listing printed – **Pass 2**

(iv) Address resolution of local symbols that occur in a two pass assembler respectively are **pass–1.**

S.53 (c)

LR is powerful than LALR parsers and LALR are powerful than SLR parsers.

S.54 (c)

Adjusting all address dependent locations, such as constants, to correspond to the allocated space is called as **relocation.**

S.56 (d)

The grammar is

$S \rightarrow iEtSS' \mid a$

$S' \rightarrow eS \mid \varepsilon$

$E \rightarrow b$

The predictive parser table M is

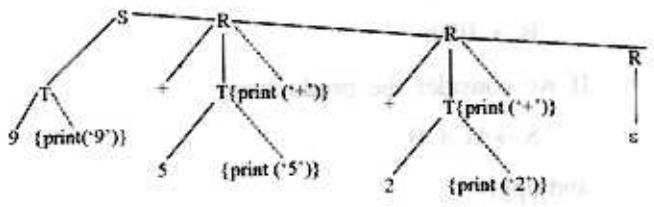
Non terminal	a	b	e	i	t	$
S	S→a			S→iEtSS'		
S'			S'→eS			S'→ε
E		E→b				

So **M[S',ε] = {S'→ε,S'→eS}**

M[S',$] = S→ε

S.57 (b)

For the input '9+5+2' the translation scheme is shown below.



S.58 (c)

To convert an arbitrary CFG to an LL(1) grammar we must **remove left recursion and left factoring** if we can't remove these then grammar becomes ambiguous and LL(1) parsers can't parse it.

S.59 (c)

A syntax directed definition is a generalization of a context free grammar is which each grammar or symbol has been associated with two subsets called the synthesized and inherited attributes of that grammar symbol.

So **inherited attributes can be evaluated only if the definition has synthesized attributes.**

S.60 (b)

SLR parser has n_1 states for a grammar G LALR parser has n_2 states for a grammar G. The states of SLR and LALR parser are the states of corresponding states in a deterministic finite automata which recognizes the variable prefixes and both deterministic finite automata contains the equal number of states so $\mathbf{n_1 = n_2}$.

S.61 (d)

Consider the grammar

$S \rightarrow CC$

$C \rightarrow cC|d$

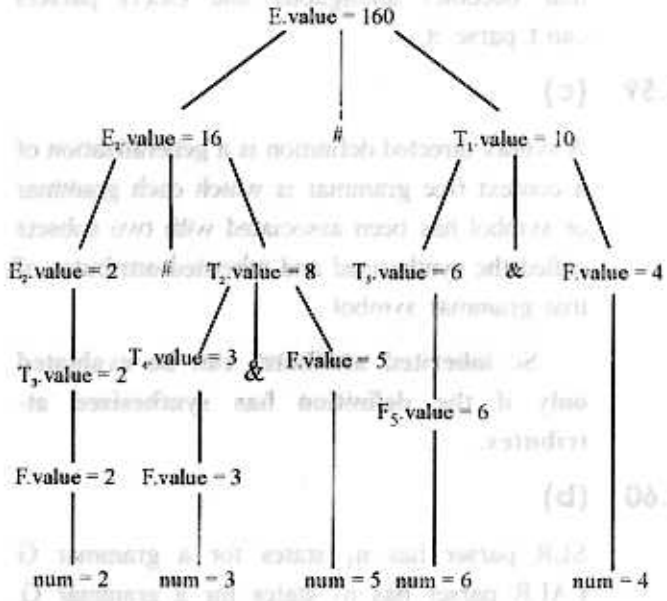
The given grammar is LR(1) grammar. Every SLR(1) grammar is an LR(1) grammar but **not every LALR(1) grammar is LR(1) grammar.** The given grammar is canonical LR(1) grammar and **every canonical LR(1) grammar is LR(1) grammar.**

S.62 (b)

In operator grammar the grammar has the property that no production side is ε or it doesn't contains adjacent nonterminals. So **P→QR and P→ε violate the requirements of an operator grammar.**

S.63 (c)

First we have to construct the parse tree.



S.64 (a)

The given grammar is ambiguous, that is the reason that it is not suitable for predictive parsing.

S.65 (d)

n→E + n |E × n | n

Input String n + n × n

⇒ n + n × n

⇒ E + n × n reduction E→n

⇒ E × n reduction E→E+n

⇒ E reduction E→E× n

So the reduction are **n, E + n, E × n**

S→(S) | a

S.66 (b)

Parsers	No. of States
SLR (1)	n_1
LR(1)	n_2
LALR(1)	n_3

The number of states of deterministic finite automata in SLR(1) and LALR(1) parsers are equal so $n_1 = n_3$. The number of states of deterministic finite automata in LR(1) or canonical LR is greater than number of states of deterministic finite automata of SLR(1) and LALR(1)

So $n_1 = n_3 < n_2$.

S.67 (c)

E→ number |E'+'E|E'×'E

shift reduce conflict, and resolves the conflict in favor of a shift over a reduce action. Consider the following configuration where shift-reduce conflict occurs.

Stack	Input
...E+Ex	E...$

S.68 (c)

With lookahead we should prefer to shift because the look ahead has higher precedence than × over + and both operators are left associative. So expression 3 × 2 + 1 = 6 + 1 = 7 will be evaluated.

S.69 (c)

S→S *•E

E→F•+E

E→F+•E

So **S→S*•E and E→F +•E** belong to the same set of canonical LR (0) item because in both productions the dot is left to E.

S.70 (d)

S→AC|CB

C→aCb|ε

A→aA|a

B→ Bb|b

If we consider the production

S→AC|CB

and apply

C→ε then

S ⇒ A|B

If we will apply either A or B

production in S⇒A|B then it generates a^nb^n or a^nb^0. So

$L = \{a^ib^j | i \neq j\}$

Consider the production S⇒AC|CB if we will apply the production C⇒aCb then it generates a^nb but when we apply the either A or B production then it generates either number of a's or b's is greater so it also generates $L = \{a^ib^j | i \neq j\}$.

S.71 (a)

If grammar generates a^nb^0 then the length is max(1,x) if it generates a^lb^m. initially we include two extra derivation so the total length is **max(1,m) + 2.**

S.72 (a)

Construct predictive parser table as follows

Nonterminal	id	*	$
S	S→FR		
R		R→*S	R→ε
F	F→id		

So **M[S,id] = {S→FR}**

and **M[R,$] = {R→ε}**

S.73 (a)

Predictive parser and **recursive descent parser are example of top down parser.**

S.74 (a)

Regular grammar is well recognized by LL(1) parsers and LR(1) parser is stronger and more powerful than LL(1) so regular grammar is also accepted by LR(1) parser. **Every regular set has LR(1) grammar.** So both statement are correct.

S.75 (a)

$S \rightarrow iC\ t\ SS_1 \mid a$

$S_1 \rightarrow eS \mid \varepsilon$

$C \rightarrow b$

$S \overset{1}{\Rightarrow} ibtSS_1$

$S \overset{2}{\Rightarrow} ibtSS_1$ [apply C→b]

After second step of derivation i, b,t are terminals but the left nonterminal symbol is S so grammar is left recursive after step 2. It should be noted that in some cases we can't observe the left recursion in initial step of derivation but after some step of derivation we can see **the left recursion.**

S.76 (c)

$S \underset{lm}{\Rightarrow} aB$

$\underset{lm}{\Rightarrow} aaBB$ (B → aBB)

$\underset{lm}{\Rightarrow} aabB$ (B → b)

$\underset{lm}{\Rightarrow} aabbS$ (B → bS)

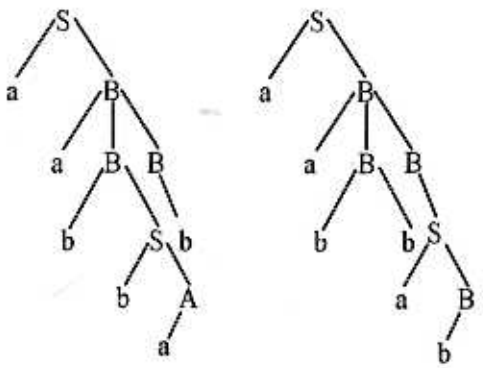
$\underset{lm}{\Rightarrow} aabbaB$ (S → aB)

$\underset{lm}{\Rightarrow} aabbab$ (B → b)

So, **aaabbab** is generated by the grammar

S.77 (b)

There are two parse tree for the string aabbab one from leftmost derivation and one from right most derivation.



S.79 (b)

LALR(1) parser uses the LR(1) items So LALR(1) parser for a grammar G can have S–R conflict if and only **if the LR(1) parser for G has S–R conflicts.**

6.4 SYNTAX DIRECTED TRANSLATION

LEVEL–1

Q.1 Compile time errors do not include

(a) lexical errors

(b) syntactic errors

(c) semantic errors

(d) None of these

Q.2 F:PROCEDURE OPTIONS (MAIN)

is an example of _____ error

(a) Deletion error

(b) Insertion error

(c) Transposition error

(d) Replacement error

Q.3 The language which are having many types, but the type of every name and expression must be calculable at compile time is called _____

(a) Strongly typed

(b) Weakly typed

(c) Dynamic typed

(d) None of these

Q.4 The range checking for certain values, array subscripts and case statement selectors are examples of _____

(a) Semantic error

(b) Dynamic error

(c) Syntactics error

(d) None of these

Q.5 Which of the following is correct about syntax directed Translation?

(i) Evaluation of the semantic rules may generate code, save information in a symbol table, issue error message or perform any other activities.

(ii) The translation of the token stream is the result obtained by evaluating the semantic rules

(a) (i)

(b) (ii)

(c) (i) and (ii)

(d) Neither (i) nor (ii)

Q.6 Which of the following is incorrect?

(a) In a syntax directed definition, terminals are assumed to have synthesized attributes only.

(b) Values for attributes of terminals are usually supplied by the lexical analyzer.

(c) The start symbol is assumed not to have any inherited attributes, less otherwise.

(d) None of these

Q.7 Which of the following is treated as restriction for translation routines that are invoked during parsing and routines must live?

(i) A grammar that is suitable for parsing may not reflect the natural hierarchical structure of the constructs in the language.

(ii) The parsing method constrains the ordering which nodes in a parse tree are considered.

(a) (i)

(b) (ii)

(c) Both (i) and (ii)

(d) Neither (i) nor (ii)

Q.8 Which of the following is incorrect about an error detection?

(i) A simple compiler may stop all activities other than lexical and syntactic analysis after the detection of the first error.

(ii) A more complex compiler may attempt to repair the error, that is transform the erroneous input into a similar but legal input on which normal processing can be resumed.

(iii) No compiler can do true corrections.

(a) (i),(ii),(iii)

(b) (i),(ii)

(c) (ii),(iii)

(d) None of these

Q.9 Which of the following is correct example about good reporting errors methods?

(i) missing right parenthesis in line 5

(ii) cryptic error "OH17"

(iii) ZAP not declared in procedure BLAH

(iv) missing declaration

(a) (i),(ii),(iii),(iv)

(b) (ii)

(c) (i) and (ii)

(d) Neither (i) nor (ii)

Q.10 Which of the following is correct?

(a) L-attributed definitions are class of syntax directed definitions whose attributes can always be evaluated in depth first order

(b) Every S attributed definition is L attributed

(c) A syntax directed definition is L attributed if each inherited attributed of x_j, $1 \le j \le n$, on the right side of $A \rightarrow X_1 X_2 \ldots\ldots X_n$, depends only on the inherited attributed of A

(d) All of the above

Q.11 A syntax directed definition that uses synthesized attributes exclusively is said to be_____

(a) annotating definition

(b) S-attributed definition

(c) grammar attributed definition

(d) L-attributed definition

Q.12 Which of the following can be corrected by using minimum hamming distance method?

(a) Syntactic Error

(b) Semantic Error

(c) Data flow optimization

(d) None of these

Q.13 A basic block can be analyzed by

(a) a DAG

(b) flow graph

(c) a graph with cycles

(d) none of these

Q.14 Advantage of using assembly language rather than machine language is that

(a) it is mnemonic and easy to read

(b) addresses are symbolic, not absolute

(c) introduction of data to program is easier

(d) all of these

Q.15 Uniform symbol table

(a) contains all constants in the program.

(b) is a permanent table of decision rules in the form of patterns for matching with the uniform symbol table to discover syntactic structure.

(c) consists of full or partial list of the token's as they appear in the program. Created by lexical analysis and used for syntax analysis and interpretation.

(d) a permanent table which list all key words and special symbols of the language in symbolic form.

Q.16 Which is a permanent database in the general model of compiler

(a) Literal table

(b) Identifier table

(c) Terminal table

(d) Reductions

Q.17 A self relocating program is one which

(a) cannot be made to execute in any area of storage other than the one designated for it at the time of its coding or translation

(b) consists of a program and relevant information for its relocation

(c) can itself perform the relocation of its address sensitive portions

(d) all of these

Q.18 Relocation bits used by relocating loader are specified by

(a) Relocating loader itself

(b) Linker

(c) Assembler

(d) Macro processor

Q.19 The value of K, in LR (K) cannot be

(a) 0

(b) 1

(c) 2

(d) none of these

Q.20 The best way to compare the different implementation of symbol table is to compare the time required to

(a) add a new name

(b) make an inquiry

(c) add a new name and make an inquiry

(d) all of these

Q.21 ud- chaining is useful for

(a) determining whether a particular definition is used anywhere or not

(b) constant folding

(c) checking whether a variable is used, without prior assignment

(d) All of the above

Q.22 du-chaining

(a) stands for use definition chaining

(b) is useful for copy propagation removal

(c) is useful for induction variable removal

(d) none of the above

Q.23 Back-patching is useful for handling

(a) conditional jumps

(b) unconditional jumps

(c) backward references

(d) forward references

LEVEL–2

Q.24 For evaluating semantic rules we use _____

(a) Parse tree methods

(b) Rule based method

(c) Oblivious methods

(d) All of the above

Q.25 Undeclared and multiple declared identifiers are examples of _____

(a) Semantic Errors

(b) Declaration errors

(c) Transposition errors

(d) None of these

Q.26 A parse tree for an S–attributed definition can always be annotated by_____

(a) evaluating the semantic rules for the attributes at each node bottom up, from the leaves to the root.

(b) evaluating the semantic rules for the attributes at each node top –down, from the leaves to the root.

(c) evaluating the semantic rules for the attributes at each node top down, from the root to leaves.

(d) evaluating the semantic rules for the attributes at each node bottom up, from the root to leaves.

Q.27 Conceptually, with_____ we perform the following actions.

Input String → Parse tree → Dependency graph → Evaluation order For semantic rules

(a) Syntax directed definition

(b) Syntax directed translation

(c) Both (a) and (b)

(d) Neither (a) nor (b)

Q.28 The process of computing the attribute values at the nodes in parse tree is called _____ parse tree.

(a) annotating

(b) decorating

(c) Both (a) and (b)

(d) Neither (a) nor (b)

Q.29 Consider the following syntax directed definition representation.

Production	Semantic Rules
$L \rightarrow En$	Print(E.val)
$E \rightarrow E_1 + T$	$E.val := E_1.val + T.val$
$E \rightarrow T$	E.val := T.val
$T \rightarrow T_1 * F$	$T.val : T_1.val \times F.val$
$T \rightarrow F$	T.val = F.val
$F \rightarrow (E)$	F.val := E.val
$F \rightarrow digit$	F.val = digit.lexval

The above representation has _____ synthesized attribute.

(a) 2

(b) 3

(c) 5

(d) 6

Q.30 What will be the locally optimized code for the statement.

X= (A–B)*C–D/(A–B)?

(a) no change, already localized

(b) there is nothing like locally optimized

(c) there is syntax error

(d) None of these

Common Data For Questions 31 to 33:

Consider the given arithmetic expression grammar with error actions inserted into the table given below.

$E \rightarrow E + E \mid E * E \mid (E) \mid id$

State on top of Stack	Action						Goto
	id	+	*	(	)	$	E
0	S3	e1	e1	S2	e2	e1	1
1	e3	S4	S5	e3	e2	acc	
2	S3	e1	e1	S2	e2	e1	6
3	e3	r4	r4	e3	r4	r4	
4	S3	e1	e1	S2	e2	e1	7
5	S3	e1	e1	S2	e2	e1	8
6	e3	S4	S5	e3	S9	e4	
7	e3	r1	S5	e3	r1	r1	
8	e3	r2	r2	e3	r2	r2	
9	e3	r3	r3	e3	r3	r3	

Now answer the following questions regarding error handling.

Q.31 e1 is called when_____

(a) an operand, beginning with either an id or left parenthesis is expected but an operator, right or the end of input is found.

(b) an operand, end with either an id or a left parenthesis is expected.

(c) we have a right parenthesis on top of the stack but not on the input

(d) None of these

Q.32 e2 is called when_____

(a) we have left parenthesis on top of the stack but not on the input

(b) we have right parenthesis on top of the stack but not on the input

(c) an operand, beginning with either an id or a left parenthesis is expected, but an operator, right parenthesis or the end of input is found.

(d) None of these

Q.33 e3 is called from states when expected_____

(a) operator and an id or right parenthesis is found

(b) id or left parenthesis is found

(c) operator and left parenthesis is found

(d) None of these

Q.34 In case of symbol table, we need to be able to _____

(i) determine whether a given name is in the table

(ii) add a new name to the table

(iii) add new information for a given name

(iv) delete a name or group of names from the table

(a) (i),(ii),(iii),(iv)

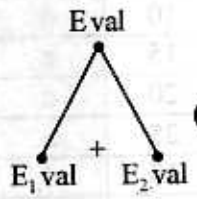
(b) (ii),(iii),(v)

(c) (iii),(v)

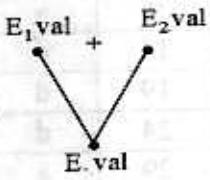
(d) (i),(iii),(iv)

Q.35 Consider the production $E \rightarrow E_1 + E_2$ then, the dependency graph for it is _____

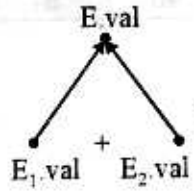
(a)



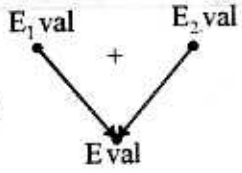
(b)



(c)



(d)



Q.36 Match the following:

Group I		Group II	
(i)	MIN (A,2*(3+B))	(a)	Insertion errors
(ii)	DO 10, I =1,100	(b)	Replacement error
(iii)	I =1 J=2;	(c)	Transposition errors
		(d)	Deletion error

(a) (i)–(d),(ii)–(a),(iii)-(b)

(b) (i)–(b),(ii)–b,(iii)–(c)

(c) (i)-(b),(ii)-(c),(iii)-(a)

(d) (i)-(d),(ii)-(b),(iii)-(d)

Q.37 In symbol table implementation using List, if the symbol table contains n names and m inquiries then the total work is _____,(by assuming c, as a constant representing the time necessary for a few machine operations).

(a) c(n+m)

(b) cn(n+m)

(c) (n+m)

(d) (c+n)m

Q.38 In a bottom up evaluation of a syntax directed definition, inherited attributes can

(a) always be evaluated

(b) be evaluated only if the definition is L–attributed

(c) be evaluated only if the definition has synthesized attributes

(d) never be evaluated

GATE QUESTIONS

Q.39 Generation of intermediate code based on an abstract machine model is useful in compilers because **[GATE 1994]**

(a) it makes implementation of lexical analysis and syntax analysis easier

(b) Syntax–directed translations can be written for intermediate code generation

(c) it enhances the portability of the front end of the compiler

(d) it is not possible to generate code for real machines directly from high level language programs

Q.40 A linker is given object modules for a set of program that were compiled separately. What information need to be included in an object module? **[GATE 1995]**

(a) Object code

(b) Relocation bits

(c) Names and locations of all external symbols defined in the object module

(d) Absolute addresses of internal symbols.

Q.41 Which of the following statements is FALSE? **[GATE 2003]**

(a) In statically typed languages, each variable in a program has a fixed type

(b) In untyped languages, values do not have any types.

(c) In dynamically typed languages, variables have no types

(d) In all statically typed languages, each variable in a program is associated with values of only single type during the execution of the program

ANSWER KEY

1	b	2	c	3	a	4	b	5	c
6	b	7	c	8	d	9	a	10	d
11	b	12	a	13	a	14	d	15	c
16	c	17	c	18	b	19	d	20	c
21	d	22	b,c	23	d	24	d	25	a
26	a	27	c	28	c	29	a	30	d
31	a	32	b	33	b	34	a	35	c
36	a	37	b	38	b	39	a	40	d
41	a								

SOLUTIONS

S.2 (c)

F.PROCEDURE OPTIONS (MAIN) ⇒ Misspelled keyword ⇒ **Transposition Error.**

S.5 (c)

(i) and (ii) are correct about syntax directed translation.

S.6 (b)

(a), (c) are correct statements about syntax directed translation because values are evaluated by the semantic rules associated with the production rules.

S.7 (c)

(i) and (ii) both are treated as restriction for translation routines that are involved during parsing.

S.8 (d)

(i),(ii),(iii) are correct statements about compiler error detection.

S.9 (a)

(i) and (iv) are good and understandable methods to show errors while (ii) and (iii) are equivalent to (i) and (iv) but not a good method to represent the errors.

S.10 (d)

(a) L-attributed definitions can always be evaluated by the depth first visit of the parse tree.

(b) Every S-attributed definition is L-attributed, the restrictions only apply to the inherited attributes (not to synthesized attributes).

(c) A syntax-directed definition is 2-attributed of each inherited attributed of x_j, where $1 \le j \le n$, on the right side of $A \rightarrow x_1\ x_2\ \ldots\ x_n$ depends only on:

(i) The attributes of the symbol $x_1, \ldots, x_{j-1}$ to the left of x_j in the production and

(ii) the inherited attribute of A.

S.16 (c)

Terminal table is a permanent database that has on entry for each terminal symbol (e.g, arithmetic operators, keywords, nonalphameric symbols).

S.21 (d)

A Use-Definition chain (UD-chain) is a data structure that consists of a use, U, of a variable and all the definitions, D, of the variable that can reach without any other intervening definitions.

S.24 (d)

For evaluating semantic rule we use (a), (b), (c).

(a) ⇒ At compile time, these methods obtain an evaluation order from a topological sort of the dependency graph constructed from the parse free for each input.

(b) ⇒At compiler construction time, the semantic rules associated with productions are analyzed, either by hand, or by a specialized tool.

(c) ⇒An evaluation order is chosen without considering the semantic rules.

S.25 (a)

Semantic error can be detected both at compile time and run time. Undeclared and multiple declared identifiers are examples of semantic errors.

S.26 (a)

A parse tree for an S attributed definition can always be annotated by **evaluating the semantic rules** for the attributes at each node bottom up from the leaves to the root.

S.27 (c)

Conceptually, with **both syntax directed definition and translation schemes,** we parse the input token stream, build the parse tree and then traverse the tree as needed to evaluate the semantic rule at the parser tree nodes.

S.28 (c)

A parse tree showing the values of attributes at each node is called an **annotated preset parse tree.** The process of computing the attribute values at the nodes is called **annotating or decorating parse tree.**

S.29 (a)

The value of a synthesized attribute at a node is computed from the value of attributes at the children of that node in the parse tree. **The given declaration represents only 2 synthesized attributes lexval and value.**

S.30 (d)

The locally optimized code will be **M=A–B, X=M*C–D/M.**

S.31 (a)

e1 is called when an operand, beginning with either an id or a left parenthesis is expected but an operator, right parenthesis or the end of input is found.

S.32 (b)

The e2 is called when, we have right parenthesis on top of the stack but not on the input.

Error e2 is called from states 0, 1, 2, 4, 5 on finding a right parenthesis where we were expecting either the beginning of a new expression (or potentially the end of input for state 1).0

S.33 (b)

Error e3 is called from state 1,3,6,7,8,9 **on finding id or left paranthesis.**

S.34 (a)

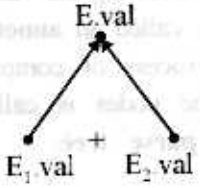
In abstract a symbol table is merely a table with two fields, a name field and an information field. We require several capabilities of the symbol table so we need to be able to (i),(ii),(iii),(iv).

S.35 (c)

From the production $E \rightarrow E_1 + E_2$ we get semantic rule as E.val: $= E_1.val + E_2.val$ and dependency graph is constructed bottom up, from the leaves to the root.

The leaves to the root.

Therefore we get, dependency graph as follows.



The tree nodes of the dependency graph marked by • represent the synthesized attributes E.val, $E_1.val$, and $E_2.val$ at the corresponding nodes in the tree.

S.36 (a)

(i) $\Rightarrow$ **example of Deletion error**

(ii) $\Rightarrow$ **example of Insertion error**

(iii) $\Rightarrow$ **example of Replacement error**

S.37 (b)

If the symbol table contains n names, the work necessary to insert a new name is proportional to n. To find data about a name we shall, on the average, search n/2 names, so the cost of an inquiry is also proportional to n. To insert n names and m inquiries the total work is **cn(n+m)**, where c is a contact representation of the time necessary for few machine operations.

S.38 (b)

Every S (synthesized) attributed definition is L-attributed. For implementing inherited attributed during bottom up parsing extends to some, but not LR grammars. Consider the following example

Production	Semantic Rule
$S \rightarrow L$	L. count : = 0
$L \rightarrow L_1 1$	L. count: = L. count+1
$L \rightarrow E$	print (L.count)

In the Example above the nonterminal L in $L \rightarrow E$ inherits the count of the number of 1's generated by S. Since the production $L \rightarrow E$ is the first that a bottom up parser would reduce by, the translator at the time can't know the number of 1' in the input. So in a bottom up evaluation of a syntax directed definition is L attributed in the given example. So we can say that L attributed definition is based on simple LR(1) grammar, but it can't be implemented always but inherited attributes can be evaluated only if the definition has synthesized attributes.

S.39 (a)

Generation of intermediate code based on an abstract machine code is useful in compiler because **it makes implementation of lexical analysis and syntax analysis easier.** Most compilers do not generate a parse tree explicitly but rather go to intermediate code directly as syntax analysis take place.

S.40 (d)

Absolute addresses of internal symbol need not be included in an object module.

6.5 Code Generation And Optimization

Level–1

Q.1 The run time support package manage_____

(a) allocation

(b) deallocation

(c) Both (a) and (b)

(d) Neither (a) nor (b)

Q.2 In programming language semantics, the term environment refers to _____

(a) a function that maps a storage location to the value held there

(b) a function that maps a name to a storage location

(c) Both (a) and (b)

(d) Neither (a) nor (b)

Q.3 In programming language semantics, the term state refers to _____

(a) a function that maps a storage location to the value held there

(b) a function that maps a name to a storage location

(c) a function that maps and name both

(d) None of these

Q.4 Which of the following cannot be used in stack allocation strategy?

(i) The value of local name must be retained when an activation ends

(ii) A called activation outlives the caller

(a) (i)

(b) (ii)

(c) (i) and (ii)

(d) Neither (i) nor (ii)

Q.5 Which of the following(s) is(are) incorrect about intermediate code representation?

(i) The indirect triples are and quadruples require about the same amount of space

(ii) The indirect triples much efficient for recording of code as compare to quadruples

(a) (i)

(b) (ii)

(c) (ii) and (i)

(d) Neither (i) nor (ii)

Q.6 The output of the code generator may be in the form of _____

(i) absolute machine language

(ii) relocate table machine language

(iii) assembly language

(a) (i),(ii)

(b) (ii),(iii)

(c) (i),(iii)

(d) (i),(ii),(iii)

Q.7 Which of the following is correct about Run Time Environment ?

(a) The allocation and deallocation of a data object is managed by the run time support package and design of the run time support package is influenced by the semantic of procedures.

(b) Each execution of a procedure is referred to as activation of the procedure and if the procedure is recursive, several of its activations may be alive at the same time .

(c) Both (a) and (b)

(d) Neither (a) nor (b)

Q.8 The main problem with generating code for boolean expressions and flow of control statement in a single pass is that _____

(a) during one single pass we may not know the lables that control must go to at the time jump statements are generated

(b) during one single pass we may know the lables but not able to go to at the time of jump statements generation.

(c) during one single pass we may know the lables but no control flow analysis in single pass for jump statements

(d) None of these

Q.9 Which of the following intermediate code representation is suitable for optimizing compiler?

(i) quadruples

(ii) triples

(iii) indirect triples

(a) (i),(ii),(iii)

(b) (i),(iii)

(c) (iii),(ii)

(d) (i),(ii)

Q.10 Translating a boolean expression into there address code without generating code for any of the boolean operators and without having the code necessarily evaluate the entire expression. This type of evaluation is called_____

(a) short circuit code

(b) jumping code

(c) control code

(d) Both (a) and (b)

Q.11 Peephole optimization is a form of

(a) Loop optimization

(b) Constant folding

(c) Local optimization

(d) None of these

Q.12 Heap allocation is required_____

(a) required for language that support recursion

(b) required for language that use dynamic scope rules

(c) required for languages that support dynamic data structures.

(d) None of these

Q.13 In which type of loader, the assembler run in one part of memory and place the assembled machine instructions and data, as they are assembled, directly into their assigned memory locations.

(a) Compile and go loader

(b) General loader

(c) Absolute loader

(d) None of these

Q.14 Match the following with reference to absolute loading

Group I		Group II	
(1)	Allocation	(i)	Loader
(2)	Linking	(ii)	Linker
(3)	Re-location	(iii)	Assembler
(4)	Loading	(iv)	Programmer

(a) (1)-iv, (2)-iii, (3)-i, (4)-ii

(b) (1)-iv, (2)-iv, (3)-iii, (4)-i

(c) (1)-ii, (2)-iii, (3)-ii, (4)-ii

(d) (1)-iii, (2)-iv, (2)-iv, (4)-ii

Q.15 The module loader performs which type of function

(a) loading

(b) assembling

(c) relocation

(d) linking

Q.16 Which of the following translation program converts assembly language programs to object programs?

(a) Assembler

(b) Compiler

(c) Microprocessor

(d) Linker

Q.17 Which of the following system program forgoes the production of object code to generate absolute machine code and load it into the physical main storage location from which it will be executed immediately upon completion of the assembly?

(a) Two pass assembler

(b) Load and go assembler

(c) Macroprocessor

(d) Compiler

Q.18 A system program that set up an executable program in main memory ready for execution is

(a) assembler

(b) linker

(c) loader

(d) text editor

Q.19 A processor is

(a) a device that performs a sequence of operations specified by instructions in memory

(b) the device where informations are stored

(c) a sequence of instructions

(d) typically characterized and time by interactive processing and time of the CPUs is time to allow quick response to each user

Q.20 Storage mapping is done by

(a) operating system

(b) compiler

(c) linker

(d) loader

Q.21 Which of the following functions is performed by loader?

(a) Allocate space in memory for the programs and resolve symbolic references between objects decks.

(b) Adjust all address dependent locations, such as address constants, to correspond to the allocated space.

(c) Physically place the machine instructions and data into memory.

(d) All of the these

Q.22 The disadvantage of "Compile and go" loading scheme is that

(a) a portion of memory is wasted because the case occupied by the assembler is unavailable to the object program.

(b) it is necessary to retranslate the users program deck every time it is run.

(c) it is very difficult to handle multiple segments, especially if the source programs are in different language and to produce orderly modular programs.

(d) all of these

Q.23 In an absolute loading scheme, which loader function is accomplished by loader?

(a) Reallocation

(b) Allocation

(c) Linking

(d) Loading

Q.24 A non relocatable program is one which

(a) cannot be made to execute in any area of storage other than the one designed for it at the time of its coding or translation.

(b) consists of a program and relevant information for its relocation.

(c) can itself perform the relocation of its address sensitive portions.

(d) all of these

Q.25 A linker is given object module for a set of program that were compiled separately. What information need not be included in an object module?

(a) Object code

(b) Relocation bits

(c) Names and locations of all external symbols defined in the object module?

(d) Absolute address of internal symbols.

Q.26 Heap allocation is required for languages

(a) that support recursion

(b) that support dynamic data structures

(c) that use dynamic scope rules

(d) none of these

Q.27 Three address code involves

(a) exactly 3 address

(b) at the most 3 address

(c) no unary operators

(d) none of these

Q.28 Access time of the symbol table will be logarithmic, if it is implemented by

(a) detect error

(b) search tree

(c) hash table

(d) self organization list

Q.29 When exceptional situation occurs outside the CPU the H/W signal given is

(a) reset

(b) interrupt

(c) hold

(d) wait

LEVEL–2

Q.30 For expression x:= y[i] the triple representation is:

(a)

	Op	arg1	arg2
(0)	=[ ]	i	x
(1)	assign	(0)	y

(b)

	Op	arg1	arg2
(0)	=[ ]	y	i
(1)	assign	x	(0)

(c)

	Op	arg1	arg2
(0)	[ ]=	x	i
(1)	assign	(0)	y

(d)

	Op	arg1	arg2
(0)	[ ]=	i	x
(1)	assign	x	(0)

Q.31 Translation for certain operation is as shown below:

80: if a< b goto 83	87 : t_2: = 1
81: t_1 = 0	88: if e < f goto 91
82: goto 84	89 : t_3: = 0
83: t_1:=1	90: goto 92
84: if c<d goto 87	91 : t_3 = 1
85: t_2: =0	92: t_4:=t_2 and t_3
86: go to 88	93: t_5:=t_1: or t_4

The, corresponding boolean expression_____

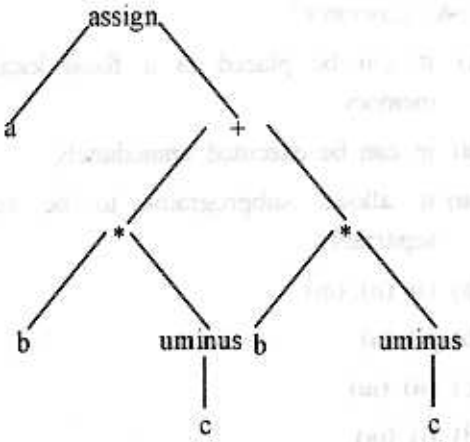
(a) a < b c < d e < f

(b) a < b or c < d or e < f

(c) a < b and <c<d and e < f

(d) a < b or c < d and e<f

Q.32 The intermediate code generated for following syntax tree is _____



(a) a b c* bc uminus * + assign
(b) a b c uminus * bc uminus* + assign
(c) a b uminus * b c * + assign
(d) None of these

Q.33 While calculating the square of the entered number one programmer use (i) and other use (ii), when while using algebraic transformation which is cheaper.

(i) x: =y**2 (** ⇒ Exponentiation operator)
(ii) x: y*y (* ⇒ Multiplication operator)

(a) (i)
(b) (ii)
(c) (i) and (ii) are equivalent (costwise)
(d) Insufficient data to decide

Q.34 The expression x[i] stored in block of consecutive locations. The value of i varies from 50 to 100, and an array x contain integer value of size 2 bytes if base address of x is 1000, then 96^{th} element of array x begins in location_____

(a) 1046
(b) 1092
(c) 1050
(d) None of these

Q.35 In case of General loader scheme, which of the following is incorrect?

(i) Reassembly is no longer necessary to run the problem again and again.
(ii) The loader is assumed to be smaller than the assembler

(a) (i)
(b) (ii)
(c) (i) and (ii)
(d) Neither (i) nor (ii)

Q.36 A linker reads four modules whose length are 300,900,1000 and 600 words respectively. If they are loaded in that order, what are the relocation constants?

(a) 0,300,1500,2700
(b) 0,300,1200,2200
(c) 300,1200,2200,2800
(d) None of these

Q.37 Which of the following statements are TRUE?

I. There exist parsing algorithms for some programming languages whose commodities are less than $\theta(n^3)$
II. A programming language which allows recursing can be implanted with static storage allocations.
III. No L attributed definition can be evaluated in the framework of bottom up parsing
IV. Code improving transformations can be performed at both source language and intermediate code level.

(a) I and II
(b) I and IV
(c) III and IV
(d) I, III and IV

Q.38 Correctly match the following pairs and determine the answer from the codes given below;

Group I		Group II	
(1)	Activation	(i)	Linking loader
(2)	Location counter	(ii)	Garbage collection
(3)	Reference counts	(iii)	Subroutine call
(4)	Address relocate	(iv)	Assembler

Codes:

	A	B	C	D
(a)	3	4	1	2
(b)	4	3	1	2
(c)	4	3	2	1
(d)	3	4	2	2

LEVEL–3

Q.39 Let A be 10 x 20 array with $low_1 = low_2 = 1$. Therefore, $n_1 = 10$ and $n_2 = 20$. By assuming w = 4, assignment x: A [y,z] is translated into the sequence of three address statements. Which one of the following is

(a) $t_1 := y*20$
$t_1 = t_1+z$
$t_2 := base_A-84$
$t_3 := 4 * t_1$
$t_4 := t_2[t_3]$
$x := t_4$

(b) $t_1 := y*20$
$t_1 := t_1+z$
$t_2 := base_A-84$
$t_3 := t_1$
$t_4 := t_2[t_3]$
$x := t_4$

(c) $t_1 := y*20$
$t_1 := t_1+z$
$t_2 := base_A-84$
$t_3 := 4* t_1$
$t_4 := t_3[t_2]$
$x := t_4$

(d) $t_1 := y*20$
$t_1 := t_1+z$
$t_2 := t_1 + base_A-84$
$t_3 := 4*t_1$
$t_4 := t_2[t_3]$
$x := t_4$

Q.40 Which one of the following is valid assumption prior to code generation

(i) Prior to code generation the front end has scanned, parsed and translated the source program into a reasonably detailed intermediate representation

(ii) The necessary type checking has taken place

(iii) the input is free of errors

(a) (i),(ii),(iii)

(b) (ii),(iii)

(c) (i),(iii)

(d) (i),(ii)

Q.41 Which one is the advantage of producing an absolute machine language program as output of code generator?

(i) it can be placed in a fixed location in memory

(ii) in can be executed immediately

(iii) it allows subprograms to be compiled separately

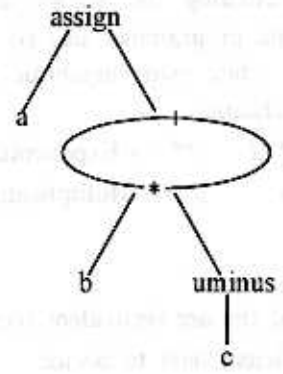
(a) (i), (ii), (iii)

(b) (i), (ii)

(c) (ii), (iii)

(d) (i), (iii)

Q.42 Consider the following DAG, the corresponding three address code for the same is _____



(a) $t_1 := -c$
$t_2 := b*t_1$
$t_5 := t_2+t_2$
$a := t_5$

(b) $t_1 := -c$
$t_2 := b*t_1$
$t_5 := t_2 + t_1$
$a := t_5$

(c) $t_1 := -c$
$t_2 := t_1*b$
$t_5 := t_1 + t_2$
$a := a_5$

(d) None of these

Q.43 Match the following with reference to cost and instructions:

Group I		Group II	

(1)	MOV b,R_0
	ADD c,R_0
	MOV R_0,a
		(a)	3

(2)	MOV b,a ADD c,a	(b)	2
(3)	MOV *R_1*,R_0 ADD *R_2*R_0	(c)	6
(4)	ADD R_2,R_1 MOV R_1,a	(d)	5
		(e)	4

(a) 1–c, 2–d, 3–b, 4–a

(b) 1–b, 2–d, 3–d, 4–a

(c) 1–e, 2–c, 3–b, 4–a

(d) 1–b, 2–c, 3–d, 4–c

Q.44 While addressing array elements if the width of each array elements is w, then i^{th} element of array A begins in location

base + (i–low)*w

where low is the lower bound on the subscript and base is the relative address of the storage allocated for the array. Then, (i) can be partially evaluated at compile time if it is rewritten as ___

(a) i*w+(base+lww–w)

(b) i–w+(base–low*w)

(c) i*w+(base–low*w)

(d) i*+w|–(base+low–w)

Q.45 The three code translation for

a or b and not c is _____

(a) t_1: = a or t_2

t_2: b and t_3

t_3: not c

(b) t_1: not c t_2

t_2: b and t_1

t_3: a or t_3

(c) t_1:= not c

t_2:= b and t_1

t_3: a or t_2

(d) None of these

Q.46 The quadruples representation for following postfix notation is _____

xyz–*yz–*+=

(a)

	Op	arg1	arg2	Result
(0)	–	z		tl
(1)	*	y		t2
(2)	–	z	tl	t3
(3)	*	y	t3	t4
(4)	+	t2	t4	t5
(5)	=	t5		a

(b)

	Op	arg1	arg2	Result
(0)	–	z		tl
(1)	*	y	tl	t2
(2)	–	z	t3	t3
(3)	*	t	t4	t4
(4)	+	t2		t5
(5)	=	t5		a

(c)

	Op	arg1	arg2	Result
(0)	–	z	tl	tl
(1)	*	y		t2
(2)	–	z	t3	t3
(3)	*	y		t4
(4)	+	t2	t4	t5
(5)	=	t5		a

(d) none of these

Q.47 The indirect triple representation for the following representation is _____

	op	arg 1	arg 2
(21)	–	c	
(22)	*	b	(21)
(23)	–	c	
(24)	*	b	(23)
(25)	+	(22)	(24)
(26)	=	a	(25)

(a)

	op	arg 1	arg 2
(21)			
(22)			(21)
(23)			
(24)			(23)
(25)		(22)	(24)
(26)			(25)

(b)

	Statement
(0)	(14)
(1)	(15)
(2)	(16)
(3)	(17)
(4)	(18)
(5)	(19)

(c)

	Statement		op	arg 1	arg 2
(0)	(21)	(21)	–	c	–
(1)	(22)	(22)	*	b	(21)
(2)	(23)	(23)	–	c	–
(3)	(24)	(24)	*	b	(23)
(4)	(25)	(25)	+	(22)	(24)
(5)	(26)	(26)	=	a	(25)

(d) Not possible

Q.48 In a simplified computer the instructions are:

OP R_j, R_i–Performs R_j OP R_i and stores the result in register R_i

OP m, R_i–Performs val OP R_i and stores the results in R_i val denotes the content of memory locations m.

MOV m, R_i –Moves the content of register R_i.

The computer has only two register, and OP is either ADD or SUB. Consider the following basic block:

$t_1 = a + b$

$t_2 = c + b$

$t_3 = e - t_2$

$t_4 = t_1 - t_3$

Assume that all operand are initially in memory. The final value of the computation should be in memory. What is the minimum number of MOV instructions the code generated for this basic block.

(a) 2

(b) 3

(c) 5

(d) 6

GATE QUESTIONS

Q.49 Consider the syntax directed definition shown below. **[GATE 2003]**

[2-Marks]

$S \rightarrow$ **id** : = E	{gen (**id**. place = E. place;);}
$E \rightarrow E_1 + E_2$	{t=newtemp(); gen (t=E_1. place + E_2. place;); E. place = t;};
$E \rightarrow$ **id**	{E.place = **id**.place;}

Here, gen is a function that generates the output code, and newtemp is a function that returns the name of a new temporary variable on every call. Assume that ti's are the temporay variable names generated by newtemp.

For the statement 'X:=Y+Z', the 3 address code sequence generated by this definition is

(a) X = Y + Z

(b) t_1 = Y + Z; X = t_1

(c) t_1 = Y; t_2= t_1 + Z; X = t_2

(d) t_1 = Y; t_2 = Z, t_3 = t_1+t_2;X = t_3

Q.50 Which of the following is NOT an advantage of using shared, dynamically linked libraries as opposed to using statically linked libraries? **[GATE 2003] [2-Marks]**

(a) Smaller sizes of executable files
(b) Lesser overall page fault rate in the system
(c) Faster program start up
(d) Existing programs need not be re linked to take advantage of newer versions of libraries.

Q.51 Consider a program P that consists of two source modules M_1 and M_2 contained in two different files. If M_1 contains a reference to a function defined in M_2 the reference will be resolved at **[GATE 2004] [1-Mark]**

(a) Edit time
(b) Compile time
(c) Link time
(d) Load time

Q.52 Consider the grammar rule $E \rightarrow E_1 - E_2$ for arithmetic expressions. The code generated is targeted to a CPU having a single user register. The subtraction operation requires the first operand to be in the register. If E_1 and E_2 do not have any common sub expression, in order to get the shortest possible code. **[GATE 2004] [1-Mark]**

(a) E_1 should be evaluated first
(b) E_2 should be evaluated first
(c) Evaluation of E_1 and E_2 should necessary be interleaved
(d) Order of evaluation of E_1 and E_2 is of no consequence

Q.53 Consider line number 3 of the follow C program

```
int main () {          /* Line 1 */
int I,N,               /* line 2 * /
fro (I=0,I<N,I++);     /* line  3 */
}
```

Identify the compiler's response about this line while creating the object module **[GATE 2005] [2-Marks]**

(a) No compilation error
(b) Only a lexical error
(c) Only syntactic errors
(d) Both lexical and syntactic errors.

Q.54 Consider the following translation scheme.

S→ER
R→*E{print('*');}R|ε
E→F+E {print('+');}|F
F→(S)|id {print (id. value);}

Here id is a taken that represents an integer and id. value represents the corresponding integer value. For an input '2*3+4' this translation scheme prints **[GATE 2006] [2-Marks]**

(a) 2 * 3 + 4
(b) 2 * + 3 4
(c) 2 3 * 4 +
(d) 2 3 4 + *

Q.55 Consider the following C code segment.

```
for (i=0;i<n;i++) {
    for (J=0;J<n;J++) {
        if(i%2)
            {x+=(4*j+5×i);
            y+=(7+4*j);
            }
        }
}
```

Which one of the following is false? **[GATE 2006] [2-Marks]**

(a) The code contains loop in variant computation.
(b) There is scope of common sub expression elimination in this code
(c) There is scope of strength reduction in this code
(d) There is scope of dead code elimination in this code

Q.56 Some code optimizations are carried out on the intermediate code because **[GATE 2008] [1-Mark]**

(a) They enhance the portability of the complier to other target processors.
(b) Program analysis is more accurate on intermediate code than on machine code.
(c) The information from data flow analysis cannot otherwise be used for optimization.
(d) The information from the front end cannot otherwise be used for optimization.

ANSWER KEY

1	c	2	b	3	a	4	c	5	b
6	d	7	c	8	a	9	b	10	d
11	b	12	c	13	a	14	b	15	a
16	a	17	b	18	c	19	a	20	d
21	d	22	d	23	d	24	a	25	d
26	b	27	d	28	b	29	b	30	b
31	d	32	b	33	b	34	b	35	d
36	b	37	b	38	d	39	a	40	a
41	b	42	a	43	c	44	c	45	c
46	b	47	c	48	c	49	d	50	c
51	c	52	b	53	c	54	d	55	d
56	b								

SOLUTIONS

S.1 (c)

The allocation and deallocation of data objects is managed by the run time support package, consisting of routines loaded with generated target code.

S.2 (b)

In programming language semantics the term environment refers **to a function that maps a name to a storage location.**

S.3 (a)

In programming language semantics, the term state refers to a function that maps a storage location to the value there.

S.4 (c)

We can not use (i) and (ii) for stack allocation.

(ii) $\Rightarrow$ This possibility cannot occur for those languages where activation trees correctly depict the flow of control between procedures.

In each of the (i) and (ii), the deallocation of activation records need not occur in a last in first out fashion, so storage cannot be organized as a stack.

S.5 (b)

The indirect triples and quadruples are equally efficient for recording of code.

S.6 (d)

The output of the code generator is the target program. Like the intermediate code this output may take on a variety of forms i.e. (i),(ii),(iii).

S.7 (c)

(a) and (b) are correct about Run-Time Environment.

S.8 (a)

Problem with code generation for boolean expression and flow of control statement.

S.9 (b)

A more important benefit of **quadruples** appears in a optimizing compiler, where statements are often moved around. But triple has a problem in doing same, but **indirect triples** present no such problem. A statement can be moved by recording the statement list.

S.12 (c)

Heap allocation is required for languages that support dynamic data structure.

S.13 (a)

Function of compile and go loader which is also called as assemble and go loader.

S.14 (b)

In case of absolute loading scheme

Allocation ⇒ done by programmer

Linking ⇒ done by programmer

Relocation ⇒ done by assembler

Loading ⇒ done by loader

S.15 (a)

Function of the module loader is simply loading.

S.28 (b)

If memory space is not the constraint, then by increasing the number of bits to K, the access time can be reduced by a factor of K. So average number of items in a bin will decrease as the number of bins increases. In the case of list, access time will be proportional to n, the number of items but we will be using as much memory space as is absolutely necessary. **In the case of search tree implementation, the access time will be logarithmic.**

S.31 (d)

For given three address code generation we get the boolean expression as a < b or c < d and e<f.

S.32 (b)

The syntax tree is one of the form of intermediate code. The postfix notation is also another from of intermediate code.

∴ The postfix notation for syntax tree is ,

a b c uminus * bc uminus * + assign

S.33 (b)

The exponentiation operator in the statement x : = y**2 usually requires a function call to implement. Using an algebraic transformation this statement can be replaced by the cheaper but equivalent statement **x: y*y.**

S.34 (b)

96^{th} element starts at = base + (i–low) *w

= 1000 + (96–50)*2

= 1000 + 46 *2

= 1092

∴ **96^{th} element stored at locations 1092.**

S.35 (d)

(i) and (ii), are correct statement about General loader scheme.

(ii) ⇒ The loader is assumed to be smaller than the assembler so that more memory is available to the user.

S.36 (b)

Relocation constant = final loading address – original loading address.

S.39 (a)

For given specification, after translation we get

t_1 : = y*20

t_1 = t_1+z

t_2: = $base_A$–84

t_3: = t_1

t_4: = $t_2[t_3]$

x : = t_4

S.40 (a)

(i) ⇒ The values of name appearing in the intermediate language can be represented by quantities that the target machine can directly manipulate (bits, integers, reals, pointers, etc).

(ii) ⇒ Type conversion operators have been inserted wherever necessary and obvious semantic errors. (e.g. attempting to index an array by a floating point number) have already been detected.

(iii) ⇒ The input is free of errors.

S.41 (b)

(i) and (ii) are the advantages of producing an absolute machine language program as output of code generator.

(iii) is the advantage of producing a relocatable code as output of code generator.

S.42 (a)

For given DAG, we contract the three address code as following.

t_1:= –c

t_2:=b*t_1

t_5:=t_2+t_2

a:=t_5

S.43 (c)

The cost of (1) ⇒ 4

The cost of (2) ⇒ 6

The cost of (3) ⇒ 2

The cost of (4) ⇒ 3

S.44 (c)

base+ (i–low)*w can be rewritten as i*w + (base–low*w). The subexpression c = base–low*w can be evaluated when the declaration of the array is seen. We assume that c is saved in the symbol table entry for A, so the relative address of [i] is obtained by simply adding i*w to c.

S.45 (c)

Boolean expression will be evaluated completely, from left to right, in a manner similar to arithmetic expression. Therefore, the translation for given expression is

t_1 := not c

t_2 := b and t_1

t_3 := a or t_2

S.46 (b)

A quadruple is a record structure with four fields, which we call op, arg1, arg2 and result. And for given postfix notation, the quadruple representation is as option (b).

S.47 (c)

Indirect triple representation is another form of three address code that has been considered is that of listing pointers to triples, rather than listing tipples themselves. The indirect triple representation for table is as option (c).

S.48 (c)

Check it out using following code.

MOV a,R_1

opr b,R_1 t_1=a + b

MOV c,R_1

MOV d,R_1

MOV b,R_2

opr R_1,R_2 t_2 = c+b

opr c,R_2 t_3 = c–t_2

MOV t_3,R_1

opr t_3,R_1 t_4 = t_1–t_3

Hence, minimum number of MOV instruction is = 5.

S.49 (d)

3 address code sequence generated by definition for X =Y + Z is

t_1 = Y; t_2 = Z; t_3 = t_1+t_2;X = t_3

because in 3 address we have two operands and one operator and these operands are different.

S.50 (c)

Statically linked libraries **program run faster from start up** since the linking has been performed during the compile time only and there is no need for the linking at run time of program and program can run faster.

S.51 (c)

Modules may be compiled separately and all linking between references is made at **link time.**

S.52 (b)

To optimize the solution evaluate the expression E_2. Then we can calculate E_1 and finally E_1 will be one of operands that will be in register and we can perform substraction directly. But if we follow the opposite then we have to make move store operation.

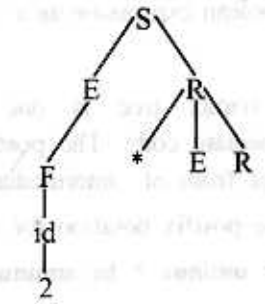
S.53 (c)

In 2nd line no miss spelling is there. It is int only.

Error is underlined function fro( )

Which is a **syntactical error rather than lexical.**

S.54 (d)



So an input 2 * 3 + 4 prints

2 3 4 + *

S.55 (d)

There is no dead code in given code segment. So there is no scope of dead code elimination in this code.

S.56 (b)

Some code optimizations are carried out on the intermediate code because **program analysis is more accurate on intermediate code than on machine code.**

UNIT–7

OPERATING SYSTEM

Unit–7

Operating System

7.1 PROCESS, IPC AND DEADLOCK

LEVEL–1

Q.1 Windows 95 is

(a) Multi tasking, Multiuser

(b) Multitasking

(c) Multiprocessing, Multiuser

(d) None of the above

Q.2 Where does the dispatcher reside?

(a) In RAM or main memory

(b) In Cache Memory

(c) In HDD

(d) In external / auxiliary device

Q.3 Heuristic scheduling is more efficient than multilevel priority scheduling.

(a) only for I/O jobs

(b) only for CPU jobs

(c) none of the above statement is true

(d) both (a) & (b) statements are true

Q.4 For a particular code to be shareable, it should be

(a) serially refusable

(b) Re-entrant

(c) Reducible

(d) None of these

Q.5 What do you understand by busy waiting?

(a) A process periodically checking on a variable

(b) A process continuously checking on a variable

(c) A process polling on a variable

(d) A process issuing an interrupt

Q.6 An event counter

(a) is an integer counter that does not decrease

(b) keeps track of the number of occurrences of events of a particular class of related events.

(c) orders events

(d) Both (a) & (b)

Q.7 Pick out a wrong statement

(a) Semaphores are used to deal with n process critical section problem

(b) Semaphores are used to solve synchronization problems

(c) Semaphores are used to solve asynchronization problems

(d) All are correct statements.

Q.8 In message passing system, how many points of synchronization are possible?

(a) 1

(b) 2

(c) 4

(d) 6

Q.9 Why mailboxes are used in message passing system? Choose most appropriate option.

(a) All processes can communicate independently

(b) One sender can talk to more than one receiver

(c) To store fixed length messages

(d) To interface with both sender and receiver

Q.10 What is true about starvation and deadlock?

(a) Starvation only leads to deadlock

(b) Starvation is an extreme case of deadlock

(c) Deadlock is an extreme case of starvation

(d) Deadlock and starvation are two unrelated concepts

Q.11 Pick out the false statement:

The side effect of using deadlock prevention algorithm is

(a) Possibly low utilization

(b) Reduced system throughput

(c) Increased waiting time

(d) All are true

Q.12 CPU scheduling may take place under

(a) When a process switches from the running state to the waiting state, I/O request or invocation of wait for the termination of one of the child processes

(b) When a process switches form the running state to the ready state, when an interrupt occurs.

(c) When a process switches from the waiting state to the ready state.

(d) All of the above

Q.13 _____ is the module that gives control of the CPU to the process selected by the short term scheduler.

(a) Dispatcher

(b) Loader

(c) Linker

(d) Scheduler

Q.14 Every time scheduling is done, all jobs present at that time are considered for scheduling. A job may get scheduled again and again. This is nothing but_____

(a) Static scheduling

(b) Dynamic scheduling

(c) Preemptive static scheduling

(d) Preemptive dynamic scheduling

Q.15 On kernel when process P_i requests an I/O operation on some device d, and the I/O operations completed successfully, then process P_i changes its state_____

(a) first to blocked and remain as it is.

(b) first to blocked and finally change to ready.

(c) first to ready and remain as it is.

(d) first to ready and finally change to blocked.

Q.16 Which of the following statements (s) is/are true regarding the Process Management?

(1) The OS allocates the CPU time to various users based on certain policy.

(2) The PCB block is maintained by the CPU.

(3) The list of blocked processes is maintained in the priority order by the OS.

(4) The process switch occurs only if a process requests an I/O before the time slice is over or it consumes the full time slice.

(a) 4 is false

(b) 1, 2, 4 are true

(c) 1, 4 are true

(d) All the above are true

Q.17 In a batch environment, a process is created_____

(a) when a new user attempts to log on.

(b) in response of Dispatcher

(c) in response to the submission of a job

(d) none of these

Q.18 In time sharing system, the process for a particular user is to be terminated_____

(a) when memory is unavailable

(b) when user uses invalid instructions

(c) when new user attempts to log on

(d) when the user logs off or turns off his or her terminal.

Q.19 In UNIX,

(a) a CPU-bound process is given higher priority than an I/O bound process.

(b) an I/O - bound process is given higher priority than CPU bound process

(c) both CPU - bound and I/O bound processes are of equal priority.

(d) it depends on the scheduling algorithm.

Q.20 If one solves the problem of Producer - Consumer using semaphores, then empty semaphore will have initial value as

(a) 1

(b) 0

(c) number of slots in the buffer

(c) any non-zero value

Q.21 Which of the following is/are the reasons for Process Creation?

(a) (i) New batch job

(ii) New user attempts to log on

(b) (i) Privileged instruction

(ii) If a user request the particular file to be printed.

(c) (i) For purpose of modularity or to exploit parallelism.

(ii) The process attempts to use a resource or file that is not allowed to use.

(d) none of the above.

Q.22 The shortest-remaining-time first is same as_____

(a) non-preemptive SJF scheduling

(b) preemptive SJF scheduling

(c) non-preemptive FCFS scheduling

(d) none of these

Q.23 Dijkstra's banking algorithm in an operating system solves the problem of

(a) deadlock avoidance

(b) deadlock recovery

(c) mutual exclusion

(d) context switching

Q.24 Which of the following are true?

(a) A re-entrant procedure can be called any number of times.

(b) A re-entrant procedure can be called even before the procedure has not returned from its previous call.

(c) Re-entrant procecdures cannot be called recursively.

(d) Re-entrant procedures can be called recursively.

Q.25 Cascading termination refers to termination of all child processes before the parent terminates

(a) normally

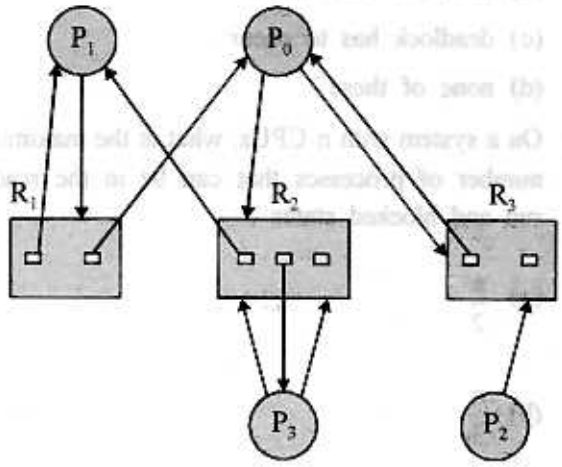
(b) abnormally

(c) normally or abnormally

(d) none of the above

Common Data For Question 26 & 27:

Consider the given figure:



Figure

Q.26 This system is in a deadlock state. This remark is

(a) true

(b) false

(c) impossible to determine

(d) unpredicatable

Q.27 Which of the following is a safe sequence?

(a) P0, P1, P2, P3

(b) P1, P0, P3, P2

(c) P2, P0, P1, P3

(d) none of the above

LEVEL–2

Q.28 A system has 3 processes sharing 4 resources. If each process needs a maximum of 2 units, then

(a) deadlock can never occur

(b) deadlock may occur

(c) deadlock has to occur

(d) none of these

Q.29 'm' processes share 'n' resources of the same type. The maximum need of each process doesn't exceed 'n' and the sum of their maximum needs is always less than m + n. In this set up

(a) deadlock can never occur

(b) deadlock may occur

(c) deadlock has to occur

(d) none of these

Q.30 On a system with n CPUs, what is the maximum number of processes that can be in the ready, run and blocked states ?

(a) $\frac{n}{2}$

(b) $\frac{n}{3}$

(c) n

(d) None

Q.31 On a system with n CPUs, what is the minimum number of processes that can be in the ready, run and blocked states?

(a) 0

(b) 1

(c) 2

(d) None

Q.32 Given system with *n* processes, how many possible ways can those processes be scheduled?

(a) n

(b) n!

(c) $\frac{n}{2}$

(d) None

Common Data For Questions 33 to 37:

On a system using round-robin scheduling, let *s* represent the time needed to perform a process switch, q represent the round-robin time quantum, and r represent the average time a process runs before blocking on I/O. CPU efficiency is

Q.33 $q = \infty$

(a) r

(b) $\frac{1}{r+s}$

(c) $\frac{r}{r+s}$

(d) None

Q.34 $q > r$

(a) r

(b) $\frac{1}{r+s}$

(c) $\frac{r}{r+s}$

(d) None

Q.35 $q < r$

(a) q

(b) $\frac{r}{q+s}$

(c) $\frac{q}{q+s}$

(d) None

Q.36 s = q = r

(a) $\frac{1}{3}$

(b) $\frac{1}{2}$

(c) $\frac{1}{5}$

(d) None

Q.37 q nearly 0

(a) 0

(b) 1

(c) 1/2

(d) None

Q.38 If m = number of resources,

n = number of processes in the system then, Banker's algorithm, although a general algorithm, may require_____ operations

(a) m * n

(b) (m * n) * (n* n)

(c) (m * m) * n

(d) m * (n * n)

Q.39 In real time system, CPU utilization may range from _____ (for a lightly loaded system) and _____ (for heavily loaded system)

(a) 0 % to 100%

(b) 20% to 100%

(c) 40% to 90%

(d) 205 to 90%

Q.40 Which of the following statement is not correct w.r.t. Deadlock?

(a) In a deadlock, processes never finish executing and system resources are tied up, preventing other jobs from starting

(b) A set $\{P_o, P_1, \ldots P_n\}$ of waiting processes must exist such that P_o is waiting for a resource that is held by P_1.P_1 is waiting for a resource that is held by P_2, ... P_{n-1} is waiting for a resource that is held by P_n, and P_n does not have to wait for any process

(c) A resource can be released only voluntarily by the process holding it, after that process has completed its task.

(d) A mutual exclusion, hold and wait, no preemption and circular wait, all these conditions are not completely independent

Q.41 A sequence of processes $<P_1, P_2, \ldots P_n>$ is a safe sequence for the current allocation state if

(a) for each P_i, the resources that P_i can still request can be satisfied by the currently available resources plus the resources held by all the P_i, with $j > i$.

(b) for each P_i, the resources that P_i can still request can be satisfied by the currently available resources plus the resources held by all the P_i with $j < i$.

(c) for each P_i, the resources that P_i can still request can be satisfied by the currently available resources plus the resources held by all the P_i, with $j < i$.

(d) for each P_i, the resources that P_i can still request can be satisfied by the currently available resources, with $j > i$

Q.42 Consider the following situation:

Consider a process which copies from a card reader to a disk file, sorts the disk file, and then prints the result to a line printer and to magnetic tape. Then solution for this situation is :

(i) All resources must be requested at the beginning of the process, then process must initially request the card reader, disk file, line printer and tape drive. It will hold the magnetic tape drive for its entire execution, even though it needs it only at the end.

(ii) The process requests initially only the card reader and disk file. It copies from the card reader to the disk and then releases both the card reader and the disk file. The process must then re-request the disk file and the line printer. After copying the disk file to the line printer, it releases both, and then requests the disk file and tape drive,. It copies the disk file to tape, then releases these two resources and terminates.

(a) Only (i)

(b) Only (ii)

(c) Both (i) and (ii)

(d) Neither (i) not (ii)

LEVEL–3

Q.43 At a particular time of computation, the value of a counting semaphore is 7. Then 20 P operations and 'x'. V operations were completed on this semaphore. If the final value of the semaphore is 5, x will be

(a) 15

(b) 22

(c) 18

(d) 13

Q.44

PROCESS	Burst time
P1	22
P2	5
P3	6

Above processes arrive in the order P1, P2, P3 and are served in FCFS order. Then average waiting time is:

(a) 16 unit

(b) 17 unit

(c) 0.17 unit

(d) 0.16 unit

Q.45 Consider the following four process, with the length of the CPU burst time in milliseconds.

Process	Arrival time	Burst time
P1	0	8
P2	1	4
P3	2	9
P4	3	5

Using the Shortest-Remaining-Time-First(SRTF) scheduling, the average waiting time is-

(a) 6.5 milliseconds

(b) 7.5 milliseconds

(c) 7.75 milliseconds

(d) None of these

Q.46 A process is executed in 100 msec. It takes 10 msec for the CPU to decide which process to execute. Then how much percentage of CPU time is used for scheduling.

(a) 10%

(b) 9%

(c) 100%

(d) 90%

Q.47 Consider the following performance table for SPFS (Shortest Path First Scheduling).

Position in Batch	Job Arrival Time (Ai)	Job Completion Time (Ci)	Weighted turn Around (Wi)
1	2	7	1.00
2	7	10	1.33
3	10	15	2.67
4	12	17	2.28
5	13	19	12.40

For this batch of process, the throughput is:

(a) 29.4

(b) 3.4

(c) 0.34

(d) 0.294

GATE QUESTIONS

Q.48 A critical region is [GATE 1987]

(a) one which is enclosed by a pair of P and V operations on semaphores

(b) a program segment that has not been proved bug-free

(c) a program segment than often causes unexpected system crashes

(d) a program segment where shared resources are accessed

Q.49 A "link editor" is a program that : **[GATE 1991]**

(a) matches the parameters of the macro definition with locations of the parameters of the macro call

(b) matches external names of one program with their location in other programs

(c) matches the parameters of subroutine definition with the location of parameters of subroutine call.

(d) acts as link between text editor and the user

(e) acts as link between compiler and user program.

Q.50 At a particular time of computation the value of a counting semaphore is 7. Then 20 P operations and 15 V operations were completed on this semaphore. The resulting value of the semaphore is **[GATE 1992]**

(a) 42

(b) 2

(c) 7

(d) 12

Q.51 Assume that the following jobs are to be executed on a single processor system

Job Id	CPU Burst time
p	4
q	1
r	8
s	1
t	2

The jobs are assumed to have arrived at time 0^+ and in the order p, q, r, s, t. Calculate the departure time (completion time) for job p if scheduling is round robin with time slice 1.

[GATE 1993]

[2-Marks]

(a) 4

(b) 10

(c) 11

(d) 12

(e) None of the above

Q.52 Consider a system having m resources of the same type. These resources are shared by 3 processes A, B and C, which have peak demands of 3, 4 and 6 respectively. For what value of m deadlock will not occur? **[GATE 1993]**

[2-Marks]

(a) 7

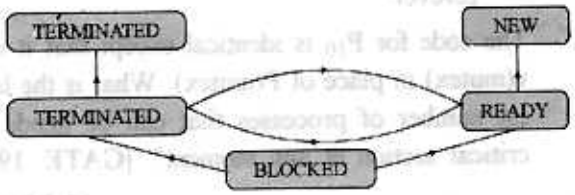
(b) 9

(c) 10

(d) 13

(e) 15

Q.53 The process state transition diagram in the figure is representative of



[GATE 1996]

[1-Marks]

(a) a batch operating system

(b) an operating system with a preemptive schedular

(c) an operating system with a non-preemptive scheduler

(d) a uni-programmed operating system

Q.54 A solution to the Dining Philosophers Problem which avoids deadlock is **[GATE 1996]**

[2-Marks]

(a) ensure that all philosophers pick up the left fork before the right fork

(b) ensure that all philosophers pick up the right before the left fork

(c) ensure that one particular philosopher picks up the left fork before the right. Fork, and that all other philosophers pick up the right fork before the left fork.

(d) None of the above

Q.55 A Critical section is a program segment [GATE 1996]

[1-Mark]

(a) which should run in a certain specified amount of time

(b) which avoids deadlocks

(c) where shared resources are accessed

(d) which must be enclosed by a pair of semaphore operations, P and V

Q.56 Each process P_i, i = 1 ..... 9 is coded as follows

```
repeat
    P(mutex)
    {critical section}
    v(mutex)
forever
```

The code for P_{10} is identical except that it uses v(mutex) in place of P(mutex). What is the largest number of processes that can be inside the critical section at any moment? [GATE 1997]

[2-Marks]

(a) 1

(b) 2

(c) 3

(d) None of above

Q.57 An operating system contain 3 user processes each requiring 2 units of resource R. The minimum number of units of R such that no deadlocks will ever arise is [GATE 1997]

[2-Marks]

(a) 3

(b) 4

(c) 5

(d) 6

Q.58 A computer has six tape drives, with n processes competing for them. Each process may need two drives. What is the maximum value of n for the system to be deadlock free? [GATE 1998]

[1-Mark]

(a) 6

(b) 5

(c) 4

(d) 3

Q.59 A counting semaphore was initialized to 10. Then 6 P (wait) operations and 4 V (signal) operations were completed on this semaphore. The resulting value of the semaphore is [GATE 1998]

[1-Mark]

(a) 0

(b) 8

(c) 10

(d) 12

Q.60 When the result of a computation depends on the speed of the processes involved there is said to be [GATE 1998]

[1-Mark]

(a) cycle stealing

(b) rare condition

(c) a time lock

(d) a deadlock

Q.61 In a resident–OS computer, which of the following systems must reside in the main memory under all situations? [GATE 1998]

[1-Mark]

(a) Assembler

(b) Linker

(c) Loader

(d) compiler

Q.62 Which of the following is NOT a valid deadlock prevention scheme? [GATE 2000]

[2-Marks]

(a) Release all resources before requesting a new resource

(b) Number the resources uniquely and never request a lower numbered resource than the last one requested

(c) Never request a resource after releasing any resource

(d) Request and all required resources be allocated before execution

Q.63 Which of the following need not necessarily be saved on a context switch between processes?

[GATE 2000]
[1-Mark]

(a) General purpose registers
(b) Translation look aside buffer
(c) Program counter
(d) All of the above

Q.64 Consider Peterson's algorithm for mutual exclusion between two concurrent processes i and j. The program executed by process is shown below.

```
repeat
    flag [i] = true;
    turn = j;
    while (P) do no-op;
    Enter critical section, perform actions, then exit critical section
        flag [i] = false;
        Perform other noncritical section actions.
    until false;
```

For the program to guarantee mutual exclusion, the predicate P in the while loop should be

[GATE 2001]
[2-Marks]

(a) flag [j] = true and turn = i
(b) flag [j] = true and turn = j
(c) flag [i] = true and turn = j
(d) flag [i] = true and turn = i

Common Data For Questions 65 & 67:

Suppose we want to synchronize two concurrent processes P and Q using binary semaphores S and T. The code for the processes P and Q is shown below.

```
Process P :              Process Q:
    while (1) {              while (1) {
W:                       Y:
    print '0';                print '1';
    print '0';                print '1';
X:                       Z:
    }                        }
```

Synchronization statements can be inserted only at points W, X, Y and Z.

Q.65 Which of the following will always lead to an output string with '001100110011'?

[GATE 2003]
[2-Marks]

(a) P(S) at W, V(T) at X, P(T) at Y, V(T) at Z, S and T initially 1.
(b) P(S) at W, V(T) at X, P(T) at Y, V(S) at Z, S initially 1 and T initially 0.
(c) P(S) at W, V(T) at X, P(T) at Y, V(S) at Z, S and T initially 1
(d) P(S) at W, V(S) at X, P(T) at Y, V(T) at Z, S initially 1 and T initially 0.

Q.66 Which of the following will ensure that the output string never contains a substring of the form 01^n0 or 10^n1 where n is odd?

[GATE 2003]
[2-Marks]

(a) P(S) at W, V(S) at X, P(T) at Y, V(T) at Z, S and T initially 1.
(b) P(S) at W, V(T) at X, P(T) at Y, V(S) at Z, S initially 1.
(c) P(S) at W, V(S) at X, P(S) at Y, V(S) at Z, S initially 1.
(d) V(S) at W, V(T) at X, P(S) at Y, P(T) at Z, S and T initially 1.

Q.67 A uniprocessor computer system only has two process, both of which alternate 10 ms CPU bursts with 90 ms I/O bursts. Both the processes answer created at nearly the same time. The I/O of both processes can proceed in parallel. Which of the following scheduling strategies will result in the least CPU utilization (over a long period of time) for this system?

[GATE 2003]
[2-Marks]

(a) First come fist served scheduling
(b) Shortest remaining time first scheduling
(c) Static priority scheduling with different priorities for the two processes
(d) Round robin scheduling with a time quantum of 5 ms

Q.68 Consider the following statements with respect to user level threads and kernel supported threads. **[GATE 2004]**

[1-Mark]

(i) Context switch is faster with kernel supported threads.

(ii) For user level threads, a system call can block the entire process

(iii) Kernel-supported thread scan be scheduled independently

(iv) User-level threads are transparent to the kernel

Which of the above statements are true?

(a) ii, (iii) and (iv) only

(b) (ii) and (iii) only

(c) (i), and (iii) only

(d) (i) and (ii) only

Q.69 Consider two processes P_1 and P_2 accessing the shared variables X and Y protected by two binary semaphores S_x and S_y respectively, both initialized to 1. P and V denote the usual semaphore operators, semaphore value. The pseudo code of P_1 and P_2 is as follows.

[GATE 2004]

[2-Marks]

P_1:	P_2:
while true do {	while true do {
L_1:.....	L_3:.....
L_2:.....	L_4:.....
X=X+1;	Y=Y+1;
Y=Y−1;	X=Y−1;
$V(S_X)$;	$V(S_Y)$;
$V(S_Y)$; }	$V(S_X)$; }

In order to avoid deadlock, the correct operators at L_1, L_2, L_3 and L_4 are respectively

(a) $P(S_Y)$, $P(S_X)$; $P(S_X)$, $P(S_Y)$

(b) $P(S_X)$, $P(S_Y)$, $P(S_Y)$, $P(S_X)$

(c) $P(S_X)$, $P(S_X)$; $P(S_Y)$, $P(S_Y)$

(d) $P(S_X)$, $P(S_Y)$; $P(S_X)$, $P(S_Y)$

Q.70 In a certain operating system, deadlock prevention is attempted using the following scheme. Each process is assigned a unique timestamp and is restarted with the same timestamp if killed. Let P_n be the process holding a resource R, P_r be a process requesting for the same resource R and $T(P_h)$ and $T(P\hat{r})$ be their timestamps respectively. The decision to wait or preempt one of the processes is based on the following algorithm.

if $T(P_r) < T(P_h)$ then

kill P_r

else

wait

Which one of the following is TRUE?

[IT-GATE 2004]

[2-Marks]

(a) The scheme is deadlock free, but not starvation free

(b) The scheme is not deadlock free, but starvation free

(c) The scheme is neither deadlock free nor starvation free

(d) The scheme is both deadlock free and starvation free

Q.71 The shell command

find. -name passwd -print is executed in/etc directory of a computer system running Unix. Which of the following shell commands will give the same information as the above command when executed in the same directory?

[IT-GATE 2005]

[1 Mark]

(a) ls passwd

(b) cat passwd

(c) grep name passwd

(d) grep print passwd

Q.72 A user level process in Unix traps the signal sent on a Ctrl-C input, and has a signal handling routine that saves appropriate files before terminating the process. When a Ctrl-C input is given to this process, what is the mode in which the signal handling routine executes?

[IT-GATE 2005]

[1 Mark]

(a) kernel mode

(b) superuser mode

(c) privileged mode

(d) user mode

Q.73 Given below is a program which when executed spawns two concurrent processes:

Semaphore X: = 0;

/* Process now forks into concurrent processes P1 & P2 */

P1 : repeat forever	P2 : repeat forever
V(X);	P(X);
Compute;	Compute;
P(X);	V(X);

Consider the following statements about processes P1 and P2:

I. It is possible for process P1 to starve.

II. It is possible for process P2 to starve.

Which of the following holds?

[IT-GATE 2005]

[2-Marks]

(a) Both I and II are true

(b) I is true but II is false

(c) II is true but I is false

(d) Both I and II are false

Q.74 Two concurrent processes P1 and P2 use four shared resources R1, R2, R3 and R4, as shown below.

P1:	P2:
Compute;	Compute;
Use R1;	Use R1;
Use R2;	Use R2;
Use R3;	Use R3;
Use R4;	Use R4;

Both processes are started at the same time, and each resource can be accessed by only one process at a time. The following scheduling constraints exist between the access of resources by the processes:

P2 must complete use of R1 before P1 gets access to R1.

P1 must complete use of R2 before P2 gets access to R2.

P2 must complete use of R3 before P1 gets access to R3.

P1 must complete use of R4 before P2 gets access to R4.

There are no other scheduling constraints between the processes. If only binary semaphores are used to enforce the above scheduling contraints, what is the minimum number of binary semaphores needed?

[IT-GATE 2005]

[2-Marks]

(a) 1

(b) 2

(c) 3

(d) 4

Q.75 An instruction set of a processor has 125 signals which can be divided into 5 groups of mutually exclusive signals as follows:

Group 1: 20 signals, Group 2: 70 signals. Group 3: 2 signals, Group 4: 10 signals, Group 5: 23 signals.

How many bits of the control words can be saved by using vertical microprogramming over horizontal microprogramming?

[IT-GATE 2005]

[2-Marks]

(a) 0

(b) 103

(c) 22

(d) 55

Q.76 Two shared resources R_1 and R_2 are used by processes P_1 and P_2. Each process has a certain priority for accessing each resource. Let T_{ij} denote the priority of P_j for accessing R_j. A process P_j can snatch a resource R_k from process P_j if T_{ik} is greater than T_{jk}.

Given the following

I. $T_{11} < T_{21}$

II. $T_{12} > T_{22}$

III. $T_{11} < T_{21}$

IV. $T_{12} < T_{22}$

Which of the following conditions ensures that P_1 and P_2 can never deadlock?

[IT-GATE 2005]

[2-Marks]

(a) I and IV

(b) II and III

(c) I and II

(d) None of the above

Q.77 Suppose n processes, $P_1,\ldots P_n$ share m identical resource units, which can be reserved and released one at a time. The maximum resource requirement of process P_i is s_i where $s_i > 0$. Which one of the following is a sufficient condition for ensuring that deadlock does not occur?

[GATE 2005]

[2-Marks]

(a) $\forall i,\ s_i < m$

(b) $\forall i,\ s_i < n$

(c) $\sum_{i=1}^{n} s_i < (m+n)$

(d) $\sum_{i=1}^{n} s_i < (m * n)$

Q.78 Consider the following code fragment:

```
if (fork ()= =0)
{a=a+5;print f ("%d, %d\n",a &a);}
else {a=a–5;print f("%d, %d\n"a, &a);}
```

Let u, v be the values printed by the parent process, and x, y the values printed by the child process. Which one of the following is TRUE?

[GATE 2005]

[2-Marks]

(a) $u = x + 10$ and $v = y$

(b) $u = x + 10$ and $v \neq y$

(c) $u + 10 = x$ and $v = y$

(d) $u + 10 = x$ and $v \neq y$

Q.79 Consider three CPU intensive processes which require 10,20 and 30 time units and arrive at times 0,2 and 6 respectively. How many context switches are needed if the operating system implements a shortest remaining time first scheduling algorithm? Do not count the context switches at time zero and at the end.

[GATE 2006]

[1-Mark]

(a) 1

(b) 2

(c) 3

(d) 4

Q.80 The atomic fetch and set x, y instruction unconditionally sets the memory location x to 1 and fetches the old value of x and y without allowing any intervening access to the memory location. Consider the following implementation of P and V functions on a binary semaphore S.

```
void P (binary_semaphore *s) {
unsigned y;
unsigned * x = & (s → value);
    do {
        fetch-and-set x, y;
        }
        while (y);
    }
void V(binary_semaphore * s) {
    S → value = 0;
}
```

which one of the following is true?

[GATE 2006]

[2-Marks]

(a) The implementation may not work if context switching is disabled in P

(b) Instead using fetch and set, a pair of normal load/store can be used.

(c) The implement of V is wrong

(d) The code does not implement a binary semaphore

Q.81 Consider three processor (process id 0,1,2 respectively) with compute time bursts 2,4 and 8 time units. All processes arrive at ties zero. Consider the longest remaining time first (LRTF) scheduling algorithm. In LRTF times are broken by giving priority to the process with the lowest process id. The average turn around time is

[GATE 2006]

[2-Marks]

(a) 13 units

(b) 14 units

(c) 15 units

(d) 16 units

Common Data For Questions 82 & 83:

Barrier is a synchronization construct where a set of processes synchronizes globally i.e. each process in the set arrives at the barrier and waits for all others to arrive and then all process leave the barrier. Let the number of processes in the set be three and S be a binary semaphore with the usual P and V functions.

Consider the following C implementation of a barrier with line number shown on left.

```
void barrier (void) {
1.  :  P(S);
2.  :  process_ arrived ++;
3.  :  V(S);
4.  :  while (process_arrived !=3);
5.  :  P(S);
6.  :  process_left ++;
7.  :  if (process_left ==3)
8.  :  process_arrived = 0;
9.  :  proces_left = 0;
10. :  }
11  :  V(S);
}
```

The variables process_arrived and process_left are shared among all processes and are initialized to zero. In a concurrent program all the three processes call the barrier function when they need to synchronize globally.

Q.82 The above implementation of barrier is incorrect. Which one of the following is true?

[GATE 2006]

[2-Marks]

(a) The barrier implementation is wrong due to the use of binary semaphore S.

(b) The barrier implementation may lead to a deadlock if two barrier invocations are used in immediate succession.

(c) Lines 6 to 10 need not be inside a critical section

(d) The barrier implementation is correct if there are only two processes instead of three.

Q.83 Which one of the following rectifies the problem in the implementation? **[GATE 2006]**

[2-Marks]

(a) Lines 6 to 10 are simply replaced by process_arrived

(b) At the beginning of the barrier the first process to enter the barrier waits until process_arrived becomes zero before proceeding to execute P(S)

(c) Context switch is disabled at the beginning of the barrier and re enabled at the end

(d) The variable process_left is made private instead of shared.

Q.84 Consider three processes, all arriving at time zero, with total execution time of 10,20 and 30 units, respectively. Each process spends the first 20% of execution time doing I/O, the next 70% of time doing computation, and the last 10% of time doing I/O again. The operating system uses a shortest remaining time first scheduling algorithm and schedules a new process either when the running processes get blocked on I/O or when the running process finishes its compute burst. Assume that all I/O operations can be overlapped as much as possible. For what percentage of time does the CPU remain idle?

[GATE 2006]

[2-Marks]

(a) 0%

(b) 10.6%

(c) 30.0%

(d) 89.4%

Q.85 Consider the following snapshot of a system running n processes. Process i is holding X_i instances of a resource R, for $1 \leq i \leq n$. Currently, all instances of R are occupied. Further, for all i, process i has placed a request for an additional Y_i instances while holding the X_i instances it already has. There are exactly two processes p and q such that $Y_p = Y_q = 0$. When one of the following can serve as a necessary condition to guarantee that the system is not approaching a deadlock?

[GATE 2006]

[2-Marks]

(a) $\min(X_p, X_q) < \max_{k \neq p,q} Y_k$

(b) $X_p, X_q < \max_{k \neq p,q} Y_k$

(c) $\min(X_p, X_q) < 1$

(d) $\min(X_p, X_q) < 1$

Q.86 An operating system uses Shortest Remaining Time first (SRT) process scheduling algorithm. Consider the arrival times and execution times for the following processes.

[GATE 2007]

[2-Marks]

Process	Execution time	Arrival time
P1	20	0
P2	25	15
P3	10	30
P4	15	45

What is the total waiting time for process P2?

(a) 5

(b) 15

(c) 40

(d) 55

Q.87 A single processor system has three resource types X, Y and Z which are shared by three processes. There are 5 units of each resource type. Consider the following scenario, where the column **alloc** denotes the number of units of each resource type allocated to each process, and the column **request** denotes the number of units of each resource type requested by a process in order to complete execution. Which of these processes will finish LAST?

	alloc	request
	XYZ	XYZ
P_0	121	103
P_1	201	012
P_2	221	120

[GATE 2007]

[2-Marks]

(a) P0

(b) P1

(c) P2

(d) None of the above, the since the system is in a deadlock

Q.88 Consider the following statements about user level threads and kernel level threads. Which one of the following statements is FALSE?

[GATE 2007]

[1-Mark]

(a) Context switch time is longer for kernel level threads than for user level threads.

(b) User level threads do not need any hardware support

(c) Related kernel level thread can be scheduled on different processor in a multiprocessor system

(d) Blocking one kernel level thread blocks all related threads

Q.89 Two processes, P1 and P2, need to access a critical section of code. Consider the following synchronization construct used by the processes:

```
/*P1 */
while(ture) {
        wants 1=true;
        while(wants 2==true);
        /* Critical
        section */
        wants 1= false;
        }
        /* Remainder section */
```

```
/* P2  */
        while (true) {
        wants 2=true;
        while (wants 1==ture);
        /* crifical
        section  */
        wants 2 = false;
        }
        /* Remainder section */
```

Here, wants 1 and wants 2 are shared variables, Which initialized to false. Which one of the following statement is TRUE about the above construct? **[GATE 2007]**

[2-Marks]

(a) It does not ensure mutual exclusion.

(b) It does not ensure bounded waiting

(c) It requires that processes enter the critical section strict alternation.

(d) It does not prevent deadlocks, but ensures mutual exclusion.

Q.90 Which of the following is NOT true of deadlock prevention and deadlock avoidance schemes?

[GATE 2008]

[2-Marks]

(a) In deadlock prevention, the request for resources is always granted if the resulting state is safe

(b) In deadlock avoidance, the request for resources is always granted if the resultmg state is safe.

(c) Deadlock avoidance is less restrictive than deadlock prevention

(d) Deadlock avoidance requires knowledge of resource requirements a priori

Q.91 The P and V operations on counting semaphore, where s is a counting semaphore, are defined as follows:

P(s): s = s – 1;

if s<0 then wait;

V(s): s = s + 1;

if s <= 0 then wakeup a process waiting on s;

Assume that P_b and V_b the wait and signal operations on binary semaphores are provided. Two binary semaphores X_b and Y_b are used to implement the semaphore operations P (s) and V(s) as follows:

[GATE 2008]

[2-Marks]

P(s) $P_b(X_b)$;

s=s–1;

if (s<0) {

V_b (X_b);

P_b (Y_b) ;

}

else $V_b(X_b)$;

V(s): $P_b(X_b)$;

s=s+1;

if (s<=0)

V_b (Y_b) ;

V_b (X_b);

The initial values of x_b and y_b are respectively.

(a) 0 and 0

(b) 0 and 1

(c) 1 and 0

(d) 1 and 1

Q.92 The enter_ CS () and leave_ CS () functions to implement critical section of a process are realized using test and set instruction as follows:

```
Void enter _CS(X)
{

    while (test and set (X))
}
Void leave_CS(X)
{
    X=0;
}
```

In the above solution, X is a memory location associated with the CS and is initialized to 0. Now consider the following statements

I. The above solution to CS problem is deadlock free

II. The solution is starvation free

III. The processes enter CS in FIFO order

IV. More than one process can enter CS at the same time.

Which of the above statement(s) is(are) TRUE?

[GATE 2009]

[2-Marks]

(a) I only

(b) I and II

(c) II and III

(d) IV only

Q.93 Consider a system with 4 types of resource R1 (3 units), R2 (2 units), R3 (3 units), R4 (2 units). A non preemptive resource allocation policy is used. At any given instance, a request is not entertained if it cannot be completely satisfied. Three processes P1, P2, P3 request the resources as follows if executed independently.

Process P1:	Process P2:	Process P3:
t=0: requests 2 units of R2 t=1 requests 1 unit of R3 t=3: requests 2 units R1 t=5 releases 1 unit of R2 and 1 unit of R1 t=7:releases 1unit of R3 t=8: requests 2 units of R4 t =10 : Finishes	t=0 requests 2 units of R3 t=2 requests 1 unit of R4 t=4: requests 1 unit R1 t=6 releases 1 unit of R3 t =8 : Finishes	t=0 requests 1 unit of R4 t=2 requests 2 units of R1 t=5: releases 2 units of R1 t=7 requests 1 unit of R2 t=8: requests 1unit of R3 t =9: Finishes

which one of the following statements is TRUE if all three processes run concurrently starting at time t = 0?

[GATE 2009]

[2-Marks]

(a) All processes will finish without any deadlock

(b) Only P1 and P2 will be in deadlock

(c) Only P1 and P3 will be in deadlock

(d) All three processes will be in deadlock

ANSWER KEY

1	b	2	a	3	d	4	b	5	c
6	d	7	b	8	c	9	a	10	c
11	c	12	d	13	a	14	b	15	b
16	c	17	c	18	d	19	b	20	c
21	a	22	b	23	a	24	b, d	25	c
26	b	27	c	28	a	29	a	30	c
31	a	32	b	33	c	34	c	35	c
36	b	37	a	38	b	39	c	40	b
41	c	42	c	43	c	44	a	45	a
46	b	47	d	48	d	49	c	50	b
51	c	52	d, e	53	b	54	a, b	55	c
56	d	57	b	58	b	59	b	60	b
61	c	62	c	63	b	64	b	65	b
66	c	67	d	68	b	69	d	70	a
71	a	72	a	73	d	74	b	75	b
76	c	77	c	78	b	79	b	80	a
81	a	82	b	83	b	84	d	85	a
86	b	87	c	88	d	89	d	90	c
91	c	92	a	93	a				

SOLUTIONS

S.1 (b)

Windows 95 is **multitasking**, multiprocessing but not multiuser.

S.2 (a)

The main memory maintains a buffer to hold the kernel and dispatcher. **The program execution takes place in RAM.** The dispactcher is the module that gives control of the CPU to the process selected by the short-term schedules.

S.3 (d)

Since in multilevel priority scheduling the priority of the processes is fixed.

S.4 (b)

A sharable code is Re-entrant, that the process accessing the shared data has to enter the critical section. Re-entrant code is commonly required in operating systems if the applications are intended to be shared in multi-use systems. A programmer writes a re-entrant program by making sure that no instructions modify the content of variable values in other instructions within the program.

S.5 (c)

Busy waiting or polling concept means that the process continuously checks for the required value in a variable or an event, thereby wasting CPU cycles; as no useful work is being done.

S.7 (b)

If the processes are synchronized then why do they need a synchronizing variable like a semaphore.

S.8 (c)

In message passing system four points of synchronization are possible. These are:

(a) At the sender process

(b) At the communication protocol on the sender end

(c) At the communication protocol on the receiver end

(d) At the receiver process

S.9 (a)

Mailbox is a buffer space reserved for a particular registered user at the Mail Server. It **enables the sender and receiver processes to communicate independently.**

S.10 (c)

Starvation leads to allocation of resources to processes after an unacceptable delay; while in deadlock the processes never get the resources.

S.11 (c)

Disadvantages of deadlock prevention is:

1. Inefficient
2. Delays process initiation/increased waiting time.
3. Subject to cylic restart
4. Disallows incremental resource requests.

S.12 (d)

CPU scheduling may take place under (a) (b) (c) and also when a process terminates.

S.13 (a)

Dispatchers are involved in the CPU scheduling function. The dispatcher is the module that gives control of the CPU to the process selected by the short term scheduler.

S.15 (b)

Kernel actions when process P_i requests an I/O operation on some device d and when the I/O operation completes, it can be described as follows:

1. Kernel creates an ECB, and initializes it as follows:

   (b) Process awaiting the event = P_i
2. The newly created ECB (Event Control Block) (Let us call it ECB_i) is added to a list of ECB's.
3. The state of P_i changed to blocked and address of ECB_i is put into the 'Event information' field of P_i's PCB.

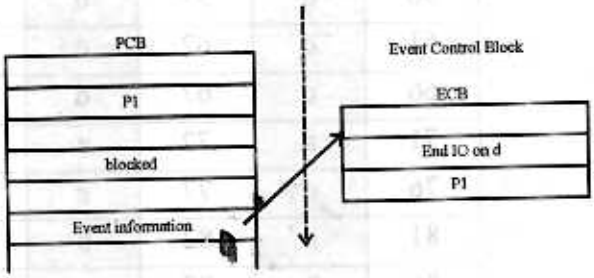


Fig. PCB - ECB interrelationship

4. P_i's PCB may now be shifted to the appropriate scheduling list.
5. When the interrupt 'End of I/O on device d' is raised, ECB_i is located by searching for an ECB with a matching event description field.
6. PCB of P_i is located from ECB_i. **State of P_i is now changed to ready.**

S.16 (c)

The PCB block is not maintained by the CPU but it is maintained by OS, one for each process. The list of blocked processes is not maintained in the priority order by the OS.

S.17 (c)

In a batch environment, a process is **created in response to the submission of a job** whereas in an interactive environment, a process is created when a new user attempts to log.

S.18 (d)

In time sharing system, the process for a particular user is to be terminated **when the user logs off or turns off his or her terminal.**

Option (a) is not related to time sharing system whereas (b) and (c) are the reasons for process termination but not for time sharing system.

S.19 (b)

In UNIX, an I/O bound process is given higher priority than a CPU-bound process, so it can issue its next I/O request quickly. This strategy maximizes the amount of parallelism, keeping both the disk and the CPU busy.

Which process to run next depends on the scheduling algorithm. So it is up to the scheduler, to choose the most important one. **Though in unix, I/O-bound processes are given higher priority than CPU-bound processes.**

S.20 (c)

The empty semaphore is used for counting the number of slots that are empty, and has initial value equal to **the number of slots in the buffer.**

S.21 (a)

Option (a) is correct option. These are reasons for process creation.

In option (b) privileged instruction is reason for process termination.

In option (c) option (ii) is reason for process termination.

S.22 (b)

The Shortest-Remaining-Time-First is same as preemptive SJF scheduling.

A preemptive version of shortest job first is shortest remaining time next.

S.25 (c)

If a process terminates either normally or abnormally then all its children must be terminated. This is called cascading termination.

S.28 (a)

At least one process will be holding 2 resources. In case of a simultaneous demand from all the processes that process will release the 2 resources, thereby **avoiding possible deadlock.**

S.29 (a)

Using Banker's algorithm, one can show that one process has to acquire all its needed resources. This process, after completing its task, will release all its resources, thereby **avoiding any possible deadlock.**

S.30 (c)

There is no limit to the number of processes that can be in the ready and blocked states. At most, **n processes may be in the run state,** since a process in the run state must be allocated to a CPU, and there are only n CPUs.

S.31 (a)

There can be zero processes in any of the three states. It is possible for all the processes to be blocked while waiting on I/O operations, leaving no processes in the ready and running states. All processes can be either in the ready or run state, leaving no processes blocked.

S.32 (b)

This is combinatorics problem of how many ways n objects can be ordered. When selecting the first object, one has n possible choices. For the second object, n – 1.

Each subsequent selection involves one less choice. The total number of possibilities is computed by multiplying all the possibilities at each point, making the answer **n!**.

S.33 (c)

Processes will run until they block. For each cycle, s units of overhead will be needed to accomplish r units of useful work. CPU efficiency is $\frac{r}{(r+s)}$.

S.34 (c)

Since processes will still run until they block, the answer is the same as for Q.33.

S.35 (c)

The number of switches required will be r/q, making the time wasted on switches sr/q. CPU efficiency is $\frac{r}{(r+sr/q)} = \frac{q}{q+s}$

S.36 (b)

Same answer as above except with q = s, the equation evaluates to $\frac{1}{2}$.

S.37 (a)

Using the equation impart Q.35, as q goes to 0. CPU efficiency goes to 0.

S.38 (b)

Since there are two processes, the first process checks the requirement of resource and the second process checks that if the resource is allocated there should not be an unsafe state. m is multiplied since it is done for every resource.

S.39 (c)

In real time system, CPU utilization may range from **40% (for a lightly loaded system) and 90%** (for a heavily loaded system)

S.40 (b)

Option (b) is not correct as like other processes P_n has to wait for a resource that is held by P_o.

Option (a) $\Rightarrow$ Deadlock definition

Option (c) $\Rightarrow$ No preemption condition for deadlock

Option (d) $\Rightarrow$ Since the circular wait condition implies the hold and wait, so the four conditions are not completely independent.

S.42 (c)

(i) $\Rightarrow$ is a solution according to protocol that can be used requires each process to request and the allocated all of its resources before it begins execution.

(ii) $\Rightarrow$ is a solution according to protocol, that allows a process to request resources only when it has none.

A process may request some resources, however it must release all the resources that are currently allocated to it.

S.43 (c)

Each P operation will decrease the semaphore value by 1 and each V operation was increase the semaphore by 1. If x value is 18, then 7 P operations will make the semaphore value 0. If this is followed by 7 V operations the value comes back to 7.

Now, again 7P operation will (total 14P operations) will make the value of semaphore to 0. This value will become Y again after 7V operations (total 14V operations). Now, 6P operations and 4V operations are remaining.

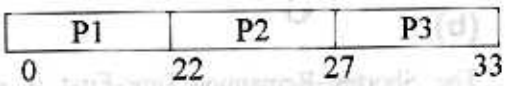
After 6P operations (total 20P operations) the value of semaphore is 1.

This will become 5 when remaining 4V operations (total 18V operations) are applied.

Final value of semaphore is 5, so the value of X is 18.

S.44 (a)

Gantt chart for the given situation is:



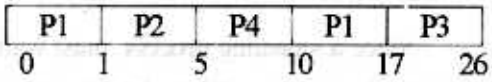
Now, the waiting time is 0 milliseconds for process P1, 22 unit for process P2 and 27 unit for process P3.

Thus, the average waiting time = $\frac{0+22+27}{3}$

$= \frac{49}{3} = 16.33 \equiv$ **16 units.**

S.45 (a)

The SRTF scheduling (or Preemptive SJF) is as depicted in the following Gantt chart,



Process P1 is started at time 0, since it is the only process in the queue. Process P2 arrives at time 1. The remaining time for Process P1 (7 ms) is larger than the time required by the process P2 (4ms.)

So process P1 is preempted and process P2 is scheduled.

$(10 - 1) + (1 - 1) + (17 - 2) + (5 - 3)$

$= \frac{9+15+2}{4} = \frac{26}{4} = \mathbf{6.5\ ms}$

S.46 (b)

If it takes 10 msec to decide to execute a process for 100 msec, then $\frac{10}{(100+10)} \simeq \mathbf{9\%}$ (approx) of the CPU is being used (or wasted) simply for scheduling the work.

S.47 (d)

$$\text{Throughput} = \frac{n}{\max(Ci) - \min(Ai)} = \frac{5}{19-2}$$

$$= \frac{5}{17} = \mathbf{0.2941}$$

where, n → number of jobs in the batch

Ci → Job completion time

Ai → Job arrival time.

S.48 (d)

A program segment where shared resources are accessed is called the critical region. If we could arrange matters such that no two processes were ever in their critical region at the same time, we could avoid races.

S.49 (c)

A link editor is a program that matches the parameters of subroutine definition with the location of parameters of subroutine call. One disadvantage of the direct linking loader is that it is necessary to allocate, relocate, link and load all of the subroutines each time in order to execute a program. Since there may be tens and often hundreds of subroutine involved, especially when we include utility routines such as SQRT, etc. this loading process can be extremely time consuming.

These problems can be solved by dividing the loading process into two separate programs a binder and a module loader. A more sophisticated binder, called linkage editor, can keep track of relocation information so that resulting load module, as an ensemble, can be further relocated and there by loaded anywhere in the core.

S.50 (b)

Each P operation will decrease the semaphore value by 1 and V operation increases the value by 1. At particular time of computation the value of semaphore is 7.

After 7P operation value of semaphore is 0.

After 7V operation value of semaphore is 7 again.

After 14P operation (i.e. after 7P again 7P operation) value of semaphore is 0.

After 14V operation (i.e. additional 7V operations) makes value of semaphore 7.

After this if we perform additional 6P making total of 20P operations makes the value of semaphore 1.

Now if we perform 1V operation.

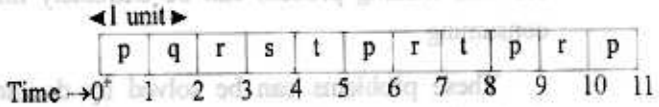
i.e. Total of 15V operation value of semaphore is $1 + 1 = 2$

$\therefore$ Resulting value of semaphore after 20P operations and 15V operations were completed on semaphore is **2**.

S.51 (c)

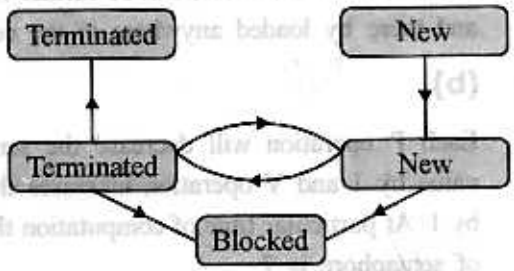
Job Id	CPU Burst time
p	4
q	1
r	8
s	1
t	2

The jobs are arrived at time 0^+ and in the order p, q, r, s, t. scheduling algorithm is Round Robin with time slice of 1 unit.



∴ **The departure time (completion time) for job P is 11.**

S.53 (b)



When process is created it is in the NEW state.

Ready: (runnable, temporarily stopped to let another process run)

Running (actually using the CPU at that instant)

Blocked (Unable to run until some external event happens)

Therefore it is an operating system with a preemptive scheduler.

S.54 (a, b)

A solution to the Dining Philosophers Problem which avoids deadlock is: ensure that all philosophers pick up the right fork before the left fork, and viceversa so option a and b are correct.

S.55 (c)

A critical section is a program segment where shared resources are accessed.

S.56 (d)

A process must execute wait (mutex) before entering the monitor and must execute signal (mutex) after leaving the monitor.

Since a signaling process, must wait until the resumed process either leaves or waits, an additional semaphore next is introduced, initialized to 0, on which the signaling process may suspend themselves.

P = 1 to 9.

If P = 9 (for max)

then **P (mutex) = 8**

then **V (mutex) = 8 + 1 = 9**

S.57 (b)

Number of users = 3

Each requires resourcesj = 2

So, **minimum number of resources for no dead lock is 4** because if there are 4 resources one of the 3 user processes will get the fourth instance of the resource and relinguish one or both of the resource (s) it is currently holding after using.

S.58 (b)

Tape, drive = 6

Each process needs = 2 drives

There are n number of processes

For deadlock free, n = 6 – 1

= 5.

With 3 processes each can have 2 drives. With four processes the distribution will be (2, 2, 1, 1), allowing the first 2 processes to finitsh. With 5 processes, the distribution will be (2, 1, 1, 1) which still allows the first to finish with six, each holding 1 tape drive and wanting another, we have a deadlock. Thus, for n < 6 the system is deadlock free.

S.59 (b)

Semaphore was initialized to 10. P operation decrease the semaphore value by 1. V operation increases semaphore value by 1.

∴ After 6P operation value of semaphore will be = 4.

After 4V operations value of semaphore will be 4 + 4 = 8.

S.60 (b)

When two or more processes are reading or writing some shared data and the final result depends on who run precisely when, are called **rare conditions**.

S.61 (c)

Loader. A program called loader performs the two functions of loading and link-editing, the process of loading consists of taking *relocatable machine code*, altering the relocatable addresses and placing the altered instruction and data in memory at the proper locations.

Link editor. The link-editor allows us to make a single program from several files of relocatable machine code. These files may have been result of several different compilations and one or more may be library files of routines provided by the system and available to any program that needs them.

S.62 (c)

Never request a resource after releasing any resource is NOT a valid deadlock prevention scheme.

S.63 (b)

Translation lookaside buffer (TLB) need not necessarily be saved on a context switch between processes. Switching from one process to another requires a certain amount of time for doing the administration-saving and loading register and memory maps, updating various table and list, flushing and reloading the memory cache, etc.

S.64 (b)

Basically, Peterson's algorithm provides guaranteed mutual exclusion by using the following two constructs-flag [ ] and turn. Flag [ ] controls the willingness of a process to be entred in critical section. While turn control the process that is allowed to be entered in critical section so, by replacing P with the following:

flag [j] = true and turn = j

Process i will not enter critical section if process j wants to enter critical section and it is process 'j' s turn to enter critical section. The same concept can be extended for more than two process.

S.65 (b)

To get the expected output. The sequence of process is **P, Q, P, Q, P, Q** and so on we must initialize **S =1 and T = 0.**

S.66 (c)

If the output string never contains a substring of the form 01^n0 or 10^n1 where n is odd then the procedures becomes.

S = 1

Process P:	Process Q:
while (1){	while (1) {
W: P(S)	**Y: P (S)**
print'0';	print '1';
print '0';	print ' 1';
X: V(S)	**Z: V(S)**
}	}

S.67 (d)

If two processes do both I/O and computing (CPU bursts) then **round robin scheduling produces maximum CPU utilization** whereas FCFS produces least CPU utilization.

S.68 (b)

Let us consider each statement separately:

(i) Statement is false because there is no connection between kernel supported threads and context switch.

(ii) Statement is true because it is the drawback of user level threads that blocking system call can block the entire process.

(iii) Statement is true because kernel supported threads having own memory areas are scheduled independently by the operating system.

(iv) Statement is false because kernel is unaware about user level threads and there is no kernel support to user level threads.

S.69 (d)

S_x and S_y are two binary semaphore if assume P means wait or V means signal then for two process P_1 and P_2 we take alternate then there in no chance of dead lock.

The code is as follows:

P_1	P_2
while true do {	while true do {
L_1: P(S_x)	**L_3: P(S_x)**
L_2: P(S_y)	**L_4: P(S_y)**
x=x + 1	y=y+1;
y=y−1;	x=y−1;
V(S_x);	V(S_y);
V(S_y); }	V(S_x); }

S.70 (a)

This is dead lock free but cause starvation.

S.71 (a)

The shell command executed in /etc is

find -- name passwd – print described as

Find	Name	Passwd	Print
↑	↑	↑	↑
For current directory	Option for specify what we have to match here (-name means match name)	(name of file)	What to do when matched (Print on unix) (Only name is printed)

same work is performed by the command **ls passwd.**

S.72 (a)

The only signals that can't be catch by 440 processor are

SIGKILL

SIGTOP

SIGKILL is signal which is generated by Ctrl-C is passed and it operates in kernel mode.

S.73 (d)

Here P_1 always 'signal' first. Now, when P1 gets executed it may starve on P(X), waiting for (X). Until this situation semaphore status is '1' done by P1. Now, suppose when P1 is executing and semaphore status in reversed to '0', waiting is done by P2 not P1, or we can say that after signal V(X) in P1, context switch occur and P2 gets chance, i.e. after executing wait P(X) it's status is blocked, in that case P1 gets the (X) and no starving occurs in P1. Similarly, for P2. Hence, **both P1 and P2 will not starve.**

S.74 (b)

P1	P2
Wait (X)	Use R1
Use R1	Signal (X)
Use R1	Wait (Y)
Signal (Y)	Use R2
Wait (X)	Use R3
Use R3	Signal (Y)
Use R4	Wait (Y)
Signal (Y)	Use R4

S.75 (b)

Horizontal microprogramming: There is need of 1 bit for each control signal.

Vertical microprogramming: It uses borrower microinstruction word width.

Ex. 512 distinct microinstructions then

$512 = 2^9$, so 9 bits needed to encode control word.

So in horizontal microprogramming needed bits are

20 + 70 + 2 + 10 + 23 = 125 bit

In vertical microprogramming.

Group 1: 20 signals ⇒ 5 bits

Group 2: 70 signals ⇒ 7 bits

Group 3: 2 signals ⇒ 1 bits

Group 4: 10 signals ⇒ 4 bits

Group 5: 23 signals ⇒ 5 bits

Total bits = 5 + 7 + 1 + 4 + 5 = 22 bits

So bits saved by vertical microprogramming are

= 125 -- 22

= 103 bits

S.76 (c)

if $T_{11} < T_{21}$

$T_{12} > T_{22}$ then

P1	P2
required R_1	required R_1
required R_2	required R_2

Assume that P1 is running and after a context switch P2 goes to running that it snatch resource from P1, because for R_2 it have high priority both process not required another resource hence deadlock. So option (a) & (b) are incorrect.

S.77 (c)

There are n processes P_1, P_2...,P_n

Number of identical resources = m

The maximum resource requirement for a process P_2 is S_p where $S_i > 0$

If all processes are in safe state or system can allocate resources to each processes P_i in some order and still avoid a dead lock. We can avoid the dead lock if each P_i can still request can be satisfied by the current available resources plus the resources held by processes P_j such that $j < i$ the maximum resource requirement of n processes is $\sum_{i=1}^{n} s_i$ and this amount must be less than m +n

So $\sum_{i=1}^{n} s_i < (m+n) = 1$.

S.78 (b)

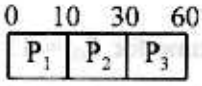
Fork ( ) creates separate copy for both parent and child. Hence, parent of 'a' and child of 'a' are different, hence v ≠ y. Now, fork ( ) returns '0' to child and non zero to parent, hence value of 'a' in child will be 10 more than that of parent. **Hence, u = x + 10 and v ≠ y is the right choice.**

S.79 (b)

Let three processes are P_1, P_2 and P_3

Process	Arrival Time	Burst time
P_1	0	10
P_2	2	20
P_3	6	30

The Gannt chart for SRTF scheduling algorithm is



Context-switches are needed only at time 10 and 30.

S.80 (a)

The P and V function is as follows:

```
1. void P (binary_semaphore *S) {
2. unsigned y;
3. unsigned* x = and &(S →value);}
4. do {
5. fetch and set x,y
6.          }    while (y) ;
```

7. }

8. void V (binary_semaphore*S) {

9. {S→value=0;}

10. }

fetch and set instruction always sets the memory location x=1 and fetches the old value of x into y. The binary semaphore *S takes only two value either 0 and 1. When we initialize S = 0 then in and fetch and set instruction change the value of x=0 to x=1 and y becomes 0. **If there are more than two process and context switching between processes is disabled in P then this implementation doesn't work properly and can't synchronize the processes.**

S.81 (a)

Process id	Arrival time	CPU Burst time
P_0 0	0	2
P_1 1	0	4
P_2 2	0	8

Remaining Time	
P_0	2
P_1	4
P_2	8

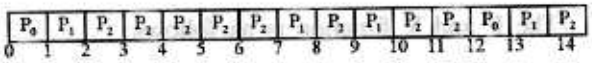
The Gannt chart for LRTF CPU scheduling algorithm.

Turn around time for P_0 = 13 – 0 = 13

Turn around time for P_1 = 14 – 1 = 13

Turn around time for P_2 = 15 – 2 = 13

Average turn around time = 39/3 = 13



So option (a) is correct.

S.82 (b)

The barrier implementation may lead to a dead lock if two barrier invocations are used in immediate succession which is due to line 3 and 7.

S.83 (b)

At the beginning of the barrier the first process to enter in line 8 the barrier waits until process_arrived becomes zero before proceeding to execute P (S).

S.85 (a)

Total number of processes = n

Process i is holding X_i instances of a resource R for $1 \le i \le n$

Process i holding resource X_i and additional required Y_i

There are two processes p and q such that $y_p = Y_q = 0$

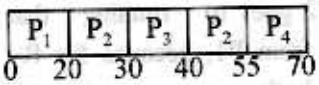
For no deadlock occurrence

min $(X_p, X_q) < (\max_{k \neq p,q} Yk)$

S.86 (b)

Process	Execution time	Arrival Time
P_1	20	0
P_2	25	15
P_3	10	30
P_4	15	45

The Gannt chart for SRT scheduling algorithm is



So the waiting time for process P2

= complection time- arrival time- execution time

= 55 – 15 – 25

= **15**

S.87 (c)

Consider the matrix

	alloc	request	
	XYZ	XYZ	
			XYZ
P_0	121	103	Available (0, 1, 2)
P_1	201	012	
P_2	221	120	

so the available matrix is (012) and the requirement of process P_1 is (0, 1, 2) so P_1 can use the available resources and after releasing the resource the matrix becomes.

	alloc	request	
	XYZ	XYZ	
			XYZ
P_0	121	103	Available (2, 1, 3)
P_1	000	000	
P_2	221	120	

In this time available matrix is (213) and request matrix of P_0 is (103). So, P_0 release the resource and the available matrix becomes (3,3,4). Now, P_2 will use the resource. **So P_2 will finish last.**

S.88 (d)

In kernel level threads the in blocking system call causes, the kernel can schedule another thread in the application for execution. So statement (d) is false about kernel level threads.

S.89 (d)

(a) P1 has control of the critical section provided wants 1 is true and wants 2 is false. P2 has control of the critical section provides wants 2 is true and wants 1 is false. So, if P1 has control it excludes P2 till it completes and vice-versa, so mutual execlusion is ensured. This eliminates choice (a).

(b) Option is false as the time spent by P1 and P2 in their critical sections is not controlled or bounded.

(c) Option is not correct for one can easily see that P1 can use the resource, release it, use it again, release it and so on without P2 over demanding it.

(d) A deadlock can arise as the assaignment to wants 1 and wants 2 is not done as an induisible operation. So, when wants 1 is set to true at the same time wants 2 can be set to true. This results in endless waiting.

S.91 (c)

In given code P(S) decrement the value of semaphore and V(S) increment the value of semaphore. P_b and V_b are wait and signal operations on binary semaphore. X_b and V_b are binary. Semaphore to avoid the mutual exclusion condition, the **initial value of X_b = 1 and the initial value of Y_b = 0 then P(S) and V(S) will work properly.**

S.93 (a)

Initially available resources

R1 R2 R3 R4

3 2 3 2

	Process	Allocate				Available			
t = 0		R1	R2	R3	R4	R1	R2	R3	R4
P_1: request 2 unit of R2									
	P1	0	2	0	0	3	0	3	2
P_2: request 2 unit of R3									
	P2	0	0	2	0	3	0	1	2
P_3: request 1 unit of R4									
	P3	0	0	0	1	3	0	1	1
t = 1									
P_1: request 1 unit of R3									
	P1	0	2	1	0	3	0	0	1
	P2	0	0	2	0				
	P3	0	0	0	1				
t = 2									
	P1	0	2	1	0	3	0	0	1
P_2: request 1 unit of R4									
	P2	0	0	2	1	3	0	0	0
P_3: request 2 unit of R1									
	P3	2	0	0	1	1	0	0	0
t = 3									
P_1: request 2 unit of R1									
(wait for 1 unit of R1)									
	P1	1	2	1	0	0	0	0	0
	P2	0	0	2	1				
	P3	2	0	0	1				
t = 4									
P_2: request 1 unit of R1									
(wait)									
t = 5									

P_1: release 1 unit of R2

and 1unit of R1

P1	0	1	1	0		1	1	0	0
P2	0	0	2	1					

P_3: release 2 unit of R1

P3	0	0	0	1		3	1	0	0

now request of P_1, 1 unit of R1 will be granted and request of P_2, 1 unit of R1 will be granted

P1	1	1	1	0		2	1	0	0
P2	1	0	2	1		1	1	0	0
P3	0	0	0	1		1	1	0	0

t = 6

P_2: release 1 unit of R3

P1	1	1	1	0					
P2	1	0	1	1		1	1	1	0
P3	0	0	0	1					

t = 7

P_1: release 1 unit of R3

P1	1	1	0	0		1	0	1	0

P_3: request 1 unit of R2

P2	1	0	1	1					
P3	0	1	0	1		1	0	2	0

t = 8

P_2: finish

P_1: request 2 unit of R4 (wait).

P1	1	0	1	1					

P_3: request 1 unit of R3

P3	0	1	1	1		1	0	1	0

t = 9

P_3: finish

P1	1	0	1	1		1	0	1	0

t = 10

P_1: finish

all process will finish without any deadlock.

7.2 File System

LEVEL–1

Q.1 The following statements is true for sharing
(a) It allows general processes to access the same portions of main memory.
(b) It allows general process to access the different portions of cache memory.
(c) It allows only one process to access the different portions of main memory.
(d) None of the above.

Q.2 If the access to the file is at random, then the following file organization may be most appropriate
(a) sequential file organization
(b) indexed sequential file organization
(c) pipe file organization
(d) indexed file organization

Q.3 File management system is typically viewed as a system services that itself is
(a) a part of the operating system
(b) served by operating system
(c) available in the main memory
(d) an independent unit

Q.4 Ordinarily File - descriptors are maintained on
(a) secondary storage
(b) primary storage
(c) registers
(d) disk

Q.5 File descriptors are brought to primary storage when
(a) file is opened
(b) file is created
(c) file is modified
(d) file is destroyed
(e) All the above

Q.6 When files & records are added and deleted disk tend to become severely?
(a) fragmented
(b) reorganized
(c) segmented
(d) none of these

Q.7 Which one is type of file attribute?
(i) Maximum size
(ii) Key position
(iii) Key length
(iv) Time of last change
(a) (i), (ii)
(b) (i), (ii), (iii), (iv)
(c) (iii), (iv)
(d) (ii), (iii) (iv)

Q.8 What will happen when we allocate small units to files?

(a) Reading of file will be slow

(b) No rotational delay

(c) No seek delay

(d) All of above

Q.9 While selecting a file organization, for a file which is stored on CD-ROM, which one is least important criteria.

(a) Ease of update

(b) Rapid access

(c) Simple maintenance

(d) Economy of storage

Q.10 For doing read / write operation, user must set :

(a) Read / Write, 0

(b) Read / Write, 1

(c) Read-Only, 1

(d) Read-Only, 0

Q.11 Which statement is incorrect in case of file?

(a) File represents both source program and object forms and data.

(b) A file is a collection of related information defined by its creator.

(c) Files are mapped by the operating system, onto physical devices

(d) A file is a sequence of bits, lines or records whose meaning is defined by its creator and user.

Q.12 The function of archive file type is

(a) compiled, machine language not linked

(b) read to run machine language program

(c) commands to the command interpreter.

(d) grouping of related files into one file, for storage

Q.13 To read a block of some bytes will require a time which is the sum of

(a) seek, rotational delay, transfer times

(b) seek, rotational delay

(c) seek delay, transfer times

(d) rotational delay, transfer times

Q.14 Which system is widely used on the large main frame computers used in commercial data processing.

(a) Tree

(b) Record sequence

(c) Byte sequence

(d) None of these

Q.15 A sequential access file has fixed-size 15 - byte records. Assuming the first record is record 1. The first byte of record 5 will be at what logical location?

(a) 41

(b) 51

(c) 61

(d) None

Q.16 'Aging' is

(a) keeping track of cache contents

(b) keeping track of waht pages are currently residing in the memory.

(c) keeping track of how many times a given page is referenced.

(d) increasing the priority of jobs to ensure termination in a finite time

LEVEL–2

Q.17 A computer system that uses memory-mapped input-output has a 16 - bit address space; address with ones in the two most significant bits refer to devices. What is the maximum amount of memory that can be referenced in such a system?

(a) 2^7

(b) 2^9

(c) 2^{11}

(d) 2^{14}

Common Data For Questions 18 to 20:

On a system using contiguous allocation, find the number of the physical block corresponding to the logical block, given the file is stored at the indicated starting physical block (assume block numbers start with 1).

Q.18 Starting physical block : 1000 ; logical block : 12

(a) 174

(b) 2074

(c) 1011

(d) None

Q.19 Starting physical block : 75 ; logical block : 2000

(a) 174

(b) 2074

(c) 1011

(d) None

Q.20 Starting physical block : 150 ; logical block : 25

(a) 1011

(b) 2074

(c) 174

(d) None

Q.21 Suppose a process creates a file, then as soon as that process is terminated

(a) The file is deleted

(b) The file can be accessed by other processes using some other name.

(c) The file cannot be accessed by other process.

(d) The file continues to exist

Q.22 Which statement is incorrect in case of Hierarchical Directory Systems?

(a) When file id is opened, the operating system searches its directory until it finds the names of all files related to same file extension.

(b) A directory entry holds the file name and a pointer to another data structure where the attributes and disk addresses are found.

(c) A directory typically contains a number of entries one per file, each file entry contains the file name, the file attributes and the disk addresses where the data are stored.

(d) Operating system extracts the attributes and the disk addresses, either directly from the directory entry or from the data structures pointed to and puts them in main memory.

Q.23 Which out of the following files does not provide default access to the unix shell.

(a) Standard input

(b) Standard output

(c) Standard error

(d) None of above

Q.24 If all passwords consiting 7 characters chosen at random from the 95 printable ASCII characters, then search space would be

(a) 7×95

(b) 7^{95}

(c) 95^{7}

(d) $2^{7} \times 95$

Q.25 Which of the following statement or statements are false?

(i) UNIX uses the UNIX file system (UFS) as base

(ii) Most CD-ROMs are written in the High Sierra format which is standard format agreed.

(iii) Windows NT supports disk file system formats of FAT, FAT32 and NTFS or Windows NT file system.

(iv) Master File table in UFS is same as superblock in NTFS.

(a) (ii), (iii) are false

(b) (i), (ii), (iv) are false

(c) Only (i) is false

(d) Only (iv) is false

Q.26 Which of the following statement (s) is/are True?

(i) The second layer in the file-system implementation is the file-system interface.

(ii) The first layer is based on the open, read, write and close calls, and file description.

(iii) Virtual File System (VFS) separates file system-generic operations from their implementation by defining a clean VFS interface.

(iv) The VFS is based on a file-representation structure, called a vnode, that are unique within only a single file system.

(a) (ii), (iii) are true

(b) Only (iv) is true

(c) (i) (iii) are true

(d) (iii), (iv) are true

Q.27 Match the following :

Group 1		Group 2	
1.	Append	(a)	Some attributes can be changed after the file has been created.
2.	Seek	(b)	processes often need to read file attributes to do their work.
3.	Get Attributes	(c)	reposition the pointer from the current position to the specific place in the file.
4.	Set attributes	(d)	restricted form of write call.

(a) 1-c, 2-b, 3-d, 4-a

(b) 1-b, 2-c, 3-a, 4-d

(c) 1-a, 2-b, 3-c, 4-d

(d) 1-d, 2-c, 3-b, 4-a

Q.28 Match the following

Group 1		Group 2	
1.	Text file	(a)	sequence of subroutines, each one is organized as declaration followed by executable statements.
2.	Source file	(b)	sequence of words organized into loader record blocks.
3.	Object file	(c)	sequence of characters organized into lines.

(a) 1-a, 2-b, 3-c

(b) 1-b, 2-a, 3-c

(c) 1-b, 2-c, 3-a

(d) 1-c, 2-a, 3-b

Q.29 Which one is not related to file?

(i) Field

(ii) Record

(iii) Database

(a) (i) and (ii)

(b) Only (ii)

(c) Only (i)

(d) (i), (ii), (iii)

Q.30 A certain moving arm disk storage with one head has following specifications:

```
Number of tracks/ recording surface = 200
Disk rotation speed  = 2400 rpm
Track storage capacity = 62500 bits
```

The average latency time (assume that the head can move from one track to another only by traversing the entire track) is

(a) 2.5 s

(b) 2.9 s

(c) 3.1 s

(d) 3.6 s

Q.31 Which one of the following is a valid Device directory attributes?

(i) Location

(ii) Current position

(iii) Usage count

(iv) Process identification

(a) (i), (ii)

(b) (i), (iii), (iv)

(c) (ii), (iv)

(d) (i), (ii), (iii), (iv)

Q.32 The statements given below are the functions of:

(i) It manages the directory structure to provide the file organization module with the information the latter needs, given a symbolic file name.

(ii) It is also responsible for protection and security

(iii) It manages meta data information

(a) basic file system

(b) file-organization module

(c) application programs

(d) logical file system

Q.33 System wide table contains

(a) Information of the access right to the file

(b) Information regarding the use of the file by the process

(c) Process independent information

(d) Accounting information

Q.34 Match the following correctly:

Group I		Group II	
i.	contiguous Allocation	1.	There is no external fragmentation
ii.	Linked Allocation	2.	Supports direct access, without suffering from external fragmentation
iii.	Indexed Allocation	3.	Suffers from external fragmentation

(a) (i) - 3, (ii) - 2, (iii) - 1
(b) (i) - 1, (ii) - 3, (iii) - 2
(c) (i) -3, (ii) - 1, (iii) - 2
(d) (i) - 2, (ii) - 1, (iii) - 3

LEVEL–3

Q.35 Consider a system in which a directory entry can store up to 16 disk block addresses. For files not larger than blocks, the 16 addresses serve as the file's index table. For files larger than 16 blocks, the addresses point to indirect blocks which in turn point to 256 file blocks each. A block is 1024 bytes. How big can a file be?
(a) 2^7
(b) 2^{11}
(c) 2^{22}
(d) None

Q.36 In a computer system where the 'best fit' algorithm is used for allocating jobs to memory partitions the following situation was encountered:

Partitions sizes in kB	4k	8k	20k					
Jobs sizes in kB	2k	14k	3k	6k	6k	10k	20k	2k
Time for execution	4	10	2	1	4	1	8	6

When will the 20K job complete?
(a) 19 units
(b) 18 units
(c) 8 units
(d) 9 units

GATE QUESTIONS

Q.37 A computer system has 6 tape drives, with n process competing for them. Each process may need 3 tape drives. The maximum value of n for which the system is guaranteed to be deadlock free is **[GATE 1992]**
(a) 2
(b) 3
(c) 4
(d) 1

Q.38 The root directory of a disk should be placed **[GATE 1993]** **[2-Marks]**
(a) at a fixed address in main memory
(b) at a fixed location on the disk
(c) anywhere on the disk
(d) at a fixed location on the system disk
(e) anywhere on the system disk

Q.39 A part of the system software, which under all circumstances must reside in the main memory, is **[GATE 1993]** **[2-Marks]**
(a) text editor
(b) assembler
(c) linker
(d) loader
(e) none of these

Q.40 A linker reads four modules whose lengths are 200, 800, 600 and 500 words, respectively. If they are loaded in that order, what are the relocation constants? **[GATE 1998]** **[1-Mark]**
(a) 0, 200, 500, 600
(b) 0, 200, 1000, 1600
(c) 200, 500, 600, 800
(d) 200, 700, 1300, 2100

Q.41 The minimum number of record movements required to merge five files A (with 10 records), B (with 20 records), C (with 15 records), D (with 5 records) and E (with 25 records) is :

[GATE 1999]

[2-Marks]

(a) 165

(b) 90

(c) 75

(d) 65

Q.42 Listed below are some operating system abstractions (in the left column) and the hardware components (in the right column) ?

[GATE 1999]

[1-Mark]

Group I		Group II	
(i)	Thread	1.	Interrupt
(ii)	Virtual address space	2.	Memory
(iii)	File system	3.	CPU
(iv)	Signal	4.	Disk

(a) (i) – 2, (ii) – 4, (iii) – 3, (iv) – 1

(b) (i) – 1, (ii) – 2, (iii) – 3, (iv) – 4

(c) (i) – 3, (ii) – 2, (iii) – 4, (iv) – 1

(d) (i) – 4, (ii) – 1, (iii) – 2, (iv) – 3

Q.43 Using a larger block size in a fixed block size file system leads to **[Gate 2003]**

[1-Mark]

(a) better disk throughput but poorer disk space utilization.

(b) better disk throughput and better disk space utilization.

(c) poorer disk throughput but better space utilization.

(d) poorer disk throughput and poorer disk space utilization.

Q.44 A Unix style I node has 10 direct pointers and one single, one double and one triple indirect pointers. Disk block size is 1 Kbyte, disk block address is 32 bits, and 48 bit integers are used. What is the maximum possible file size?

[Gate 2004]

[2-Marks]

(a) 2^{24} bytes

(b) 2^{32} bytes

(c) 2^{34} bytes

(d) 2^{48} bytes

Q.45 Which of the following commands or sequences of commands will rename a file x to file y in an Unix system?

I. mv y, x II. mv x, y

III. cp y, x (rm x) IV. cp x, y (rm, x)

[IT-GATE 2004]

[1 Mark]

(a) II and III

(b) II and IV

(c) I and III

(d) II only

Q.46 A student wishes to create symbolic links in a computer system running Unix. Three text files named " file 1" , "file 2", and "file 3" exist in her current working directory and the student has read and write permissions for all three files. Assume that file 1 contains information about her hobbies, file 2 contains information about her friends and file 3 contains information her courses. The student executes the following sequence of commands from her current working directory?

In – s file 1 file 2

In – s file 2 file 3

Which of the following types of information would be lost from her files system

I. Hobbies

II. Friends

III. Courses **[IT-GATE 2005]**

[1 Mark]

(a) I and II only

(b) II and III only

(c) II only

(d) I and III only

Q.47 Group-1 contains some CPU scheduling algorithms and Group-2 contains some applications. Match entries in Group-1 to entries in Group-2

Group 1		Group 2	
P.	Gang Scheduling	1.	Guaranteed Scheduling
Q.	Rate Monotonic Scheduling	2.	Real time Scheduling
R.	Fair Share Scheduling	3.	Thread Scheduling

[Gate 2007]

[1-Mark]

(a) P-3; Q-2; R-1

(b) P-1, Q-2; R-3

(c) P-2; Q-3; R-1

(d) P-1, Q-3; R-2

Q.48 Which of the following system calls results in the sending of SYN packets? **[Gate 2008]**

[1-Mark]

(a) socket

(b) bind

(c) listen

(d) connect

Q.49 The data blocks of a very large file in the Unix file system are allocated using. **[Gate 2008]**

[1-Mark]

(a) Contiguous allocation

(b) Linked allocation

(c) indexed allocation

(d) an extension of indexed allocation

Q.50 Consider a disk system with 100 cylinders. The requests to access the cylinders occurs in following sequence:

4, 34, 10, 7, 19, 73, 2, 15, 6, 20

Assuming that the head is currently at cylinder 50, what is the time taken to satisfy all requests if it takes 1 ms to move from one cylinder to adjacent one and shortest seek time first policy is used?

[Gate 2009]

[2-Marks]

(a) 95 ms

(b) 119 ms

(c) 233 ms

(d) 276 ms

Q.51 In which one of the following page replacement policies, Belady's anomaly may occur?

[Gate 2009]

[1-Mark]

(a) FIFO

(b) Optimal

(c) LRU

(d) MRU

Q.52 The essential content(s) in each entry of page table is/are **[Gate 2009]**

[1-Mark]

(a) Virtual page number

(b) Page frame number

(c) Both virtual page number and page frame number

(d) Access right information

ANSWER KEY

1	a	2	d	3	a	4	a	5	a
6	a	7	b	8	a	9	a	10	d
11	a	12	d	13	a	14	a	15	c
16	d	17	d	18	c	19	b	20	c
21	d	22	a	23	d	24	c	25	d
26	a	27	d	28	d	29	d	30	a
31	d	32	d	33	c	34	c	35	c
36	a	37	a	38	b	39	d	40	b
41	d	42	c	43	a	44	c	45	b
46	b	47	a	48	d	49	d	50	b
51	a	52	b						

SOLUTIONS

S.2 (d)

An **indexed file organization** contains records ordered by a record key. The record key uniquely identifies the record and determines the sequence in which it is accessed with respect to other records. The record access modes allowed for indexed files are sequential, random or dynamic because of record keys. For example, we could access the file through employee department rather than through employee number, i.e. records are accessed according to the value we place in a key field.

S.7 (b)

The file attributes varies considerably from system to system.

Option (b) $\Rightarrow$ (i), (ii), (iii), (iv) are types of file attributes.

(i) **Maximum size** $\Rightarrow$ File may grow to a maximum size.

(ii) **Key position** $\Rightarrow$ offset of the key within each record

(iii) **Key length** $\Rightarrow$ Number of bytes in key fields

(iv) **Time of last change** $\Rightarrow$ Date and time, file was last changed

S.8 (a)

Using small allocation units means that each file will consist of many blocks. Reading each block normally requires a seek and a rotational delay, so **reading a file consisting of many small blocks will be slow.**

S.9 (a)

A file stored on CD-ROM will never be updated and so **ease of update is not an important issue.**

S.10 (d)

For doing read / write operation, user must set **Read - only flag to 0**, and for read only operation it is set to 1.

S.12 (d)

	File type	Function
1.	Object	Compiled, machine language not linked
2.	Executable	Read to run machine language program
3.	**Archive**	**Related files grouped into one file, sometimes compressed, for archiving or storage**
4.	Batch	Commands to the command interpreter

S.13 (a)

Total time = seek time + rotational delay + transfer times.

S.14 (a)

Tree type is widely used on the large main frame computers used in **commercial** processing.

S.15 (c)

Record 5 is preceded by 4 records, each of 15 bytes. The fifth record will start at byte **(4 × 15) + 1 = 61.**

S.17 (d)

At most 2^{14} memory locations can be accessed because among 16-bits, 2 bits are for I/O devices.

S.18 (c)

If logical block 1 is stored in physical block 1000, 2 will be stored in 1001, 3 in 1002 and 12 in 1011. 1000 – 1 + 12 = **1011.**

S.19 (b)

75 – 1 + 2000 = **2074.**

S.20 (c)

150 – 1 + 25 = **174.**

S.21 (d)

When a process create a file, it gives the file a name. When the process terminates, the **file continues to exist**, and can be accessed by other processes using its name.

S.22 (a)

When file id is opened, the operating system searches its directory until it finds the name of the file to be opened **not all files related to same file extension.**

S.23 (d)

A program like the shell does not have to open the terminal in order to read from it or write to it. Instead, when it (or any other program) starts up, it automatically has access to a file called **standard input (for reading)**, a file called **standard output (for writing normal output)** and a file called **standard error (for writing error messaged).**

S.24 (c)

If all passwords consisting of 7 characters chosen at random from the 95 printable ASCII characters, **the search space becomes 95^7,** which is about 7×10^{13}.

S.25 (d)

Most CD-ROMs are written in High Sierra format, which is a Standard format agreed upon by CD-ROM manufactures. Without such a standard, there would be little or no inter-operability between systems trying to use CD-ROMs.

Unix uses the UNIX file system (UFS) as a base. Windows NT supports disk file-system formats of FAT, FAT32 and NTFS (or Window NT file system), as well as CD-ROM, DUD, and floppy-disk file-system formats. **Master file table is in NTFS not in VFS.**

S.26 (a)

The file system implementation consists of three major layers. **The first layer is the file-system interface, based on open, read, write and close calls, and file descriptors.**

The second layer is called the Virtual file System(VFS) layer; it serves two important functions.

(i) **It separates file-system-generic operations from their implementation by defining a clean VFS interface.** Several implementations for the VFS interface may

coexist on the same machine, allowing transparent access to different types of file systems mounted locally.

(ii) **The VFS is based on a file-representation structure, called a vnode, that contains a numerical designator for a network-wide unique file.** (UNIX inodes are unique within only a single file system). This network wide uniqueness is required for support of network file system. The kernel maintains one vnode structure for each active node (file or directory).

S.27 (d)

Append	This call is **restricted form of write call.** It can only add data to the end of the file.
Seek	For random access file, a method is needed to specify from where to take the data. Seek **repositions the pointer from the current positions the specific place in the file.**
Get Attributes	**Processes often need to read file attributes to do their work.**
Set Attributes	Some of the attributes of the file are user settable and **can be changed after the file has been created.**

S.28 (d)

Text file-sequence of characters organized into lines.

Source file-sequence of subroutimes, each one is organized as declaration followed by executable statements.

Object file-sequence of words organized into loader recoder blocks.

S.29 (d)

(i) **Field**: A field is the basic element of database.

(ii) **Record**: A record is a collection of related field that can be treated as a unit by some application program.

(iii) **Database**: A database is a collection of related data.

S.31 (d)

(i), (ii), (iii), (iv) are all valid device directory attributes.

Location : A pointer to the device and location on that device of the file.

Current position : A pointer to the current read or write position in the file.

Process Identification : Time, data and process identification may be kept for creation, last modification and last use. These can be useful for protection and usage monitoring.

Usage Count : Usage count is the no. of times a user logged in the directory.

S.32 (d)

The logical file system manages metadata information. Metadata includes all of the file system structure, excluding the actual data (or contents of the files).

The logical file system manages the directory structure to provide the file organization module with the information the latter needs, given a symbolic file name. It maintains file structure via file control blocks. A file control block (FCB) contains information about the file, including ownership, permissions, and location of the file contents. **The logical file system is also responsible for protection and security.**

S.33 (c)

The OS uses two levels of internal tables : a per process table and a system wide table.

The per process table tracks all files that a process has opened and is stored as information regarding the use of the file by the process. Access rights to the file and accounting information can also be included. Each entry in the per-process table in turn points to the system wide open file table. **The system wide table contains process independent information** such as the location of the file disk, access and file size.

S.34 (c)

Contiguous allocation suffer from the problem of external fragmentation. As files are allocated and deleted, the free disk space is

broken into little pieces. External fragmentation exists whenever free space is broken into chunks. It becomes a problem when the largest contiguous chunk is insufficient for a request; storage is fragmented into a number of holes, no one of which is large enough to store the data. Depending on the total amount of disk storage and the average file size, external fragmentation may be minor or major problem.

In linked Allocation there is no external fragmentation, and any free block on the free-space list can be used to satisfy a request.

Indexed allocation supports direct access, without suffering from external fragmentation, because any free block, on the disk may satisfy a request for more space.

S.35 (c)

The largest possible file will have $16 = 2^4$ indirect blocks. Since each indirect block can point to $256 = 2^8$ blocks, the largest file has $4096 = 2^{8+4}$ blocks. Given a block size of $1024 = 2^{10}$, **the maximum file size is $2^{4+8+10} = 2^{22}$.**

S.36 (a)

The 20k partition will be occupied by the 14k job which will require 10 units of time for execution. After its completion the 10k job will be occupying this 20k partition and on its completion 20k will be taking up the partition.

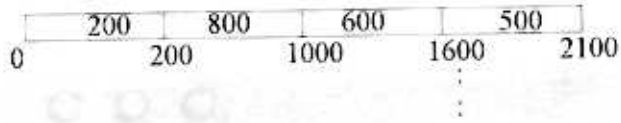
Hence total time = 10 + 1 + 8 = 19 units.

Other jobs will be occupying the other partitions.

S.37 (a)

The maximum value of n for which the system is guaranteed to be deadlock free is 2. Two processes can never lead to deadlock as the peak time demand of 6 (3 + 3) tape drives can be satisfied. But 3 processes can lead to deadlock if each hold 2 drives and then demand one more.

S.40 (b)



∴ From above figure

The relocation constants are 0, 200, 1000, 1600.

S.41 (d)

Keep the records of A fixed and insert the records of the other files in A in sorted order. This would require 20 + 15 + 5 + 25 = **65 record movements**.

S.42 (c)

Threads refer to the execution of different processes simultaneously in the **CPU. Virtual address space** is implemented in **memory. The file-system** is the way in which data in a **disk** is organized. A **signal** is acted upon using the mechanism of **interrupts.**

S.43 (a)

A larger block size guarantees that more data from a single file can be written or read at a time into a single block without having to move the disk's head to another spot on the disk.

The less time spend moving your heads across the disk, the more continuous reads/writes per second. The smaller the block size, the most frequent it is required to move before a read/write can occur.

On the other hand, **the bigger the block size, the more chances to have more wasted space per file** on the last block.

Normally, small block size is ok when we have hundreds of small files-the worst case scenario here is when the last block used by each file is almost empty (10–20%), we end up losing some storage space.

When the files are big, a large block size is better, but again the last block is not always fully issued.

S.44 (c)

The number of disk block pointer that will fit in one block.

$$= \frac{2^{10}\,\text{byte}}{32\,\text{bit}}$$

$$= \frac{2^{10}\,\text{byte}}{4\,\text{byte}} = 256$$

Maximum file size due to single indirection pointer

$= 256 \times 1$

$= 256$ KB

Maximum file size due to direct pointer

$= 10 \times 1$

$= 10$ KB

Maximum file size due to double indirection pointer

$= 256 \times 256 \times 1$ KB

Maximum file size due to triple indirection pointer

$= 256 \times 256 \times 256 \times 1$KB

So the maximum file size

$= 256 \times 256 \times 256 \times 1$ KB

$= 2^8 \times 2^8 \times 2^8 \times 2^{10}$ byte

$= 2^{24} \times 2^{10}$ byte

$= \mathbf{2^{34}}$ **byte.**

S.45 (b)

Command, **mv x, y**

cp x, y(rm x)

will rename a file x to y in Unix system

S.46 (b)

The command ln - s file 1 file 2 works as

It links file 1 with file 2 in argument

It works just as windows shortcut

So, with first statement, we have

file 2 → file 1

So, contents of file 2 is removed and now it is pointing to file 2, By second command li – s file 2 file 3 we got

file 3 → file 2 → file 1

So,**finally only we have information of file 1, i.e., information of II & III would be lost.**

S.47 (a)

Group 1		Group 2	
P.	Gang Scheduling	3.	Thread Scheduling
Q.	Rate Monotonic Scheduling	2.	Real time Scheduling
R.	Fair Share Scheduling	1.	Guaranteed Scheduling

S.48 (d)

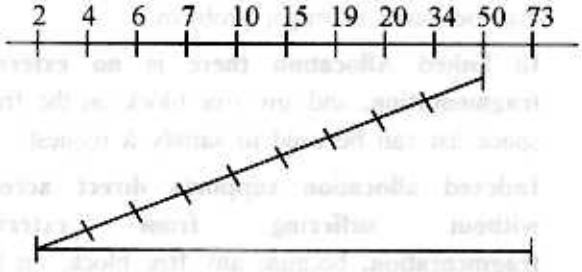
Connect system call is responsible to synchronize the packets.

S.49 (d)

The data blocks of a very large file in the unix file system are **allocated using an extension of indexed allocation** or EXT2 file system.

S.50 (b)

Head if currently at cylinder 50,



50 – 34 = 16

34 – 20 = 14

20 – 19 = 1

19 – 15 = 4

15 – 10 = 5

10 – 7 = 3

7 – 6 = 1

6 – 4 = 2

4 – 2 = 2

73 – 2 = 71

Total moves = 16 + 14 + 1 + 4 + 5 + 3 + 1 + 2 + 2 + 71 = 119

S.51 (a)

In Belady's Anomaly if no. of frame are increased then no. of page fault increases. This behaviour is found **only with FIFO.**

S.52 (b)

Page frame number are most important whereas virtual page number may not be stored entirely.

○ ○ ○

7.3 MEMORY MANAGEMENT

LEVEL–1

Q.1 On a simple paging system with 2^{24} bytes of physical memory, 256 pages of logical address space, and a page size of 2^{10} bytes, No. of bits needed to store an entry in the page table (how wide is the page table)?

(a) 4

(b) 7

(c) 9

(d) 14

Q.2 In case of memory management with bit maps, which is true:

(a) the size of allocation unit and size of bit map is independent

(b) the size of allocation unit and size of bit map is directly proportional

(c) the size of allocation unit and size of the bit map is inversely proportional

(d) none of these

Q.3 In case of swapping system, fifty percent rule is used to

(a) reduce the unused memory

(b) reduce the number of processes

(c) reduce the number of holes

(d) reduce the degree of multiprogramming.

Q.4 Which statement is incorrect in case of multiprogramming?

(a) A reason for multiprogramming a computer is that most process spend a substantial fraction of their time waiting for disk I/O to complete.

(b) Multiprogramming and CPU utilization are totally independent.

(c) The CPU utilization as a function of number of processes is also called as degree of multiprogramming.

(d) Multiprogrammed systems must provide device scheduling, deadlock handling concurrent control.

Q.5 For the given reference string, find the number of page hit if optimal page replacement technique used and the program has three page frames available to it.

7 0 1 2 0 3 0 4 2 3 0

(a) 4

(b) 5

(c) 6

(d) 7

Q.6 Which is invisible to the programmer, and usually which is visible to the programmer.

(a) segmentation, paging

(b) paging, segmentation

(c) partitioning, paging

(d) paging, partitioning

Q.7 Which out of following are true?

(i) For efficient CPU utilization, execution time of each process should be long relative to the swap time.

(ii) Total transfer time is inversely proportional to the amount of memory swapped.

(a) Only (i)

(b) Only (ii)

(c) Neither (i) not (ii)

(d) Both (i) and (ii)

Q.8 For virtual memory to be practical and effective we need the following:

(i) there must be hardware support for the paging and / or segmentation scheme to be employed.

(ii) the operating system must include software for managing the movement of pages and / or segments between secondary memory and main memory.

(a) only (i)

(b) only (ii)

(c) both (i) and (ii)

(d) neither (i) not (ii)

Q.9 Overlay is

(a) a part of an operating system

(b) a peeific memory location

(c) a single contiguous memory that was used in the olden days for running large programs by swapping.

(d) overloading the system with many user files.

Q.10 Fence register is used for

(a) CPU protection

(b) memory protection

(c) file protection

(d) all of the above

Q.11 The hardware which is used for mapping virtual address to physical addresses without going through the page table is called:

(a) A translation look through buffer

(b) A translation lookaside buffer

(c) A translation control buffer

(d) A translation page buffer

LEVEL–2

Q.12 A computer system has 4k word cache organized in a block-set-associative manner, with 4 blocks per set, 64 words per block. The number of bits in the SET and WORD fields of the main memory address format is

(a) 15, 4

(b) 6, 4

(c) 7, 2

(d) 4, 6

Common Data For Questions 13 & 14:

Consider a logical address space of 8 pages of 1024 words mapped into memory of 32 frames.

Q.13 How many bits are there in the logical address?

(a) 9 bits

(b) 11 bits

(c) 13 bits

(d) 15 bits

Q.14 How many bits are there in the physical address?

(a) 9 bits

(b) 11 bits

(c) 13 bits

(d) 15 bits

Q.15 On a system using fixed partitions with sizes 2^{16}, 2^{24} and 2^{32}, how many bits must the limit register have?

(a) 16

(b) 24

(c) 32

(d) None

Common Data For Questions 16 to 19:

On a system using paging and segmentation, the virtual address space consists of up to 8 segments where each segment can be up to 2^{29} bytes long. The hardware pages each segment into 256 byte pages. Bits in the virtual address are

Q.16 Segment number

(a) 1
(b) 2
(c) 3
(d) None

Q.17 Page number

(a) 14
(b) 17
(c) 21
(d) None

Q.18 Offset within page

(a) 2
(b) 4
(c) 8
(d) None

Q.19 Entire virtual address

(a) 16
(b) 24
(c) 32
(d) None

Q.20 Consider the following organization and select the correct option?

Device drivers in ROM
User program
Operating system in RAM

(a) We cannot organize memory in this way.
(b) We can organize memory in this way, but only one process at a time can be running in it.
(c) We can organize memory in this way, and more that one process at a time can be running in it.
(d) We can organize memory in this way but running of process depends upon BIOS which is placed at RAM

Q.21 For the given reference string, find the number of page faults, if first in first out (FIFO) page replacement technique is used. The program has three page frames available to it.

0 2 1 3 5 4 6 3 4 7 7 3 3 5 5 3 1 1 1 7 2 3 4 1

(a) 12
(b) 15
(c) 18
(d) 20

Q.22 Match the following:

Group I		Group II	
(1)	Fixed partitioning	(a)	No external fragmentation
(2)	Dynamic partitioning	(b)	No internal fragmentation
(3)	Simple paging	(c)	Inefficient use of memory due to internal fragmentation
(4)	Simple segmentation	(d)	Inefficient use of processor due to the need for compactions to counter external fragmentation

(a) 1-c, 2-d, 3-a, 4-b
(b) 1-a, 2-b, 3-c, 4-d
(c) 1-b, 2-a, 3-b, 4-c
(d) 1-d, 2-c, 3-b, 4-a

LEVEL–3

Q.23 Disk requests come to a disk driver for cylinders 10, 22, 20, 2, 40, 6 and 38, in that order at a time when the disk drive is reading from cylinder 20. The seek time is 6 ms per cylinder.

The total seek time, if the disk arm scheduling algorithm is first-come-first-served is

(a) 360 ms
(b) 876 ms
(c) 850 ms
(d) 1022 ms

Common Data For Questions 24 & 25:

Consider following page trace :

4, 3, 2, 1, 4, 3, 5, 4, 3, 2, 1, 5

Percentage of page faults that would occur if FIFO page replacement algorithm is used with

Q.24 Number of frames for the Job M = 3, will be

(a) 8

(b) 9

(c) 10

(d) 12

Q.25 Number of frames for the Job M = 4, will be

(a) 8

(b) 9

(c) 10

(d) 12

Q.26 Consider a memory system in which f is fraction of memory occupied by holes and s be the average size of the n processes and ks be the average holes size for some k > 0. The total memory m is

(a) $ns\left(\frac{1+k}{2}\right)$

(b) $ns\left(1+\frac{k}{2}\right)$

(c) $\frac{n}{s}\left(\frac{1+k}{2}\right)$

(d) $\frac{n}{s}\left(1+\frac{k}{2}\right)$

Q.27 Suppose that process spends a fraction p of its time in I/O wait state. With n process in memory at once, the probability that all n processes are waiting for I/O is

(a) 1/p

(b) p^n

(c) $1 - p^n$

(d) $1/p^n$

Q.28 For the given reference string, find the number of page faults, if least recently used page replacement technique is used and the program has three page frames available to it.

7 5 6 8 5 9 5 0 8 9 5 9 8 6 8 5 6 7 5 6

(a) 12

(b) 8

(c) 15

(d) 9

Q.29 Assuming, we have 256K memory available and a resident monitor of 40K. Also consider the following job queue

Job	Memory	Time
1	60K	10
2	100K	5
3	30K	20
4	70K	8

FCFS job scheduling, would cause immediate external fragmentation of

(a) 66K

(b) 46K

(c) 26K

(d) 80K

Q.30 Given memory partitions of 100k, 500k, 200k, and 600k in order, then processes of 212k, 417k, 112k and 426k come in order into memory, then by using first fit algorithm, which of the following statements is correct?

(a) Process P_3 and P_4 not find enough space, so they will have to wait

(b) Any of the processes does not have to wait

(c) Process P_4 does not find enough space, so it will have to wait

(d) None of these

GATE QUESTIONS

Q.31 A memory page containing a heavily used variable that was initialized very early and is in constant use is removed when [GATE 1994]

[1-Mark]

(a) LRU page replacement algorithm is used

(b) FIFO page replacement algorithm is used

(c) LFU page replacement algorithm is used

(d) None of the above

Q.32 In a virtual memory system the address space specified by the address lines of the CPU must be ... than the physical memory size and ..... than the secondary storage size. [GATE 1995]

[2-Marks]

(a) smaller, smaller

(b) smaller, larger

(c) larger, smaller

(d) larger, larger

Q.33 Which of the following page replacement algorithms suffers from Belady's anomaly? [GATE 1995]

[1-Mark]

(a) Optimal replacement

(b) LRU

(c) FIFO

(d) Both (a) and (c)

Q.34 The address sequence generated by tracing a particular program executing in a pure demand paging system with 100 records per page, with a free main memory frame is recorded as follows. What is the number of page faults?

0100, 0200, 0430, 0499, 0510, 0530, 0560, ,0120, 0220, 0240, 0260, 0320, 0370. [GATE 1995]

[2-Marks]

(a) 13

(b) 8

(c) 7

(d) 10

Q.35 In a paged segmented scheme of memory management, the segment table itself must have a page table because [GATE 1995]

[1-Mark]

(a) the segment table is often too large to fit in one page

(b) each segment is spread over a number of pages

(c) segment tables point to page table and not to the physical locations of the segment

(d) the processor's description base register points to a page table

Q.36 Dirty bit for a page in a page table [GATE 1997]

[1-Mark]

(a) helps avoid unnecessary writes on a paging device

(b) helps maintain LRU information

(c) allows only read on a page

(d) none of the above

Q.37 Thrashing [GATE 1997]

[1-Mark]

(a) reduces page I/O

(b) decreases the degree of multiprogramming

(c) implies excessive page I/O

(d) improves the system performance

Q.38 Locality of reference implies that the page reference being made by a process [GATE 1997]

[1-Mark]

(a) will always be to the page used in the previous page reference

(b) is likely to be to one of the pages used in the last few page references

(c) will always be to one of the pages existing in memory

(d) will always lead to a page fault.

Q.39 Which of the following is an example of a spooled device? **[GATE 1998]**

[1-Mark]

(a) The terminal used to enter the input data for the C program being executed

(b) An output device used to print the output of a number of jobs

(c) The secondary memory device in a virtual storage system

(d) The swapping area on a disk used by the swapper

Q.40 If an instruction takes i microseconds and a page fault takes an additional j microseconds, the effective instruction time if on the average a page fault occurs every k instruction is :

[GATE 1998]

[2-Marks]

(a) $i+\frac{j}{k}$

(b) $i + j * k$

(c) $\frac{i+j}{k}$

(d) $(i + j) * j$

Q.41 Which of the following actions is /are typically not performed by the operating system when switching context from process A to process B?

[GATE 1999]

[2-Marks]

(a) Saving current register values and restoring saved register values for process B.

(b) Changing address translation tables.

(c) Swapping out the memory image of process A to the disk

(d) Invalidating the translation look-aside buffer.

Q.42 Which of the following is/are advantages of virtual memory ? **[GATE 1999]**

[2-Marks]

(a) Faster access to memory on an average

(b) Processes can be given protected address spaces

(c) Linker can assign addresses independent of where the program will be loaded in physical memory

(d) Programs larger than the physical memory size can be run.

Q.43 Let m[0] .... m[4] be mutexes (binary semaphores) and p[0] .... p[4] be processes. Suppose each process P[i] executes the following :

wait (m[i]; wait (m [(i + 1) mod 4]),

.....

release (m[i]); release (m[(i + 1)mod 4]);

This could cause **[GATE 2000]**

[1-Mark]

(a) Thrashing

(b) Deadlock

(c) Starvation, but not deadlock

(d) None of the above

Q.44 Suppose the time to service a page fault is on the average 10 milliseconds, while a memory access takes 1 microsecond. Then a 99.99% hit ratio result in average memory access time of

[GATE 2000]

[2-Marks]

(a) 1.9999 milliseconds

(b) 1 millisecond

(c) 9.999 microseconds

(d) 1.9999 microseconds

Q.45 Which of the following need not necesarily be saved on a context switch between processes?

[GATE 2000]

(a) General purpose registers

(b) Translation look aside buffer

(c) Program counter

(d) All of the above

Q.46 Consider a machine with 64 MB physical memory and a 32-bit virtual address space. If the page size is 4KB, what is the approximate size of the page table? **[GATE 2001]**

[2-Marks]

(a) 16 MB

(b) 8 MB

(c) 2 MB

(d) 24 MB

Q.47 Consider a virtual memory system with FIFO page replacement policy. For an arbitrary page access pattern, increasing the number of page frames in main memory will

[GATE 2001]

[1-Mark]

(a) always decrease the number of page faults

(b) always increase the number of page faults

(c) sometimes increase the number of page faults

(d) never affect the number of page faults

Q.48 Where does the swap space reside?

[GATE 2001]

[1-Mark]

(a) RAM

(b) Disk

(c) ROM

(d) On-chip cache

Q.49 Which of the following statements is false?

[GATE 2001]

[1-Mark]

(a) Virtual memory implements the translation of a program's address space into physical memory address space

(b) Virtual memory allows each program to exceed the size of the primary memory

(c) Virtual memory increases the degree of multiprogramming

(d) Virtual memory reduces the context switching overhead

Q.50 More than one word are put in one cache block to **[GATE 2001]**

[1-Mark]

(a) exploit the temporal locality of reference in a program

(b) exploit the spatial locality of reference in a program

(c) reduce the miss penalty

(d) none of the above

Q.51 Which combination of the following features will suffice to characterize an OS as a multi programmed OS?

(i) More than one program may be loaded into main memory at the same time for execution.

(ii) If a program waits for certain events such as I/O, another program is immediately scheduled for execution.

(iii) If the execution of a program terminates, another program is immediately scheduled for execution. **[GATE 2002]**

[2-Marks]

(a) (i)

(b) (i) and (ii)

(c) (i) and (iii)

(d) (i), (ii) and (iii)

Q.52 The optimal page replacement algorithm will select the page that **[GATE 2002]**

[1-Mark]

(a) Has not been used for the longest time in the past

(b) Will not be used for the longest time in the future

(c) Has been used least number of times

(d) Has been used most number of times

Q.53 In a system with 32-bit virtual addresses and 1 KB page size, use of one-level page tables for virtual to physical address translation is not practical because of **[GATE 2003]**

[1-Mark]

(a) the large amount of internal fragmentation

(b) the large amount of external fragmentation

(c) the large memory overhead in maintaining page tables

(d) the large computation overhead in the translation process

Common Data For Questions 54 & 55:

A processor uses 2-level page tables for virtual to physical translation. Page tables for both levels are stored in the main memory. Virtual and physical addresses are both 32-bits wide. The memory is byte addressable. For virtual to physical address translation, the 10 most significant bits of the virtual address are used as index into the first level page table while the next 10-bits are used as index into the second level page table. The 12 least significant bits of the virtual address are used as offset within the page. Assume that the page table entries in both levels of page tables are 4 bytes wide. Further, the processor has a translation look-aside buffer (TLB), with a hit rate of 96%. The TLB caches recently used virtual page numbers are the corresponding physical page numbers. The processor also has a physically addressed cache with a hit rate of 90%. Main memory access time is 10 ns, cache access time is 1 ns, and TLB access time is also 1 ns.

Q.54 Assuming that no page faults occur, the average time taken to access a virtual address is approximately (to the nearest 0.5 as) **[GATE 2003]**

[2-Marks]

(a) 1.5 ns

(b) 2 ns

(b) 3 ns

(d) 4 ns

Q.55 Suppose a process has only the following pages in its virtual address space: two contiguous code pages starting at virtual address 0×00000000, two contiguous data pages starting at virtual address 0×00400000, and a stack page starting at virtual address 0×FFFFF000. The amount of memory required for storing the page tables of this process is **[GATE 2003]**

[2-Marks]

(a) 8 KB

(b) 12 KB

(c) 16 KB

(d) 20 KB

Q.56 Consider a system with a two level paging scheme in which regular memory access takes 150 nanoseconds, and servicing a page fault takes 8 milliseconds. An average instruction takes 100 nanoseconds of CPU time, and two memory accesses. The TLB hit ratio is 90%, and the page fault rate is one in every 10,000 instructions. What is the effective average instruction execution time? **[GATE 2004]**

[2-Marks]

(a) 645 nanoseconds

(b) 1050 nanoseconds

(c) 1215 nanoseconds

(d) 1230 nanoseconds

Q.57 The minimum number of page frames that must be allocated to a running process in a virtual memory environment is determined by **[GATE 2004]**

[1-Mark]

(a) the instruction set architecture

(b) page size

(c) physical memory size

(d) number of processes in memory

Q.58 Consider the following set of processes, with the arrival times and the CPU burst times given in milliseconds.

Process	Arrival Time	Burst Time
P1	0	5
P2	1	3
P3	2	3
P4	4	1

What is the average turnaround time for these processes with the preemptive shortest remaining processing time first (SRPT) algorithm? **[GATE 2004]**

[2-Marks]

(a) 5.50

(b) 5.75

(c) 6.00

(d) 6.25

Q.59 The semaphore variables full, empty and mutex are initialized to 0, n and 1, respectively. Process P_1 repeatedly adds one item at a time to a buffer of size n and process P_2 repeatedly removes one item at a time from the same buffer suiting the programs given below. In the program, K, L, M and N are unspecified statements.

```
P1: while (1) {
        K; P(mutex);
        Add an item to the buffer;
        V(mutex); L;
    }
P2: while (1) {
        M; P(mutex);
        Remove an item from the buffer;
        V(mutex); N;
    }
```

The statements K, L, M and N are respectively

[IT-GATE 2004]

[2-Marks]

(a) P(full), V(empty), P(full), V(empty)

(b) P(full), V(empty), P(empty), V(full)

(c) P(empty), V(full), P(empty), V(full)

(d) P(empty), V(full), P(full), V(empty)

Q.60 We are given 9 tasks T_1, T_2...T_9. The execution of each task requires one unit of time. We can execute one task at a time. T_i has a profit P_i and a deadline d_i profit P_i is earned if the task is completed before the end of the d_i^{th} unit of time.

Task	T_1	T_2	T_3	T_4	T_5	T_6	T_7	T_8	T_9
Profit	15	20	30	18	18	10	23	16	25
Deadline	7	2	5	3	4	5	2	7	3

(i) Are all tasks completed in the schedule that given maximum profit? **[GATE 2005]**

[2-Marks]

(a) All takes are completed

(b) T_1 and T_6 are left out

(c) T_1 and T_8 are left out

(d) T_4 and T_6 are left out

(ii) What is the maximum profit earned? **[GATE 2005]**

[2-Marks]

(a) 147

(b) 165

(c) 167

(d) 175

Q.61 A computer system supports 32-bit virtual addresses as well as 32-bit physical addresses. Since the virtual address space is of the same size as the physical address space, the operating system designers decide to get rid of the virtual memory entirely. Which one of following is true? **[GATE 2006]**

[2-Marks]

(a) Efficient implementation of multi user supports is no longer possible.

(b) The processor cache organization can be made more efficient now.

(c) Hardware support for memory management is no longer needed.

(d) CPU scheduling can be made more efficient now.

Q.62 A CPU generates 32 bit virtual addresses. The page size is 4 KB. The processor has a translation look aside buffer (TLB) which can hold a total of 128 page table entries and is 4 way set associative. The minimum size of the TLB tag is. **[GATE 2006]**

[2-Marks]

(a) 11 bits

(b) 13 bits

(c) 15 bits

(d) 20 bits

Q.63 A virtual memory system uses First In First Out (FIFO) page replacement policy and allocates a fixed number of frames to a process. Consider the following statements.

P: Increasing the number of page frames allocated to a process sometimes increases the page fault rate.

Q: Some program do not exhibit locality of reference.

Which one of the following is TRUE? **[GATE 2007]**

[2-Marks]

(a) Both P and Q are true, and Q is the reason for P

(b) Both P and Q are true, but Q is not the reason for P

(c) P is false, but Q is true

(d) Both P and Q are false

Common Data For Questions 64 & 65:

A process has been allocated 3 page frames. Assume that none of the pages of the process are available in the memory initially. The process makes the following sequence of page reference (reference string) 1,2,1,3,7,4,5,6,3,1.

Q.64 If optimal page replacement policy is used, how many page faults occur for the above reference string? **[GATE 2007]**

[2-Marks]

(a) 7

(b) 8

(c) 9

(d) 10

Q.65 Least Recently Used (LRU) page replacement policy is a practical approximation to optimal page replacement. For the above reference string, how many more page faults occur with LRU than with the optimal page replacement policy? **[GATE 2007]**

[2-Marks]

(a) 0

(b) 1

(c) 2

(d) 3

Q.66 A process executes the following code

```
for (i=0; i<n;i++) fork ();
```

The total number of child processes created is **[GATE 2008]**

[2-Marks]

(a) n

(b) 2^n-1

(c) 2^n

(d) $2^{n+1}-1$

Q.67 A processor uses 36 bit physical addresses and 32 bit virtual addresses, with a page frame size of 4 Kbytes. Each page table entry is of size 4 bytes. A three level page table is used for virtual to physical address translation, where the virtual address is used as follows.

- Bits 30–31 are used to index into the first level page table
- Bits 21–29 are used to index into the second level page table
- Bits 12–20 are used to index into the third level page table
- Bits 0–11 are used as offset within the page

The number of bits required for addressing the next level page table (or page frame) in the page table entry of the first, second and third level page tables are respectively. **[GATE 2008]**

[2-Marks]

(a) 20, 20 and 20

(b) 24, 24 and 24

(c) 24, 24 and 20

(d) 25, 25 and 24

Q.68 A multilevel page table is preferred in comparison to a single level page table for translating virtual address to physical address because **[GATE 2009]**

[2-Marks]

(a) it reduces the memory access time to read or write a memory location

(b) it helps to reduce the size of page table needed to implement the virtual address space of a process.

(c) it is required by the translation lookaside buffer

(d) it helps to reduce the number of page faults in page replacement algorithms.

ANSWER KEY

1	d	2	c	3	c	4	b	5	a
6	b	7	a	8	c	9.	c	10	b
11	b	12	d	13	c	14	d	15	c
16	c	17	c	18	c	19	c	20	b
21	b	22	a	23	d	24	b	25	c
26	b	27	b	28	a	29	c	30	c
31	b	32	c	33	c	34	c	35	a
36	a	37	c	38	b	39	b	40	a
41	c, d	42	b, d	43	b	44	a	45	d
46	c	47	c	48	b	49	d	50	b
51	d	52	b	53	c	54	d	55	c
56	d	57	a	58	a	59	d	60(i, ii)	d, a
61	c	62	a	63	b	64	a	65	c
66	b	67	b	68	b				

SOLUTIONS

S.1 (d)

Each entry contains 1 bit to indicate if the page is valid and **14 bits** to specify the page frame number.

S.2 (c)

In the memory management with Bit maps, the size of the allocation unit is an important design issue. The smaller the allocation unit, the larger the bit map i.e. **they are inversely proportional.**

S.3 (c)

The fifty percent rule has its origin in a fundamental asymmetry between process and holes. **When two holes are adjacent in memory, they are merged into a single hole.** Adjacent process are not merged.

S.4 (b)

CPU utilization can be improved when multiprogramming is used. Therefore, both are dependent.

S.5 (a)

Optimal page replacement technique replaces that page which will not be used for the longest period of time.

Thus for the given reference string, optimal page replacement would give:

7	0	1	2	0	3
7	7	7	2	2	2
	0	0	0	0	0
		1	1	1	3
F	F	F	F	H	F

0	4	2	3	0
2	2	2	2	2
0	4	4	4	0
3	3	3	3	3
H	F	H	H	F

Total number of pages = 11

∴ **Number of hits = 4**

S.6 (b)

Paging is **invisible** to the programmer, **segmentation** is usually **visible** and is provided as a convenience for organizing programs and data.

S.7 (a)

For efficient CPU utilization, execution time for each process should be long relative to the swap time.

Major part of the swap time is transfer time. **The total transfer time is directly proportional to the amount of memory swapped.**

S.12 (d)

There are 64 words in a block.

So 4 K cache has (4 × 1024)/64 = 64 blocks. Since 1 set has 4 blocks, there are 16 sets. 16 **sets needs 4 bits for representation**. If a set has 64 words. **Then, the word field has 6 bits.**

S.13 (c)

Logical address will have 3 bits to specify the page number (for 8 pages).

10 bits to specify the offset into each page (2^{10} = 1024 words)

= **13 bits.**

S.14 (d)

For (2^5), 32 frames of 1024 words each (page size = frame size)

we have **5 + 10 = 15 bits.**

S.15 (c)

Since the system has only one limit register, it must be large enough to accommodate addresses in any partition. The largest possible partition is 2^{32}. **Therefore, the limit register must have 32 bits.**

S.16 (c)

Since $8 = 2^3$, 3 bits are needed to specify the **segment number.**

S.17 (c)

With $256 = 2^8$ byte pages, a 2^{29} byte segment can have $\frac{2^{29}}{2^8} = 2^{21}$ pages. Thus, **21 bits** are needed to specify the **page number.**

durations. Spatial locality refers to the use of data elements within relatively close storage locations.

To exploit the spatial locality, more than one word are put into cache block.

S.52 (b)

Optimal page replacement algorithm will select the page that **will not be used for the longest time in the future.** Optimal page replacement algorithm is best page replacement algorithm but it is impossible to implement.

S.53 (c)

Given, page size = 1 KB = 2^{10} bit

Virtual address is 32 bit long

So required number of pages = $2^{32}/2^{10} = 2^{22}$

2^{22} is a very large so the large memory overhead in maintaining page tables.

S.54 (d)

Average time taken to access a virtual address

$=[(.96)(1+0.9+0.1\times11)] + [(0.04(21+0.9+0.11)]$

$= 3.87$ ns

$=$ **4 ns**

S.55 (c)

Total amount of memory required for starting the page tables of this process.

$= 4 \times 2^{12}$ byte

$= 4 \times 4 \times 2^{10}$ byte

$=$ **16 KB**

S.56 (d)

Let the probability of page fault $F = 1/2 \times 10^2$

Total memory access time $= 0.90 \times 150 + 0.10 \times 300 = 135 + 30 = 165$ns

So **instruction execution time**

$= 100 + 2\,[165 + 1/2 \times 10^4 \times 8 \times 10^6]$ns

$= 100 + 2 \times 565$ ns

$= 100 + 1130$ ns $=$ **1230 ns**

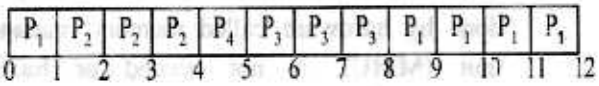
S.57 (a)

Such process needs minimum number of pages based on **instruction set architecture.**

S.58 (a)

Process	Arrival Time	Burst Time
P_1	0	5
P_2	1	3
P_3	2	3
P_4	4	1

The Gannt chart for SRPT CPU scheduling algorithm is



Process P_1 arrive at time 0, at time unit 1 the remaining time of P_1 is 4 and remaining time of P_2 is 3 so P_1 preempted, at the end of time unit 2 the remaining time of P_2 is 2 and P_3 is 3 so P_2 is not preempted At time unit 4 the remaining time of P_2 is 0 and P_4 is scheduled. The same process repeated upto time unit 12.

The turnaround time is the interval between submission and completion of a given process.

Calculate turnaround time for each process as

$P_1 = 12 - 0 = 12$

$P_2 = 4 - 1 = 3$

$P_3 = 6$, $P_4 = 1$

So average turnaround time

$$= \frac{12+3+6+1}{4}$$

$= 22/4 \approx$ **5.50**

S.59 (d)

This is the algorithm of solution of consumer-producer process with the help of semaphore.

So, **K = P(empty)** P for wait

L = V(full) V for signal

M = P(full)

N = V(empty)

S.60(i)(d)

The given problem is job scheduling problem 9 tasks are $T_1, T_2, \ldots, T_9$.

j is intially empty then according to deadlines it includes $\{T_1, T_2, T_3, T_5, T_7, T_8, T_9,\}$ So **T_4 and T_6 can't be include in J.**

S.60(ii) (a)

Total profit earn by algorithm

$= 15 + 20 + 30 + 18 + 23 + 16 + 25$

$= \mathbf{147}$

S.61 (c)

If the computer system supports 32 bit virtual addresses as well as 32 bit physical addresses. Then address translation from virtual to physical done by hardware called memory management unit (MMU), is not needed or **hardware support for memory management is no longer needed.**

S.62 (a)

Size of virtual address = 32 bits

Page size = 4 kB

No. of pages = 128 = 2^7 so 7 = bit are needed for page table entry.

Page size = 4 kB = 4.2^{10} byte = 2^{12} so 12 bits

So 4 kB page size requires 12 bits

4 ways associative memory cell needs 2 bits (2^2 =4)

Tag minimum size in the TLB = 32–21 = **11 bits.**

S.63 (b)

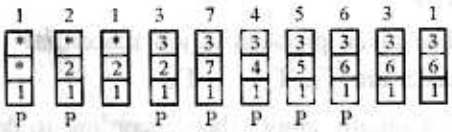
P and Q both are true but Q is not the reason for P because increasing the number of page frames allocated to a process sometimes increases the page fault rate or it is not concerned with replacement policy.

S.64 (a)

Number of page frames = 3

1.2,1.3,7,4,5,6,3,1

10 page frames



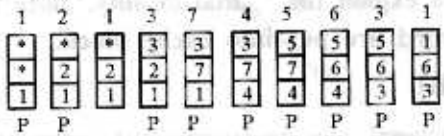
Referring string

so total number of page fault = 7

S.65 (c)

Consider the LRU page replacement policy

10 page Frames



So total number of page fault =9

Difference = no. of page fault in LRU – No. of page fault in optional = 9 – 7 = **2 more page fault**

S.66 (b)

fork() system call creates the child process initially number of processes is 0. After first fork() it creates a single process after second fork() it creates one parent two child process, after n fork () the total number of processes is $\mathbf{2^n - 1}$.

S.67 (b)

Number of bits required for first level page table = **24**

Number of bits required for second level page table = **24**

Number of bits required for third level page table = **24**

S.68 (b)

Multilevel page table is preferred to reduce the size of page table needed to implement the virtual address space of a process.

○ ○ ○

7.4 Input Output and Disk Scheduling

LEVEL–1

Q.1 File I/O initiation and termination is done by

(a) Basic I/O supervisor

(b) Physical I/O supervisor

(c) Logical I/O supervisor

(d) Both (a) & (b)

Q.2 In file organization, where users and application are concerned with seconds. I/O is done on

(a) an unblocked basis

(b) a second basis

(c) a block basis

(d) Both (a) & (c)

Q.3 Memory Access Time is

(a) Minimum time required in initiations of two successive memory operations

(b) Time that escapes between initiation of operation end of operation

(c) Time required to access devices.

(d) None of these

Q.4 The aim of disk scheduling algorithm is to:

(a) Minimize disk head movement

(b) Minimize waiting time

(c) Both (a) and (b)

(d) None of these

Q.5 In a certain disk-scheduling algorithm, the disk arm sweeps back and forth, across the disk surface, servicing all requests in its path. It changes direction only when there are no more requests to service in the current direction. Which is this algorithm?

(a) FCFS

(b) SCAN

(c) SSTF

(d) None of these

Q.6 Which type of disk scheduling has the following characteristics that discriminates sharply against the innermost and outermost tracks resulting in better throughput and mean response time tend to be longer for moderate loads useful in batch processing etc.:

(a) Scan

(b) SRT

(c) SSTF

(d) N-scan

Q.7 For batch process, the access is

(a) Random access

(b) Sequential access

(c) Indexed sequential

(d) Indexed sequential access

Q.8 The file manager

(a) Does not perform the actual reading and writing of data

(b) Performs the actual reading and writing of data

(c) Performs only actual reading

(d) Performs only actual writing

Q.9 The interface between the controller and the device is

(a) High-level interface

(b) Low-level interface

(c) Arbitrary-level interface

(d) Abstract - level interface

Q.10 In order to block a process, the system will do

(a) DOWN on a semaphore

(b) Send a message to the blocked process

(c) UP on a semaphore

(d) SIGNAL on a condition variable in a monitor

Q.11 Function of device-independent software is

(a) Uniform interface to the user-level software

(b) I/O functions common to all devices

(c) Both (a) and (b)

(d) Neither (a) nor (b)

Q.12 Minor device number is

(a) Passed as parameter to the driver

(b) Used to locate the appropriate device

(c) The actual number of devices in the system

(d) None of the above.

Q.13 The functions of Device drivers I/O system layer is :

(a) Perform I/O operation

(b) Spooling

(c) Allocation

(d) Check Status

Q.14 Which out of the following scheduling algorithms, always handle the closest request next?

(a) SSF algorithm

(b) Elevator algorithm

(c) FCFS algorithm

(d) Round robin algorithm

Q.15 Which out of following is not an escape sequence?

(a) Scroll screen up by n lines

(b) Move cursor to (x,y)

(c) Insert character at cursor

(d) None of these

Q.16 Break mode is

(a) Compromise between raw and cooked mode

(b) Cooked mode

(c) Raw mode

(d) Default mode

Q.17 Cooked mode is

(a) Driver delivering corrected lines to the user programs

(b) Raw mode

(c) Character oriented

(d) Driver accepting input and passing it upward unmodified.

Q.18 Which out of the following are not human-interface device?

(a) Screen

(b) Mouse

(c) Keyboard

(d) Disks

Q.19 Which statement is incorrect about I/O Devices?

(a) The character I/O device is not addressable and does not support seek operation.

(b) In case of I/O devices, common block sizes range from 128 bytes to 1024 bytes.

(c) A character device delivers or accepts a stream of characters, without regard to any block structure.

(d) A block I/O device is one that stores information in fixed size blocks, each one with its own address.

Q.20 Major device number is

(a) Passed as a parameter to the driver

(b) Used to locate the appropriate driver

(c) The actual number of devices in the system

(d) None of the above

Q.21 Function of interrupt handler I/O system layer is
(a) Wakeup driver when I/O completed
(b) Blocking
(c) Setup device register
(d) Spooling

Q.22 Controller error is
(a) request for non existent sector
(b) physical damage of disk block
(c) refusal to accept commands
(d) None of these

Q.23 Match the following

Group I		Group II	
(i)	Human Readable	(a)	Digital line drivers
(ii)	Machine readable	(b)	Printers
(iii)	Communication	(c)	Tape drives

(a) (i) -a, (ii) - c, (iii) -b
(b) (i) -b, (ii) -c, (iii) -a
(c) (i) -b, (ii) -a, (iii) -c
(d) (i) -c, (ii) -b, (iii) -a

Q.24 In order to unblock the process, the system will do
(a) WAIT on a condition variable in a monitor
(b) SIGNAL on a condition variable in a monitor
(c) RECEIVE on a message
(d) DOWN on a semaphore

Q.25 Working set (t, k) at an instant of time, t, is the set of
(a) k future references that the operating system will make
(b) future references that the operating system will make in the next 'k' time units
(c) k references with high frequency
(d) pages that have been referenced in the last k time units

Q.26 Supervisor call
(a) is a call made by the supervisor of the system
(b) is a call with control functions
(c) are privileged calls that are used to perform resource management functions, which are controlled by the operating system.
(d) is a call made by someone working in root directory

Q.27 Sector inteleaving in disk is done by
(a) the disk manufacturer
(b) the disk controller cord
(c) the operating system
(d) none of the above

Q.28 The first-fit, best-fit and the worst-fit algorithm can be used for
(a) contiguous allocation of memory
(b) linked allocation of memory
(c) indexed allocation of memory
(d) all of the above

Q.29 In a system that does not support swapping
(a) the compiler normally binds symbolic addresses (variable) to relocatable addresses.
(b) the compiler normally binds symbolic addresses to physical addresses.
(c) the loader binds relocatable addresses to physical addresses.
(d) binding of sysmbolic addrsses to physical addresses noramlly takes place during execution

LEVEL–2

Common Data For Questions 30 to 35:

On a disk with 1000 cylinders, numbers 0 to 999, compute the number of tracks the disk arm move to satisfy all the requests in the disk queue. Assume the last request serviced was at track 345 and the head is moving towards track 0. The queue in FIFO order contains requests for the following tracks : 123, 874, 692, 475, 105, 376. Perform the computation for the following scheduling algorithms:

Q.30 FIFO
(a) 1013
(b) 1713
(c) 2013
(d) None

Q.31 SSTF
(a) 1198
(b) 1298
(c) 1398
(d) None

Q.32 SCAN
(a) 1019
(b) 1119
(c) 1219
(d) None

Q.33 LOOK
(a) 809
(b) 909
(c) 1009
(d) None

Q.34 C-SCAN
(a) 1267
(b) 1567
(c) 1967
(d) None

Q.35 C-LOOK
(a) 1107
(b) 1207
(c) 1507
(d) None

Common Data For Questions 36 to 38:

A disk has 8 sectors per track and spins at 600 rpm. It takes the controller 10 ms from the end of one I/O operation before it can issue a subsequent one. How long does it take to read all 8 sectors using the following interleaving systems?

Q.36 No interleaving
(a) 400 ms
(b) 500 ms
(c) 600 ms
(d) 800 ms

Q.37 Single interleaving
(a) 50 ms
(b) 100 ms
(c) 150 ms
(d) 200 ms

Q.38 Double interleaving
(a) 75 ms
(b) 175 ms
(c) 275 ms
(d) None

Common Data For Questions 39 to 40:

Calculate the No. of bits needed to store the free list under the following conditions, if an array of free blocks were used to implement the free list?

Q.39 16 bits per disk address ; 500,000 blocks total ; 200,000 free blocks
(a) 1,200,000
(b) 2,200,000
(c) 3,200,000
(d) None

Q.40 16 bits per disk address ; 500,000 blocks total ; 0 free blocks
(a) 0
(b) 1
(c) 5
(d) 10

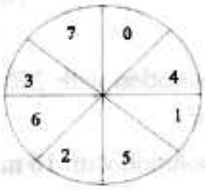
Q.41 Match the following

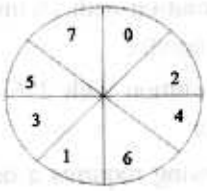
Group I		Group II	
(i)	Programming error	(a)	the arm sent to cylinder 6 but it went to 7
(ii)	Transient checksum error	(b)	disk block physically damaged
(iii)	Permanent checksum error	(c)	request for non existent sector
(iv)	Seek error	(d)	controller refuses to accept commands

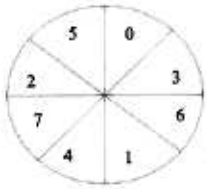
(a) (i) - a, (ii) - d (iii) -c, (iv) -b
(b) (i) - b, (ii) - a, (iii) - c, (iv) - d
(c) (i) -c, (ii) - d, (iii) - b, (iv) -a
(d) (i) -d, (ii) - c, (iii) - b, (iv) -a

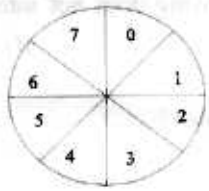
Q.42 In a multi-user operating system, 20 requests are made to use a particular resource per hour, on an average. The probability that no requests are made in 45 minutes is
(a) e^{-15}
(b) e^{-5}
(c) $1-e^{-5}$
(d) $1-e^{-1}$

Q.43 Which out of following is an example of double interleaving?

(a) 

(b) 

(c) 

(d) 

GATE QUESTIONS

Q.44 The sequence ..... is an optimal nonpreemptive scheduling sequence for the following jobs which leaves the CPU idle for ..... unit (s) of time

Job	Arrival Time	Burst Time
1	0.0	9
2	0.6	5
3	1.0	1

[GATE 1995]
[2-Marks]

(a) {3, 2, 1}, 1
(b) {2, 1, 3}, 5
(c) {3, 2, 1}, 0
(d) {1, 2, 3}, 5

Q.45 Which scheduling policy is most suitable for a time-shared operating system? **[GATE 1995]**
[1-Mark]

(a) Shortest Job First
(b) Round Robin
(c) First Come First Serve
(d) Elevator

Q.46 Four jobs to be executed on a single processor system arrive at time 0^+ in the order A, B, C, D. Their burst CPU time requirements are 4, 1, 8, 1 time units respectively. The completion time of A under round robin scheduling with time slice of one time unit is **[GATE 1996]**
[2-Marks]

(a) 10
(b) 4
(c) 8
(d) 9

Q.47 When an interrupt occurs, an operating system **[GATE 1997]**
[1-Mark]

(a) ignores the interrupt
(b) always changes state of interrupted process after processing the interrupt
(c) always resumes execution of interrupted process after processing the interrupt
(d) may change state of interrupted process to 'blocked' and schedule another process.

Q.48 The correct matching for the following pairs is **[GATE 1997]**
[1-Mark]

Group I		Group II	
(A)	Disk scheduling	(1)	Round robin
(B)	Batch processing	(2)	SCAN
(C)	Time sharing	(3)	LIFO
(D)	Interrupt processing	(4)	FIFO

(a) A–3, B–4, C–2, and D–1
(b) A–4, B–3, C–2, and D–1
(c) A–2, B–4, C–1, and D–3
(d) A–3, B–4, C–3, and D–2

Q.49 Consider n processes sharing the CPU in a round-robin fashion. Assuming that each process switch takes s seconds, what must be the quantum size q such that the overhead resulting from process switching is minimized but, at the same time each process is guaranteed to get its turn at the CPU at least every t seconds?

[GATE 1998]
[2-Marks]

(a) $q \le \frac{t-ns}{n-1}$

(b) $q \ge \frac{t-ns}{n-1}$

(c) $q \le \frac{t-ns}{n+1}$

(d) $q \ge \frac{t-ns}{n+1}$

Q.50 A multi-user, multi-processing operating system cannot be implemented on hardware that does not support **[GATE 1999]**
[1-Mark]

(a) Address translation
(b) DMA for disk transfer
(c) At least two modes of CPU execution (privileged and non-privileged)
(d) Demand paging.

Q.51 System calls are usually invoked by using
[GATE 1999]
[1-Mark]

(a) a software interrupt
(b) polling
(c) an indirect jump
(d) a privileged instruction

Q.52 Which of the following disk scheduling strategies is likely to give the best through put?
[GATE 1999]
[1-Mark]

(a) Farthest cylinder next
(b) Nearest cylinder next
(c) First come first served
(d) Elevator algorithm

Q.53 A graphics card has on borad memory of 1 MB. Which of the following modes can the card not support? **[GATE 2000]**
[2-Marks]

(a) 1600 × 400 resolution with 256 colours on a 17 inch monitor
(b) 1600 × 400 resolution with 16 million colours on a 14 inch monitor
(c) 800 × 400 resolution with 16 million colours on a 17 inch monitor
(d) 800 × 800 resolution with 256 colours on a 14 inch monitor

Q.54 Which of the following requires a device driver?
[GATE 2001]
[1-Mark]

(a) Register
(b) Cache
(c) Main memory
(d) Disk

Q.55 Which of the following does not interrupt a running process? **[GATE 2001]**
[2-Marks]

(a) A device
(b) Timer
(c) Scheduler process
(d) Power failure

Q.56 Consider a set of n tasks with known runtimes $r_1, r_2, \ldots\ldots r_n$ to be run on a uniprocessor machine. Which of the following processor scheduling algorithms will result in the maximum throughput? **[GATE 2001]**
[1-Mark]

(a) Round-Robin
(b) Shortest-Job-First
(c) Highest-Response-Ratio-Next
(d) First-Come-First-Served

Q.57 Which of the following scheduling algorithm is non-preemptive? **[GATE 2002]**
[1-Mark]

(a) Round Robin
(b) First-In First-Out
(c) Multilevel Queue Scheduling
(d) Multilevel Queue Scheduling with Feedback

Q.58 Consider an operating system capable of loading and executing a single sequential user process at a time. The disk head scheduling algorithms used is First come First Served (FCFS). If FCFS is replaced by Shortest Seek Time First (SSFT), claimed by the vendor to given 50% better benchmark results, what is the expected improvement in the I/O performance of user programs? **[GATE 2004]**

[1-Mark]

(a) 50%

(b) 40%

(c) 25%

(d) 0%

Q.59 A disk has 200 tracks (numbered 0 through 199). At a given time, it was servicing the request of reading data from track 120, and at the previous request, service was for track 90, the pending requests (in order of their arrival) are for track numbers.

30 70 115 130 110 80 20 25.

How many times will the head change its direction for the disk scheduling policies SSTF (Shortest Seek Times First) and FCFS (First Come First Serve)? **[IT-GATE 2004]**

[2-Marks]

(a) 2 and 3

(b) 3 and 3

(c) 3 and 4

(d) 4 and 4

Q.60 We wish to schedule three processes P1, P2 and P3 on a uniprocessor system. The priorities, CPU time requirements and arrival times of the processes are as shown below.

Process	Priority	CPU time required	Arrival time (hh:mm:ss)
P1	10 (highest)	20 sec	00:00:05
P2	9	10 sec	00:00:03
P3	8 (lowest)	15 sec	00:00:00

We have a choice of preemptive or non-preemptive scheduling. In preemptive scheduling, a late-arriving higher priority process can preempt a currently running process with lower priority. In non-preemptive scheduling, a late arriving higher priority process must wait for the currently executing process to complete before it can be scheduled on the processor.

What are the turnaround times (time from arrival till completion) of P2 using preemptive and non-preemptive scheduling respectively?

[IT-GATE 2005]

[2-Marks]

(a) 30 sec, 30 sec

(b) 30 sec, 10 sec

(c) 42 sec, 42 sec

(d) 30 sec, 42 sec

Q.61 Which of the following statements about synchronous and asynchronous I/O is NOT true? **[GATE 2008]**

[2-Marks]

(a) An ISR is invoked on completion of I/O in synchronous I/O but not in asynchronous I/O

(b) In both synchronous and asynchronous I/O an ISR (Interrupt Service Routine) is invoked after completion of the I/O

(c) A process making a synchronous I/O call wants until I/O is complete, ubt a process making an asynchronous I/O calls does not wait for completion of the I/O

(d) In the case of synchronous I/O, the process waiting for the completion of I/O is woken up by the ISR that is invoked after the completion of I/O.

ANSWER KEY

1	a	2	c	3	a	4	d	5	b
6	c	7	b	8	a	9	b	10	a
11	c	12	a, b	13	d	14	a	15	d
16	a	17	a	18	d	19	a	20	b
21	a	22	c	23	b	24	b	25	d
26	c	27	c	28	a	29	a, c	30	c
31	b	32	c	33	c	34	c	35	c
36	d	37	d	38	c	39	c	40	a
41	d	42	a	43	c	44	a	45	b
46	d	47	d	48	c	49	b	50	a, b, c
51	a	52	b	53	b	54	d	55	c
56	a	57	b	58	d	59	a	60	d
61	b								

SOLUTIONS

S.4 (d)

Goal of Disk scheduling:

- High throughput
- Fairness

S.9 (b)

The interface between the controller and the device is often a very **low-level interface.**

S.10 (a)

The process can block itself by doing, a DOWN on a semaphore, a WAIT on a condition variable or a RECEIVE on a message.

S.11 (c)

The basic function of the device-independent software is to perform the I/O functions that are common to all devices, and to provide a uniform interface to the user-level software.

S.12 (a, b)

Minor device number is **passed as a parameter to the driver** to specify the unit to be read or written. Also minor numbers are used by the operating system to determine the actual device to be accessed by the user-end request for the special device file.

S.13 (d)

The functions of Device drivers I/O system layer is

(i) Setup device registers

(ii) Check status

S.14 (a)

SSF (Shortest Seek First) always handles the closest request next to minimize seek time.

S.15 (d)

All the other options are escape sequence.

S.17 (a)

Line-oriented philosophy says that the driver handles all the interline editing and just **delivers corrected lines to the user programs.** This is also referred as cooked mode.

S.18 (d)

Screen, keyboard and mouse are human interface devices while **disk is a storage device.**

S.19 (a)

When accessing a device file, the major number selects which device is being called to perform input/output operation. This call is done with the minor member as a parameter and it is entirely upto the driver how the minor number is being interpreted.

S.23 (b)

Human readable

Suitable for communicating with the computer user.

Example include printers and video display terminals.

Machine readable

Suitable for communicating with electronic equipment.

Examples are disk and **tape drives**, sensors, controllers and actuators.

Communication

Suitable for communicating with remote devices.

Examples are **digital line drivers and modems**

S.24 (b)

When the interrupt happens, the interrupt procedure does whatever it has to in order to unblock the process that started it. In some systems it will do an UP on a semaphore In other it will do a **SIGNAL on a condition variable in a monitor.**

S.30 (c)

The tracks travelled to will be 345, 123, 874, 692, 475, 105 and 376, making the total distance 222 + 751 + 182 + 217 + 370 + 271 = **2013**.

S.31 (b)

The tracks travelled to will be 345, 376, 475, 692, 874, 123 and 105, making the total distance 529 + 769 = **1298**.

S.32 (c)

The tracks travelled to will be 345, 123, 105, 0, 376, 475, 692 and 874, making the total distance 345 + 874 = **1219**.

S.33 (c)

The tracks travelled to will be 345, 123, 105, 376, 475, 692 and 874, making the total distance 240 + 769 = **1009**.

S.34 (c)

The tracks travelled to will be 345, 123, 105, 0, 999, 874, 692, 475 and 376, making the total distance 345 + 999 + 623 = **1967**.

S.35 (c)

The tracks travelled to will be 345, 123, 105, 874, 692, 475 and 376, making the total distance 240 + 769 + 498 = **1507**.

S.36 (d)

The disk makes 10 revolutions per second or one revolution in 100 ms. In 10 ms, less than one sector will have passed under the read/write head.

The next sector cannot be read until the disk makes almost a complete revolution. It will require 8 revolutions to read all 8 sectors. At 100 ms per revolution, it will take **800 ms**.

S.37 (d)

The next sector will spin under the read/write head almost as soon as the next I/O operation is issued. Two revolutions will be needed to read all 8 sectors, making the total read time **200 ms**.

S.38 (c)

A total of 2.75 revolutions will be needed to read all 8 sectors, for a total time of **275 ms**.

S.39 (c)

200,000 addresses × 16 bits per address = **3,200,000 bits.**

S.40 (a)

0 addresses × 16 bits per address = **0 bits**.

S.42 (a)

The arrival pattern is a Poission distribution.

$P(k \text{ requests}) = e^{-\mu T} (\mu T)^k / k!$

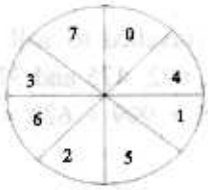
Here $k = 0$, $\mu = 20$, $T = 3/4$.

So required probability is e^{-15}.

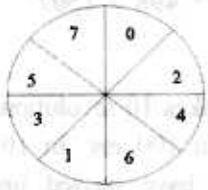
S.43 (c)

Skipping blocks to give the controller time to transfer data to memory is called interleaving.

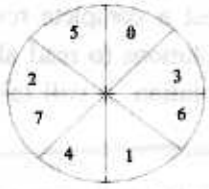
Example of no interleaving:



Example of single interleaving:



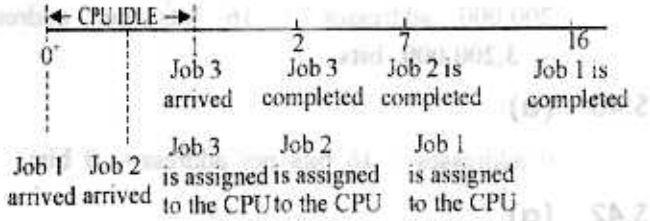
Example of double interleaving:



S.44 (a)

The sequence {3, 2, 1} is an optimal non preemptive scheduling sequence for the following jobs which leaves the CPU idle for 1 unit(s) of time.

The optimal non preemptive scheduling sequence.



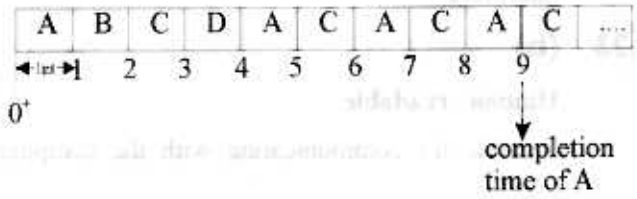
S.45 (b)

Round Robin scheduling algorithm is especially used for time sharing system. It is similar to FCFS but preemption is added to switch between processes, small unit of time called time quantum or time slice is defined.

S.46 (d)

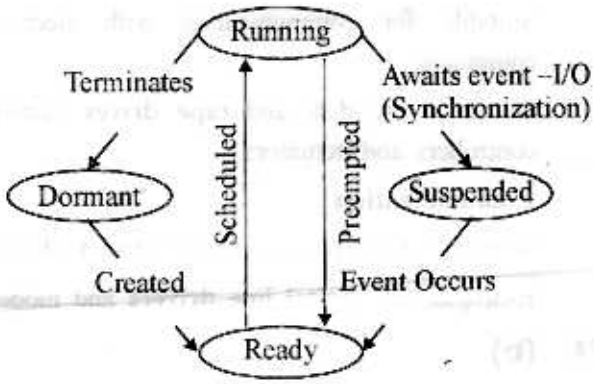
Jobs arrive at time 0^+ in order A, B, C, D.

Their burst CPU time requirements are 4, 1, 8,1 time units respectively. CPU scheduling algorithm is Round Robin with time slice of one time unit.



Completion time of A is 9 unit.

S.47 (d)



We are having confusion between option (b) and option (d). When interrupt occurs process which is currently running changes the state. After processing the interrupt it is not always the case that same interrupted process is invoked by Operating system may be higher priority process is present in the ready state. Operating System may take that process for execution.

S.48 (c)

Disk scheduling – SCAN

Batch processing – FIFO

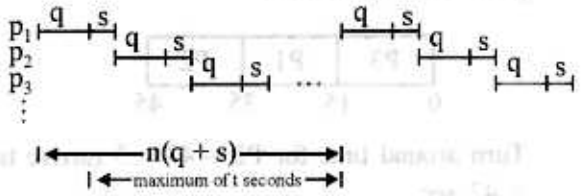
Time sharing – Round Robin

Interrupt processing – LIFO

S.49 (b)

The diagram below shows the use of the CPU by a sequence of processes p_1, p_2, p_3, ... under round-robin scheduling. The segments labeled "q" represent the length of the time quantum, and

the segments labeled "s' represent the time it takes to do the process switch.



The problem statement requires a process to be idle for at most t seconds; this interval is indicated in the diagram. From this we derive the following constraint:

$$n(s + q) - q \leq t,$$

which can be transformed into

$$q \leq \frac{t - ns}{n - 1}$$

To reduce switching overhead, q should be as large as possible, subject to the above restriction. Hence,

$$q = \frac{t - ns}{n - 1}$$

S.50 (a, b, c)

Address translation is required to give different address spaces to different processes and user **DMA is required for any disk transfer**. **Two modes of execution are required to prevent current users from harming one another**. Demand paging is not really required for a multi-user, multi processing operating system, though it would help in implementing the same.

S.51 (a)

System calls are usually invoked by using a software interrupt.

S.52 (b)

Nearest Cylinder next will give the best throughput.

S.53 (b)

Monitor size doesn't matter here. So, we can easily deduct that answer should be (b) as this has the highest memory requirements. Let us verify it.

Number of bits required to store a 16 M colors pixel = ceil (log2(16*1000000)) = 24

Number of bytes required for 1600 × 400 resolution with 16M colors = (1600 × 400 × 24)/ 8 which is 192000000 (greater than 1 MB).

S.55 (c)

Schedular does not interrupt a running process. There are three types of schedular. Long term schedular works with batch queue and selects the next job to be executed. Medium term schedular is used for swapping in and swapping out operation from memory. Short term schedular allocates processor among the pool of ready processes residents in memory.

S.56 (a)

Round Robin scheduling algorithm will always result in maximum throughput.

S.57 (b)

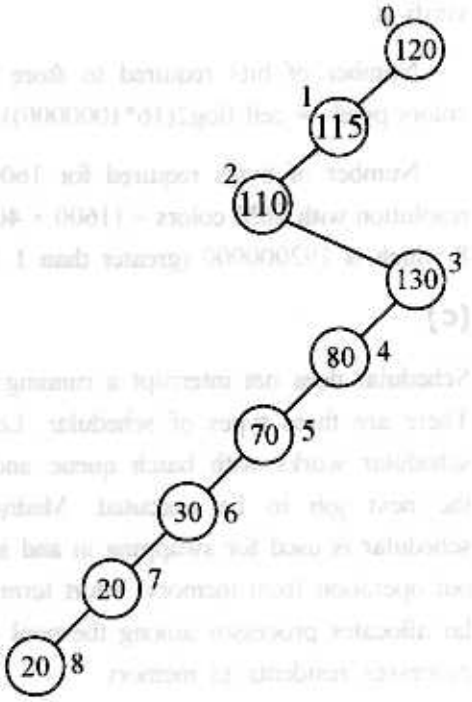
First in first out is non preemptive. That is process which comes first is assigned to the CPU until the process's CPU utilization is completed. After completing the process 1 next process is assigned to the CPU and so on.

S.58 (d)

The I/O performance of a user program determined by many input and output devices not only by the disk. When we uses FCFS and replace it by SSTF it improves only disk driver performance not the entire I/O performance. So I/o improvement performance of user program is **0%**.

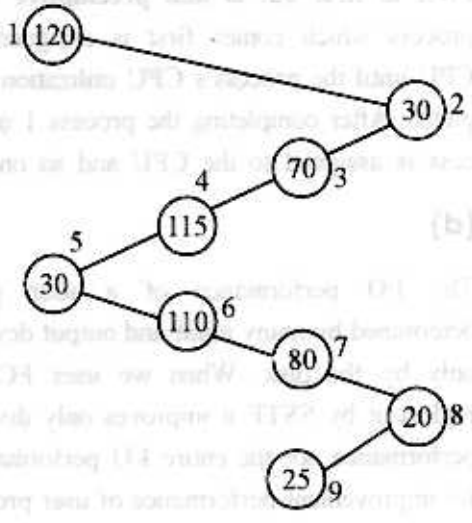
S.59 (a)

SSTF:



Direction change only at points 2 and 3, i.e., **2 times.**

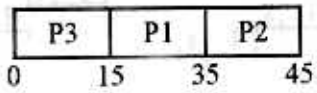
FCFS:



Direction changes at points 2, 5 and 8 i.e., **3 times.**

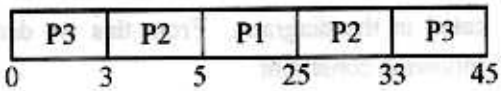
S.60 (d)

For Non-preemptive scheduling sequence of processes execution is



Turn around time for P2 = 43 – 3 (arrive time) = 42 sec

For preemptive scheduling sequence of processes execution is



Turn around time for P2 = 33 – 3 = 30 sec

S.61 (b)

ISR is invoked after complection of I/O irrespective of the type of I/O.

7.5 PROTECTION AND SECURITY

LEVEL–1

Q.1 Threat monitoring technique is

(a) the system can check for suspicious patterns of activity in an attempt to detect a security violation

(b) a time sharing system that counts the number of incorrect passwords given

(c) both (a) and (b)

(d) none of these

Q.2 The security of a system can be improved by

(a) threat monitoring

(b) audit log

(c) both (a) and (b)

(d) none of these

Q.3 A fire wall

(a) is a computer or router that sits between the trusted and untrusted

(b) it limits network access between the two security domains and maintains and logs all connections

(c) a firewall may need to allow http to pass

(d) all of these

Q.4 Encryption

(a) is one common method of protecting information transmitted over unreliable links

(b) Encrypted information is accessed by an unauthorized person or program, it will be useless unless it can be decoded.

(c) both (a) and (b)

(d) none of these

Q.5 In encryption mechanisms what is secret?

(a) Encryption and decryption algorithm

(b) key 'k' value

(c) both (a) & (b)

(d) data 'd'

Q.6 An entry in the access matrix A(i,k)

(a) i^{th} object is accessing the k^{th} resource

(b) the process executing in domain i can access k^{th} object

(c) the process executing k^{th} domain can access the object 'i' times

(d) none of the above

Q.7 Encryption is context dependent with respect to
(a) block cipher techniques
(b) private key encryption
(c) public key encryption
(d) stream cipher techniques

Q.8 Which implementation of access instruction is most deficient?
(a) global table
(b) access list
(c) capability list
(d) all similar efficiencies

Q.9 Capability list for a domain is defined as
(a) list of objects, with access rights applicable to them
(b) an ordered set <domain, right set>
(c) an ordered triple <domain, object, rights-set>
(d) list of objects and the operations allowed on those objects

Q.10 A process 'p_i' can access the object 'OK', if
(a) it is running a domain D_i
(b) it is running a domain which contains key
(c) it is running a domain which contains key top the lock in object OK.
(d) all the above

Q.11 A process operates within a field which specifies the resources that the process may access. The field is known as
(a) Protection domain
(b) Security domain
(c) Security Range
(d) Protection Range

Q.12 Which scheme of revocation does not allow selective revocation
(a) Reacquisition
(b) Back-pointers
(c) Keys
(d) Indirection

Q.13 Which statement is incorrect in case of domain related to protection?
(a) A domain is an abstract concept which can be realized in a variety of ways
(b) We cannot define a domain for each user but the set of objects which can be accessed depend on the identity of the user.
(c) Each process may be a domain
(d) Each procedure may be a domain

Q.14 In access matrix, row represents and column represents.
(a) domains, objects
(b) access rights, objects
(c) domains, access rights
(d) access rights, domains

Q.15 Match the following:

	Group I		Group II
1.	Interruption	(a)	Authenticity
2.	Interception	(b)	Integrity
3.	Modification	(c)	Availability
4.	Fabrication	(d)	Confidentiality

(a) 1 – b, 2 – a, 3 – d, 4 – c
(b) 1 – b, 2 – c, 3 – d, 4 – a
(c) 1 – c, 2 – a, 3 – b, 4 – d
(d) 1 – c, 2 – d, 3 – b, 4 – a

Q.16 Which are the advantages of Language-Based protections?
(i) protection needs are simply declared, rather that programmed as a sequence of calls on procedure of an operating system.
(ii) protection requirements may be stated independently of the facilities provided by a particular operating system.
(a) (i)
(b) (ii)
(c) neither (i) not (ii)
(d) both (i) & (ii)

Q.17 The best efficiency is obtained when enforcement of protection is

(a) directly supported by hardware

(b) directly supported by microcode

(c) indirectly supported by software

(d) Both (a) and (b)

Q.18 Authentication in Centralized Environment can be achieved by.

(i) A secret, known only to the user

(ii) Something possessed only by the user.

(iii) Some human characteristics of the user.

(a) only (i)

(b) (i) & (iii)

(c) (i), (ii), (iii)

(d) (i) & (ii)

Q.19 Pass phrase is password design technique mainly used for which of the following situation?

(a) Password length -- very short

(b) Password length – very long

(c) The password -- not enclosed – suppression and Encryption.

(d) The password – enclosed – non suppression and Encryption.

Q.20 Which statement is correct about capability used method of Access control Matrix?

(a) Small protection domains can be constructed in keeping with the principle of least privileges.

(b) The binding between high-level names and capabilities may be done using the directories to hold the pairs of names versus capabilities.

(c) Object naming and protection can be provided at runtime.

(d) (a) (b) (c) are correct.

Q.21 Need to know principle:

(a) is useful in limiting the amount of resources for a process in the system.

(b) is useful in limiting the amount of damage a faulty process can cause in the system.

(c) is not useful in limiting the amount of resources for a process in the system.

(d) is not useful in limiting the amount of damage a faulty process can cause in the system.

Q.22 Which statement is true, while distinguishing capabilities from other data?

(i) Each object has a tag to denote its type as either a capability or as accessible data.

(ii) The program address space splits into two parts one for program space (i.e. for data and instructions) and other part, containing the capability list.

(a) (ii)

(b) Both (i) and (ii)

(c) (i)

(d) Neither (i) nor (ii)

Q.23 Which one is not a valid scheme for implementing revocation for capabilities?

(a) Back–pointers

(b) Reacquisition

(c) Directions

(d) Keys

Q.24 The Trojan horse problem is due to.

(a) A code and a data segment both misuses its environment simultaneously

(b) A data segment misuses its environment

(c) A code segment misuse its environment

(d) None of these

Q.25 In case of UNIX system the number of protection fields are associated with each file.

(a) four

(b) three

(c) two

(d) none of these

Q.26 Match the following:

Group I		Group II	
1.	Confidentiality	(a)	requires that a computer system be able to verify the identity of a user.
2.	Integrity	(b)	requires that the information in a computer system only be accessible for reading bit authorized parties.
3.	Availability	(c)	requires that computer system assets can be modified only by Authorized parties.
4.	Authenticity	(d)	requires that computer system assets are available to authorized parties.

(a) 1 –b, 2 – a, 3 – d, 4 – c

(b) 1 – a, 2 – b, 3 – c, 4 – d

(c) 1 – a, 2 – d, 3 – b, 4 – c

(d) 1 – b, 2 – c, 3 – d, 4 – a

Q.27 When a user passes an object as an argument to a procedure, it may be necessary to ensure that the procedure cannot modify the object. This restriction can be implemented by.

(a) Passing an access right that does not have the modification-read right

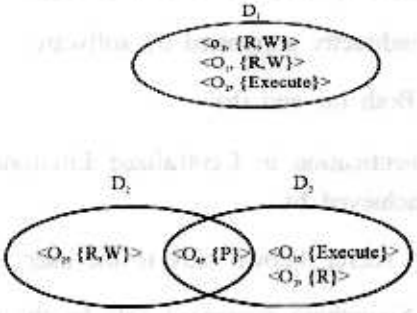
(b) Passing an access right that does not have the modification-write

(c) Passing an access right that has the modification-write right

(d) Passing an access right that has the modification-read right

Q.28 Consider a following system diagram with three protection domains with following operations. Which of the following statement is incorrect?

O – object, R – Read, W – write, P – print



(a) Domain D_2 and D_3 cannot share the object O_4 as printing so above diagrams incorrect.

(b) The access right $<O_4, \{P\}>$ is shared by both D_2 and D_3 implying that a process executing in either one of these two domains can print object O_4.

(c) If we want to share the domain we have to declare objects in all domains i.e. D_1, D_2 and D_3.

(d) Domains D_1, D_2, D_3 need not be disjoint; they may share access rights.

Q.29 Which of the following statement is incorrect?

(a) Virus are programs attacking the nodes on the network and spreading to other nodes.

They consume all the resources on the network and affect the response time.

(b) The virus detection program checks for the integrity of the binary files and written with a clear intention of infecting other programs.

(c) Redirect is one of the virus infection method in which the normal control flow of a program is changed to execute some other (normally viral) code which could exist as an appended portion of an otherwise normal program.

(d) The command processor infectors and general purpose infectors are the types of viruses.

Q.30 Which of the following statements is incorrect?

(a) The Access rights should be verified at every request from the subject.

(b) No access right should be granted to a process as a default. Each subject should have to demand the access right explicitly.

(c) It is not sufficient to verify access right only at the time of opening a file, the verification must be made at every read / write request.

(d) Checking for the access right only at the beginning and not checking subsequently may also degrade the system performance.

Q.31 The UNIX system uses to avoid the necessity of keeping its password list secret is.

(a) a variant of the algorithmic passwords

(b) an array of user passwords

(c) a link list representation of passwords

(d) none of these

Q.32 In UNIX, Access Control List (ACL) is kept in which part?

(a) BFD

(b) i–node

(c) file directory

(d) None of these

Q.33 A file F wants to perform both read and write operation on it, then domain for a file contains access right as.

(a) <{read, write}>

(b) <file F, {read, write}>

(c) <{read, write}, file F>

(d) <file F, read, write>

Q.34 Which statement is true about Revocation?

(i) Revocation is immediate

(ii) Revocation can be general or selective

(iii) Revocation can be total or partial

(iv) Revocation can be permanent or temporary

(a) (ii), (iii)

(b) only (i)

(c) (ii), (iii), (iv)

(d) (i), (ii), (iii), (iv)

Q.35 Which statement is correct in case of Security design?

(i) The design of the security system should not be a secret

(ii) Every process should be given the least possible privileges that are necessary for its execution.

(iii) No access rights should be granted to a process as a default.

(a) (ii), (iii)

(b) (i), (ii)

(c) (i), (ii), (iii)

(d) Only (ii)

Q.36 The role of protection in a computer system is to provide a mechanism for the enforcement of the policies governing resource use.

Then, identify the correct option related to establishment of these policies.

(i) Policies are fixed in the design of the system, while others are formulated by the management of a system.

(ii) Policies are defined by the individual users to protect their own files and programs.

(iii) Policies for resource use may vary, depending on the application, and they may be subject to change over time

(a) (i), (iii)

(b) (i), (ii), (iii)

(c) (i), (ii)

(d) (ii), (iii)

Q.37 In case of Access Matrix implementation using Global Table, whenever an operation M is executed on an object O_j within domain D_i, the global table is searched for a triple $<D_j, O_j, R_k>$ Where $M \in R_k$. If this triple is found then _____ otherwise it is _____.

(a) allowed to continue, exception is raised

(b) not allowed to continue, otherwise allowed to continue

(c) exception is raised, otherwise allowed to continue

(d) allowed to continue, otherwise not allowed to continue

Q.38 Consider a situation where one file is running as information source, and in between an unauthorized party gains the access over that source file and simultaneously that source also pass towards destination, then such a type of security threat is called as.

(a) Interruption

(b) Interception

(c) Fabrication

(d) Modification

Q.39 Consider a situation where user invokes any service program. During this process, user takes i^{th} risk that the program may malfunction and either damages the data or retains same access rights to the files to be used later. Similarly, the service program may have some private files that should not be accessed directly by the calling user program. This problem is called as

(a) Limiting propagation of access rights problem

(b) Trojan horse problem

(c) Mutually suspicious subsystem

(d) Revocation problem

Q.40 Memory protection is of no use in a

(a) single user system

(b) non-multiprogramming system

(c) non-multitasking system

(d) none of the above

Q.41 Memory protection is normally done by the

(a) processor and the associated hardware

(b) operating system

(c) compiler

(d) user program

Q.42 The access matrix consists of

(a) Global table

(b) Access list

(c) Capability list

(d) All of the above

Q.43 Malicious access are

(a) unauthorized reading of data

(b) unauthorized modification of data

(c) unauthorized destruction of data

(d) all of the above

Q.44 What encryption algorithm is used for storing the password database?

(a) RSA algorithm

(b) DES algorithm

(c) a modification of the DES algorithm

(d) MDS algorithm

Q.45 Trojan-Horse programs

(a) are legitimate programs that allow unauthorized access

(b) do not usually work

(c) are hidden programs that do not show up on the system

(d) usually are immediately discovered

Q.46 Domain and objects are represented by

(a) lines and tables respectively

(b) rows and columns respectively

(c) tables and columns respectively

(d) none of the above

Q.47 A firewall

(a) is a computer or router that sits between the trusted and untrusted

(b) it limits network access between the two security domains and maintains and logs all connections.

(c) may need to allow http to pass

(d) all of the above

Q.48 Which types of malicious program require a host program?

(i) Trapdoors (ii) Logic bombs

(iii) Viruses (iv) Worms

(a) (i), (ii) and (iv)

(b) (ii), (iii) and (iv)

(c) (ii) and (iii)

(d) (i), (ii) and (iii)

Q.49 With which type of security are passwords assigned to resources?

(a) group level

(b) user level

(c) share level

(d) resource level

Q.50 Some computer systems support dual mode operation the user mode and the supervisor or monitor mode. These refer to the modes:

(a) by which user programs handle their data

(b) by which the OS executes user programs

(c) in which the processor and associated hardware operate

(d) of memory access

Q.51 Convoy effect is a result of

(a) one log CPU bound process and many other CPU bound processes are waiting

(b) many CPU bound processes and less I/O bound processes

(c) many CPU and I/O bound processes

(d) proper mix of CPU and I/O bound processes

Q.52 Which statement is incorrect in case of domain related to protection?

(a) we cannot define a domain for each user but the set of objects which can be accessed depend on the identity of the user

(b) a domain is an abstract concept which can be realized in a variety of ways

(c) each process may be a domain

(d) each procedure may be a domain

Q.53 Match List-1 with List-2 and select the correct answer using the codes given below the lists:

List-1		List-2	
A.	Virtual uniprocessor	1.	No determine ordering of events
B	Deadlock resolution	2.	A message received by P_j from P_i that is not shown as sent by P_i
C	Concurrrency	3.	The desirable user perspective is a distributed system
D	Inconsistency	4.	Always requires the abortion of one more executing processes

Codes:

	A	B	C	D
(a)	4	3	1	2
(b)	3	4	2	1
(c)	3	4	1	2
(d)	3	2	4	1

GATE QUESTIONS

Q.54 Dynamic linking can cause security concerns because **[GATE 2002]**

[2-Marks]

(a) Security is dynamic

(b) The path for searching dynamic libraries is not known till runtime

(c) Linking is insecure

(d) Cryptographic procedures are not available for dynamic linking

ANSWER KEY

1	c	2	a	3	d	4	c	5	b
6	b	7	d	8	c	9	d	10	c
11	a	12	b	13	b	14	a	15	d
16	d	17	d	18	c	19	a	20	d
21	a	22	a	23	a	24	a	25	c
26	c	27	b	28	c	29	a	30	d
31	a	32	d	33	a	34	b	35	d
36	c	37	d	38	c	39	a	40	d
41	a	42	d	43	d	44	c	45	a
46	b	47	d	48	d	49	c	50	c
51	a	52	a	53	c	54	d		

SOLUTIONS

S.2 (a)

Audit log is not related to security but it is a method used for tracking temporal information. Anytime something significant happens we write some record indicating what happened and when happened.

Threat monitoring is concerned with system's security.

S.5 (b)

In designing security systems, it is wise to assume that the details of the cryptographic algorithm are already available to the attacker. Therefore, Only **secrecy of the key provides security.**

S.6 (b)

Access matrix has two possible representations

(i) Only hold information of the rows-each row corresponds to the access rights of a domain over all objects it can use. If the domain has no rights over an object no information is stored. This approach is known capability lists.

(ii) Only hold information on the columns-each column represents the access rights hold over the object. No information is stored about domains that have no access. This approach is known as access list.

S.11 (a)

A process operates within a **protection domain** which specifies the resources that the process may access and domain is a collection of access right.

S.12 (b)

The capabilities do not point to the objects directly, but instead point indirectly. It does not allow selective revocation.

S.13 (b)

A domain may be defined for each user. The set of objects which can be accessed depend on the identity of the user. Domain switching occurs when the user is changed, generally when a user logs–out and another user logs–in.

S.15 (d)

1. **Interruption** is an attack on **availability.**
2. **Interception** is an attack on **confidentially.**
3. **Modification** is an attack on **integrity.**
4. **Fabrication** is an attack on **authenticity.**

S.16 (d)

When protection is declared along with data typing, the designers of each subsystem can specify their requirements for protection, as well as their need for use of other resources in a system. Such a specification should be given directly as a program is composed, and in the language in which the program itself is stated and (i) and (ii) are advantages of this approach.

S.18 (c)

Authentication in centralized Environment can be achieved by

(i) **A secret** like password

(ii) **Something possessed only by the user** like the artifact such as a magnetic badge.

(iii) **Some human characteristics of the user** like handwritten signatures, fingerprints, retina images.

S.19 (a)

Pass phrase is used, especially if the password length is very short. Along with each password a link or meaningful message or phrase is predetermined.

S.20 (d)

(a) (b) (c) are the advantages of capability based method of Access control Matrix.

S.21 (a)

The term "Need to know" describes the restriction to data which is considerd very sensitive even having necessary approvals or security clearance, to access certain information, one would not be given access to such information, unless one has a specific need to know resource.

S.23 (a)

The capabilities do not point to the objects directly, but instead point indirectly. It does not allow selective revocation.

S.27 (b)

The situation given in the problem is a protection problem and this is implemented by passing an access right that does not have the modification right.

S.29 (a)

The definition given in the option (b) is for worm not for virus.

(b) (c) (d) are correct statements.

S.40 (d)

Even in a non-multiprogramming system, memory protection may be used, when, for example, spooling is being used.

S.48 (d)

Trapdoors, logic bombs and viruses are malicious programs that require a host program.

S.51 (a)

CPU bound processes requires lot of processor time, resulting in long wait for I/O bound processor. This effect is called convey effect. It results in lower CPU and I/O devices utilization.

S.52 (a)

A domain may be defined for each user. The set of objects which can be accessed depend on identity of the user. Domain switching occurs when the user is changed, generally when a user log-out and another user log-in.

S.54 (d)

Dynamic linking can cause security concerns because cryptographic procedures are not available for dynamic linking.