II APPENDIX

APTITUDE QUESTIONS IN C

[1] BASICS

1)
```
#include<stdio.h>
int a;//Statement 1
int abc(int a)//Statement 2
{
int a;//Statement 3
a=5;//statement 4
}
int main() {
abc();
return 0;
}
```

Which ‘a’ would be changed when this code is executed?

(a) Statement 1 (b) Statement 2 (c) Statement 3 (d) Compilation error

ANSWER: (d) Compilation error

REASON: “error: ‘a’ redeclared as different kind of symbol”,shows this error since a is redeclared which is not allowed in C. Removing statement 3 will make it compile fine and change value for a in statement 4 giving “::” before a will change the value in statement 1 (global variable).

2)
```
#include<stdio.h>
int main()
{
  int a,b;
  printf("%d",scanf("%d",&a));
  return 0;
}
```

What would be the output (assume the given input is 5)?

(a) 5 (b) 1 (c) –1 (d) Compilation error

ANSWER: (b) 1

REASON: Since printf() has a right to left associativity ,the inner scanf is first executed,scanf() returns the number of inputs given,it returns 1 and that is printed in outer printf().

(3)
```
#include <stdio.h>

int fun(int a)
{
  a>5?return 0:return 1;
  }
int main() {
  printf("%d",fun(60));
  return 0;
}
```

What would be the result?

(a) Compilation error (b) Runtime error (c) 0 (d) 1

ANSWER: (a) Compilation error

REASON:“error: expected expression before ‘return’ ”.The three components of the ternary operator are expressions and return is a **statement.** Since return is a statement, hence it cannot be used where a value is expected.

4)
```
nt main() {
  float a=3.567;
  int b=3;
  a==b?printf("True "):printf("False ");
  (int)a==b?printf("True "):printf("False ");
          a==3.567?printf("True"):printf("False");//statement 1
  return 0;
}
```

What would be the output?

(a) False True True (b) False False False (c) False True False (d) True False True

ANSWER: (c) False True False

REASON: The first one is false since 3.567!=3.The second one is true since a is “type casted” to an int(3) and compared with b whose value is also 3.The third one is false since the expression 3.567 is a **double constant** since ‘a’ is float, it is promoted to double and the values are compared now with more precision bits causing a mismatch(giving false).Type casting statement 1 to 3.567f can result in true.

5)
```
#include<stdio.h>
main()
{
```

```
Int a =23;
char b = 'a';
int sum;
sum = a+b;
printf("%d",sum);
}
```

Note: The ASCII value of a is 97

What would be the result?

(a) 23 (b) 120
(c) Compilation error (d) No output but compiles fine

ANSWER: (b) 120

REASON: The data types which are smaller than int or unsigned int are converted to **either int or unsigned int.** Here char being a smaller data type than int is self promoted to char before the result is added up giving us a result of 120.

6)
```
#include <stdio.h>
int main() {
printf("\\\\\\");//statement 1
printf("//////");//statement 2
return 0;
}
```

How many slashes will be printed at output for statement 1 and 2?

(a) 6 6 (b) 6 3 (c) 3 6 (d) 3 3

ANSWER: (c) 3 6

REASON: A single \ is an "escape character" which is used to indicate a special character sequence until another \ is encounter ,so a single "\\" is an escape character that prints a single backslash(\). So three backslash are printed.

7)
```
#include<stdio.h>
int main() {
int i;
i = 10;
printf("%d", i , ++i);
  return 0;
}
```

What would be the output?

(a) 10 (b) 11 (c) Compilation error (d) 10 11

ANSWER: (b) 11

REASON: The printf() function has an associativity from "right to left",first ++i is incremented to 11 and now i value becomes 11,since there is only one %d, 11 is printed as the result.

```
8) #include <stdio.h>
   int main(){
     int a=5;
     if(a==5 &&  (a=6) )
     printf("%d  ",a);
     if(a=6 && a==6)
     printf("%d",a);
     return 0;
   }
```

What would be the output?

(a) Compilation error (b) 6 6 (c) 6 1 (d) Runtime error

ANSWER: (c) 6 1

```
9) #include<stdio.h>
   int main()
   { int a,b;
   a =1,2,3; // statement 1
   b =(4,5,6);// statement 2
   printf("%d %d",a,b);
   return 0;
   }
```

What would be the output?

(a) 1 4 (b) 1 6 (c) 3 6 (d) Compilation fails

ANSWER: (b) 1 6

REASON: Commas in C can work as eitherseparators or operators. Here in the following code, they work as an operator. In statement 1, "assignment operator has a greater precedence over comma, so statement 1 is interpreted as (a=1),2, 3.In statement 2, commas are inside parenthesis, so comma is executed first and the final value 3 is assigned to a.

```
10) #include <stdio.h>
    int main(){
      int a=5,b=10;
      a&=b;
      if(a)
      printf("Output present");
    }
```

What would be the output of the following?

(a) No Output (b) Output Present (c) Compilation error (d) None of the above

ANSWER: (c) Compilation error

REASON: In C, there are no logical assignment operators. The statement a = a&b gives an error.

```
11) #include <stdio.h>
    int main(){
      printf("%d",printf("hey"));
    }
```

What would be the output?

(a) hey (b) 3hey (c) hey3 (d) Compilation error

ANSWER: (c) hey3

REASON: The printf() has a right to left associativity,so first inner printf is executed (hey) and this inner returns 3 which is printed by the outer printf().

```
12) #include<stdio.h>
    int main()
    {
      if(printf("Hello"))
        printf("Hello");
      else
        printf("World");
      return 0;
    }
```

What would be the output?

(a) Hello (b) Hello World (c) Hello Hello (d) Compilation error

ANSWER: (c) Hello Hello

REASON: The printf() inside the if condition prints "Hello" and returns 5. In C, any other value other than zero is taken as true, so another Hello is printed.

[2] POINTERS

```
13) #include<stdio.h>
    #include<string.h>
    char *a;
    char* fun()
    {
    return(a);
            }
    {
            char *ptr;
            a=(char*)malloc(20);
            //statement 1
            strcpy(a,"Done");
            strcpy(ptr,fun());
            free(a);
```

```
        printf("%s ",ptr);
return 0;
        }
```

What line inserted at statement will produce output 'Done'?

(a) Not possible (b) ptr=(char*)malloc(10)
(c) ptr="Done" (d) strcpy(ptr,"Done")

ANSWER: (b) ptr=(char*)malloc(10)

REASON: malloc() allocates required size of bytes and returns a void pointer pointing to the first byte of the allocated space. Before even placing the values, it is important to allocate memory to it first.

14)
```
#include <stdio.h>
int main() {
  int x=50,*y;
  y=&x;
  x++;
  printf("%d %d",x,*y);
  return 0;
}
```

What will be the output?

(a) 50 50 (b) 51 51 (c) 50 51 (d) 51 50

ANSWER: (b) 51 51

REASON: A pointer variable y is assigned to x.Say, if x is stored in address location 1000,then y is pointing to 1000 and *y is pointing to the value in 1000(which is 50). When we increment x and the pointer is pointing to the value x,it is the incremented value of x that y holds now.

15)
```
#include <stdio.h>
char * fun()
{
    return "hello";
}
int main()
{
  printf("%s",fun()+printf("he"));
}
```

What would be the output?

(a) hello (b) 7hellohe (c) he (d) Compilation error

ANSWER: (a) hello

REASON: Since printf has a right to left associativity, first "he" is printed. This inner printf() statement returns the value 2 that is to be incremented by the pointer function "char *fun()".So **the function returns "llo"** and the final output is hello.

16)
```
#include<stdio.h>
int main()
{
  char *p;
  p ="hello";
  printf("%c %c",*p,*&*p++);
  p++;
  printf("%c",*&*p);
}
Note: & is a reference operator.
```

What will be the output?

(a) h el (b) e hl (c) l eh (d) Compilation error

ANSWER: (b) e hl

REASON: Since & is a reference operator, *& will be dereferencing and referencing and *&p++ is same as *p++.Again printf() goes from right to left,so *p++, when it has reached the *p value, is incremented by 1 and hence, ‘e’ and then ‘h’ is printed.After that the pointer is incremented and now points to ‘l’ and that is printed.

17)
```
#include<stdio.h>
#include<string.h>
int main(){
Register  a = 25;
int *p;
p=&a;
printf("%d ",*p);
*p++;
Printf("%d",*p);
return 0;
}
```

What would be the output?

(a) 25 adress location (b) address location adress location (c) Compilation error

ANSWER: (c) Compilation error

REASON: “error: address of register variable ‘a’ requested” is displayed,since when one says a variable is register then it may be stored in CPU(register) and accessing this address is invalid. Although register keyword can be used with pointer variables like “register int*a =&i”.

18)
```
#include<stdio.h>
int* call(){
  int x=15;
  ++x;
    return &x;
}
```

```
void main(){
       int *ptr;
    ptr=call();
    printf("%d",*ptr);
  }
```

What would be the output of the following?

(a) 15 (b) 6 (c) Compilation error (d) Garbage value

ANSWER: (d) Garbage value

REASON: Here the variable x is a local variable and its scope is within the function call() and after returning the address, the variable x becomes dead but still ptr is pointing to that location . This is commonly called as **Dangling Pointer.**

19)
```
#include<stdio.h>
#include<stdlib.h>
int main()
{
char s[]= "Hello World";
int i = 0;
while(*(s++)) //statement1
i++;
printf("%d",i);
return 0;
}
```

Predict the following output?

(a) 11 (b) 0 (c) Compilation error (d) Segmentation fault

ANSWER: (c) Compilation error

REASON: "**error: value required as increment operand"**is displayed. Here,'s' is a character array and array names are not modifiable,trying to modify its value say at statement1 will cause an error.

20)
```
#include<stdio.h>
Int main()
{
Struct node
{
int a;
int b;
int c;
int d;
};struct node q = {1,2,3,4}
Struct node *ptr = &q;
Printf("%d",//insert code here); // statement 1
return 0;
}
```

What code when inserted at statement 1 would produce an output '2'?

(a) *(*ptr+1) (b) **ptr++ (c) *((int)ptr+1) (d) *((int*)ptr+1)

ANSWER: (d) *((int*)ptr+1)

REASON: Since we need an output 2 to be displayed,we need to increment the ptr pointer by 1 point to the next location. Initially ptr holds an address location, say 1000 and *p holds the address of the structure's initial address, say 2000, which holds the value 1.So we need to double de-reference it.

21)
```
#include<stdio.h>
int main()
{
const int x=5;
const int *ptr;
ptr = &x;//statement 1
*ptr = 10;//statement 2
printf("%d\n", x);
return 0;
}
```

Predict the output or spot errors (if any)?

(a) Output:5 (b) Error in statement 1 (c) Error in statement 2 (d) Output:10

ANSWER: (c) Error in statement 2

REASON: Since the constant pointer is pointing to a constant variable,its value that it is pointing to cannot change,thus results an error in that statement.

22)
```
#include <stdio.h>
   int main(void)
{
   int a = 20;
   int b = 40;
   const int *const ptr = &a;  //statement1
   printf("ptr: %d\n", *ptr);
   ptr = &b;
   *ptr = 100;              //statement 2
   return 0;                // statement 3
}
```

Predict the errors in the code (if any)?

(a) Error in statement 1 and 2
(b) Error in statement 2 and3
(c) Error in statement 1 and 3
(d) Works fine

ANSWER: (b) Error in statement 2 and 3

REASON: There is no error in statement 1 as it is a constant pointer to a constant. Since it is a constant pointer it cannot change its reference address from a to b. and also cannot change the value that it is pointing to.

[3] MACROS

23) Predict the following output?

```
#include<stdio.h>
#define x 4+1
int main()
{
int i;
i=x;
printf("%d",i);
i=x*x;
printf("%d",i);
i = x*x*x;
printf("%d",i);
return 0;
}
```

(a) 5 9 13 b) 5 25 125 (c) 4 16 64 (c) 5 10 14

ANSWER: (a) 5 9 13

REASON: x=4+1=5. x*x = 4+(1*4)+1=9 x*x*x = 4+(1*4)+(1*4)+1.=13

24)
```
#include <stdio.h>
#define HEY_I_FIND_MIN(x,y) x>y?y:x;//statement  1
int main() {
  int a=10,b=8,max;
  max=HEY_I_FIND_MIN(++a,b);
  printf("%d %d %d",a,b,max);
  return 0;
}
```
Output
```
11 8 8
```

What would be the output if direction of operator is reversed in statement 1 (i.e., "<" this operator is used)?

(a) 12 8 12 (b) 11 8 11 (c) 11 8 8 (d) 11 8 10

ANSWER: (a) 12 8 12

25)
```
#include <stdio.h>
#define square(x) (x*x)
int main(){
     int a,b=5;
     a= square(b+1);
     printf("%d",a);
     return 0;
}
```

What would be the output?

(a) 11 (b) 36 (c) Garbage Value (d) Compilation error

ANSWER: (a) 11

REASON: 5+(1*5)+1 =11.Multiplication has greater priority than addition.

26)
```
#include <stdio.h>
#define MACRO(num, str) {\    //statement 1
printf("%d", num);\
printf(" is");\
printf(" %s number", str);\
printf("\n");\
}
int main(void)
{
int num =5 ;
if (num)
MACRO(num, "Odd");
else         //statement 2
MACRO(num, "Even");
return 0;}
```

Predict the output.

(a) 5 is odd (b) 5 is even (c) Error in statement 1 (d) Error in statement 2

ANSWER: (d) Error in statement 2

REASON: "error: **'else' without a previous 'if'" error** is generated,this is because when the marco function is expanded below if,there are many statements below if,so "else" does not have a if.If we put curly braces{ } for MACRO(num,"odd"),then we will get the output-"5 is odd".

27)
```
#include<stdio.h>
#define CUBE(x) (x*x*x)
int main()
{
  int a, b=3;
  a = CUBE(b++);//statement1
  printf("%d, %d\n", a, b);
  return 0;
}
A.   27, 3
B.   27, 4
C.   27, 6
D.   Error
```

Answer: Option C

REASON: When the macro is expanded in statement1 3++ * 3++ *3++,a is stored with a value (3*3*3) and b is incremented thrice.

[4] STRING FUNCTIONS

28)
```
#include<stdio.h>
#include<string.h>
int main()
{
  int a,b;
  a = sizeof("hello");
  b=strlen("hello");
  printf("%d %d",a,b);
    return 0;
}
```

What would be the output if the code was executed?

(a) 5 5 (b) 6 5 (c) 6 6 (d) 6 5

ANSWER: (b) 6 5

REASON: The sizeof() even takes the null charcter(\0) into account, whereas strlen doesn't take the null character.

29)
```
#include<stdio.h>
#include<string.h>

int main()
{
char *a[5]={"guess","the","out","put"};
char **ptr[]={a,a+1,a+2,a+3},***p;
p=ptr;
strcpy(a+2,a+3);
p+=2;
printf("%s",**p);
return 0;
  }
```

WHAT will happen if strcpy is changed to strcat?

(a) Output will be:output (b) output will be:put

(c) Runtime error d) Compile error

ANSWER: (c) Runtime error

REASON: The size of strcat(a+2,a+3) would give a[2]=output which takes 6 characters which exceeds the size of a block in the array.

[5] LABEL STATEMENTS

30)
```
#include <stdio.h>
int main() {
int i = 1;
  switch(i)
  {
    ++i;
    case 1:
      printf("Case1");
      break;
    case 2:
      printf("Case2");
      break;
  }
    return 0;
}
```

What would be the output?

(a) Error: Invalid printf statement after switch statement

(b) Case1 (c) Case2 (d) None of the above

ANSWER: (b) ase1

REASON: After the switch(i) was executed, the control transfers to case 1 since i value was 1 and case 1 statements are executed.

31)
```
#include<stdio.h>
Int main()
{
Int a =1;
Switch(i)
{
Case 1:
printf("hello");
goto default;
case 2 :
printf("hello");
break;
default:
printf("world");
}
}
```

Predict the output?

(a) hello world (b) hello (c) Runtime error (d) Compilation error

ANSWER: (d) Compilation error

REASON: "error: **expected identifier or '*' before 'default'",** the label default is already a keyword and cannot be called. One can use a label before default to execute both case statements and default statements.

32)
```
main()
   {
int i =1 ;
while(i<=10)
{
printf("%d",i);
If(i>2)
goto here;
hello:
i++;
}
Here:
printf("2");
goto hello;
}
```

What would be the output?

(a) 12324252 (b) 12222 (c) Infinite loop (d) Compilation error

ANSWER: (c) Infinite loop

REASON: The loop doesn't terminate,it swings to and fro from label hello to label here.

[6] FILE HANDLING

33) Assume a file "hello.txt"exist
```
#include<stdio.h>
#include<stdlib.h>
int main()
{
File *fp = fope("hello.txt",wx)
if(fp==NULL)
{
exit(0);
}
else
{
fputs("Hello world",fp);
puts("implemented");
close(fp);
}
return 0;
]
```

What would be the output?

(a) ”Hello World” contents are written to file hello.txt
(b) Compilation error
(c) Runtime error
(d) Program exits

ANSWER: (d) Program exits

REASON: In “wx” the file pointer will return “null” if the file already exists.

34) Assume a file called “trial”

```
#include<stdio.h>
int main()
{
unsigned char ch; //statement1
File *fp
fp = fopen(“trial”,”r”);//statement 2
while((ch=getc(fp)!=EOF) //statement 3
printf(“%c”,ch);
fclose(fp);
return 0;
}
```

Point out the errors(if any) in the code?

(a) Statement 1 (b) Statement 2 (c) Statement 3 (d) No error

ANSWER: (c) Statement 3

REASON: EOF returns -1 which cannot be compared to an unsigned char.

35) Assume a filename “example.txt”

```
#include<stdio.h>
int main()
{
FILE* pFILE;
pFILE=fopen(“example.txt”,”wb”);
fputs(“this is an apple.”,pFILE);
fseek(pFILE,9,SEEk_SET);
fputs(“sam”,pFILE);
fclose(pFILE);
return 0;
}
```

Predict the following output?

(a) this is a sample (b) this is an sample (c) this is asampple (d) Compilation error

ANSWER: (c) this is asampple

REASON: fseek() sets the position indicator associated with the *stream* to a new position. SEEK_SET indicates the beginning of the file. The 9th position from the beginning of the file is the character 'n' to write 3 positions, from there would overwrite 'n',a space,'a'.

[7] ENUMERATIONS

36)
```
#include <stdio.h>
enum day {sunday, monday, tuesday, wednesday, thursday = 10, Friday, saturday};
int main(){
  printf("%d %d %d %d %d", sunday, monday, Friday,Saturday);
    return 0;
}
```

Predict the following output?

(a) 0 1 11 12
(b) 1 2 11 12
(c) 0 1 11 12 garbage value
(d) Compilation error

ANSWER: (c) 0 1 11 12 garbage value

REASON: There are five %d for four specifiers in the printf statement, so final one has to be a garbage value. ENUM is a user defined data type where many integral constants are assigned a name. Default starting value is 0(from left to right).

37)
```
Predict the following output?
#include <stdio.h>
int hello(int a ,int b=1)
{
  return a+b;
}
int main()
{
  printf("%d",hello(5,6));
  return 0;
}
```

(a) 11 (b) 6 (c) 7 (d) Error in output

ANSWER: (d) Error in output

REASON: There are no default arguments in C. Alternatively we can use var_args to specify values.

[8] STRUCTURES

38)
```
What would be the output?(Note: This is compiled in a turbo C/C++ compiler)
a)int main() {
```

```
struct STUDENT{
int rollno;
char stdname[20];
float percentage;
} *s1;
printf("\Enter student details: ");
scanf("%d %s %f",&s1->rollno,s1->stdname,&s1->percentage);
printf("\nThe entered details are: ");
printf("Roll: %d, Name: %s, Percentage: %f ",s1->rollno,s1->stdname,s1->percentage);
getch();
return 0;
}
```

What would be the output(if any)if custom inputs were "123","ram","81.5"?

(a) rollno:123
```
Name:ram
Percentage:81.5
```
(b) Compilation error (c) Runtime error (d) None of the above

ANSWER: (c) Runtime error

REASON: "error: *Scanf: floating point formats not linked* and the program gets terminated abnormally". This is because s1->is a pointer to a structure pointing to a float variable, as the compilers like turbo C compilers do no link floating points unless necessary. This is solved by adding a dummy floating point function

39)
```
struct print
{
int x;
int y;
};
#include<stdio.h>
int main()
{
struct print x={100,200,300};
struct print y=x;
if(x==y)
{
printf("the contents are the same");
}
else
printf("the contents are similar but a different address");
return 0;
}
```

What would be the output?

(a) The contents are the same
(b) The contents are similar but a different address

(c) Compilation error
(d) Compiles fine but no output

ANSWER: (c) Compilation error

REASON: Two instance of a structure can never be compared,only their individual members are compared.

40) Predict the output when executed on a turbo C++ compiler?
```
struct door
{
  int i;
  float f;
  char *c;
};
union bell
{
  int i;
  float f;
  char *c;
};
int main()
{    struct door s;
     union bell s1;
     printf("%d",sizeof(s1));
     printf("%d",sizeof(s));
     return 0;
}
```
Note: To deallocate the memory without free() function,reallocate the pointer to realloc(ptr,0)

(a) 8 4 (b) 4 4 (c) 8 8
(d) Compilation error since same variable names used

ANSWER: (a) 8 4

REASON: Structure "door" takes into account all the data that it encloses as its size(an integer variable so 2,float 4 and character pointer 2 2+4+2=8), whereas union takes only the highest size data type as its size float 4.This is a basic difference between union and structure.

[9] BIT FIELDS IN C

Note: We use bit fields for better memory utilization.

41)
```
#include <stdio.h>
struct book
{
  unsigned int x: 5; //statement1
```

```
    unsigned int y: 8;
  };
  struct hello
  {
    int x: 5;
    unsigned int: 0;   //statement2
    unsigned int y: 8;
  };
  int main()
  { struct book s;
  struct hello a;
    printf("%d, sizeof(s));
    printf("%d", sizeof(a));
  return 0;
  }
```

Predict the output (or detect errors if any)?

(a) 4 4 (b) 4 8 (c) Error in statement 1 (d) Error in statement 2

ANSWER: (b) 4 8

REASON: The unsigned int x:5 means that's x has stored 5 bits in structure. Bitfields are used to save memory; they are used to ensure that we use only the memory we require. The bit fields in the structure are not in an alignment. There is no error is statement 1but rather forces an alignment to the next state statement "unsigned int y:0" which makes the total size of struct hello as 8 bytes.

42)
```
#include <stdio.h>
  struct test
  {
        int book: 5; //statement 1
      int hello;
  };
  int main()
  {
    struct test t;
    printf("%p ,%p",&t.book,&t.hello); //statement 2
      return 0;
  }
```

Predict the following output(or errors if any)?

(a) address of book , address of hello
(b) Error in statement1
(c) Error in statement 2
(d) Compiles fine but no output

ANSWER: (c) Error in statement2

REASON: "error: cannot take address of bit-field 'book'".We cannot have pointers to bit fields because they may not start at an address boundary.

43) Predict the following output?

```
struct hello
{
  unsigned int x[10]: 5;//error in statement 1
};
int main()
{

}
```

(a) Compiles fine but no output (b) Error in statement 1

(c) Compilation error int main() has no return

ANSWER: (b) Error in statement 1

REASON: In C, array of bit fields are not allowed

44)
```
#include<stdio.h>
int main()
{
struct hello
{
unsigned a:5;
unsigned c:5;
unsigned b:6;

}book;
char *p;
struct hello *ptr,book1={1,3,3}; // statement 1
p=&book1;
p++;
printf("%d",*p);
return 0;
}
```

Predict the following output?

(a) 5 (b) 12

(c) Compilation error in statement 1 d) 3

ANSWER: (b) 12

REASON: Binary value of a=1 is 00001 (in 5 bit).Binary value of c=3 is 000011 (in 6 bit)
Binary value of b=3 is 00011 (in 5 bit).Let address of book1 is 500 which is initialized to char pointer p. Since char is one byte data type, so p++ will be 501. *p means content of memory location 501 which is (00001100)and its binary equivalent is 12. Hence output is 12.

[10] STORAGE CLASSES IN C

45)
```
#include<stdio.h>
int main(){
  int i;
  auto char c; //statement1
  printf("%d  %c",i,c); //statement2
  return 0;
}
```

Predict the following output?

(a) Garbage garbage
(b) Compilation error in statement 1
(c) Compilation error in statement 2
(d) None of the above

ANSWER: (a) garbage garbage

REASON: Default initial value of auto is garbage.

46)
```
#include <stdio.h>
extern int i;    //extern variable
int main(){
  printf("%d",i);
  return 0;
}
```

(a) 0 (b) garbage value (c) Compilation error (d) Runtime error

ANSWER: (a) 0

REASON: The extern storage class is used to give reference of a global variable that is visible to ALL the program files.The default initial value stored here is 0.

45)
```
#include<stdio.h>
int main()
{
  register int a = 10;..statement 1
  int *ptr = &a;) //statement2
  printf("%d", *ptr);
  return 0;
}
```

Predict the output(or find errors if any)?

(a) 10 (b) address (c) Error in statement1 d) Error in statement2

ANSWER: (d) Error in statement 2

REASON: Trying to access the address (using & operator)of register may give you an error since register variables are stored in register(cpu) instead of memory.

47)
```
#include<stdio.h>
int main()
{
  int a = 10;
  register int *ptr = &a;
  printf("%d", *ptr);
  return 0;
}
```

Predict the following output?

(a) 10 (b) address (c) Compilation error (d) Runtime error

ANSWER: (a) 10

REASON: Although one cannot access the address of a register variable,we can associate pointer with register variables.

48)
```
#include<stdio.h>
int main()
{
  int a = 10;
  register static int *ptr = &a;
  printf("%d", *ptr);
  return 0;
}
```

Predict the following output?

(a) 10 (b) address (c) Compilation error (d) None of the above

ANSWER: (c) Compilation error

REASON: "error: **multiple storage classes in declaration specifiers"** is displayed. Register is a storage class and C doesn't allow multiple storage classes specifiers for a variable.

49)
```
#include<stdio.h>
int main(){
  int a=10;
  {
    int a=20;
    printf("%d",a);
  }
  a++;
  printf(" %d",a);
  return 0;
}
```

Predict the following output?

(a) 20 21 (b) 20 12 (c) 20 11 (d) Compilation error

ANSWER: (c) 20 11

REASON: The ‘a’ within the inner curly braces has its scope only within that braces. So when printed outside, a=10 is incremented and printed.

```
50) #include<stdio.h>
    int main(){
      register int a,b;
      scanf("%d%d",&a,&b);
      printf("%d  %d",a,b);
    }
```

Assume the custom inputs are 5 and 5,predict the following output?

(a) 5 5 (b) 5 6 (c) address address (d) Compilation error

ANSWER: (d) Compilation error

REASON: We cannot deference a register variable since it has no memory address. So using scanf() on a register variable causes an error.

```
51) #include <stdio.h>
    static int a;
    static int a=15;    //statement 1
    static int a;
    int main(){
      static int a;
      printf("%d",a);   //statement 2
      return 0;
    }
```

Predict the following output(or errors if any)?

(a) 15 (b) 0 c) Error in statement 1 (d) Error in statement 2

ANSWER: (b) 0

REASON: A same static variable can be used many times but we can initialize it only once,so the above code works fine. Since we are requesting the value of the local scope ‘’a”, its default value 0 printed.

```
52) #include <stdio.h>
    static int i=10;   //statement 1
    i=15;              //statement 2
    int main(){

       printf("%d",i);
       return 0;
    }
```

Predict the following output?

(a) 10 (b) 15 (c) 0 (d) Compilation error

ANSWER: (d) Compilation error

REASON: We cannot assign a value globally. Sstatement1 is where we are initializing 1 to 10 and so no problems are there but statement 2 we are assigning 1 to 15 which cannot be done globally.

53)
```
#include <stdio.h>
static int i=10;   //statement 1

int main(){
  i=15;            //statement 2
  printf("%d",i);
  return 0;
}
```

Predict the following output?

(a) 10 (b) 15 (c) Error in statement 2 (b) 0

ANSWER: (b) 15

REASON: Assigning operation can be within a local scope.

54)
```
#include <stdio.h>
static int i=10;
int main(){
  i=5;
  ++i;
  for(i=0;i<5;i++){
    static int a=10; //statement 1.
    printf("%d",a++);//statement 2

  }
  printf("%d",i);
  return 0;
}
```

Predict the following output?

(a) 10 10 10 10 10 6 (b) 10 11 12 13 14 5 (c) 10 11 12 13 14 6 (d) 10 10 10 10 10 6

ANSWER: (b) 10 11 12 13 14 5

REASON: A static variable will initialize its value only once. So "static int a =10"is executed only once. In the 'for loop', I value is initialized to 5 and loop breaks and that value is printed.

55)
```
#include <stdio.h>

int main(){
  extern int i;
  int i=10;
  printf("%d",i);
  return 0;
}
```

Predict the following output?

(a) 10 (b) address (c) Compilation error (d) None of the above

ANSWER: (c) Compilation error

REASON: "error: **declaration of 'i' with no linkage follows extern declaration".**Error says that extern variables should follow a global declaration.

```
56) #include <stdio.h>
    int main()
    {
      typedef static int points;
      points x = 5;
      printf("%d ", x);
      x++;
      printf("%d ", x);
      return 0;
    }
```

Predict the output?

(a) 5 6 (b) 5 5 (c) Compilation error (d) None of the above

ANSWER: (c) Compilation error

REASON: "error: multiple storage classes in declaration specifiers". Since typedef is a storage class,it cannot be static.

[11] C PUZZLES

```
57)void fun()
    {
      // statement1
    }
    int main()
    {
      int i = 10;
        fun();
        i = 15;
        printf("%d", i);
        return 0;
    }
```

Insert a code at statement 1 so the printf() changes the value in main() to 12

Hint: Use MACRO ARGUMENTS

ANSWER: #define printf(x, y) printf(x, 12);

58) Write a C program that printfs “hello world” without using semicolon.

```
#include<stdio.h>
int main()
{
     if( printf( “hello world” ) )
     {    }
}
```

59) Write a c program to printf “ hello world” without using main()

HINT: Use a macro that defines main

```
#include<stdio.h>
#define fun main
intfun(void)
{
     printf(“hello world”);
     return0;
}
```

60) Program to check number is whether EVEN or ODD without using any arithmetic and relationaloperators.

HINT: Check for 0th bit

```
#include <stdio.h>

int main()
{
     int number=15;

     (number & 0x01) ? printf(“%d is an EVEN Number.”, number) :  printf(“%d is an ODD
Number.”,number) ;

     printf(“\n”;)
     return 0;
}
```