B.Tech I Year

Regular Course Handbook

Subject Name: Prog. for Problem Solving (Unit-4)

BCS101 / BCS201: PROGRAMMING FOR PROBLEM SOLVING

Content	Contact Hours
Unit -1:	8
Introduction to Components of a Computer System: Memory, Processor, I/O Devices, Storage, Operating System, Concept of Assembler, Compiler, Interpreter, Loader and Linker. **Idea of Algorithm:** Representation of Algorithm, Flowchart, Pseudo Code with Examples, From Algorithms to Programs, Source Code. **Programming Basics:** Structure of C Program, Writing and Executing the First C Program, Syntax and Logical Errors in Compilation, Object and Executable Code. Components of C Language. Standard I/O in C , Fundamental Data types, Variables and Memory Locations, Storage Classes.	
Unit-2:	8
Arithmetic Expressions and Precedence : Operators and Expression Using Numeric and Relational Operators, Mixed Operands, Type Conversion, Logical Operators, Bit Operations, Assignment Operator, Operator precedence and Associatively. **Conditional Branching:** Applying if and Switch Statements, Nesting if and Else and Switch.	
Unit-3:	8
Iteration and Loops: Use of While, do While and for Loops, Multiple Loop Variables, Use of Break , Goto and Continue Statements. **Arrays:** Array Notation and Representation, Manipulating Array Elements, using Multi Dimensional Arrays. Character Arrays and Strings, Structure, union, Enumerated Data types, Array of Structures, Passing Arrays to Functions.	
Unit-4:	8
Functions: Introduction, Types of Functions, Functions with Array, Passing Parameters to Functions, Call by Value, Call by Reference, Recursive Functions. **Basic of searching and Sorting Algorithms:** Searching & Sorting Algorithms (Linear Search , Binary search , Bubble Sort, Insertion and Selection Sort)	
Unit-5:	8
Pointers: Introduction, Declaration, Applications, Introduction to Dynamic Memory Allocation (Malloc, Calloc, Realloc, Free), String and String functions , Use of Pointers in Self-Referential Structures, Notion of Linked List (No Implementation) **File Handling:** File I/O Functions, Standard C Preprocessors, Defining and Calling Macros and Command-Line Arguments.	

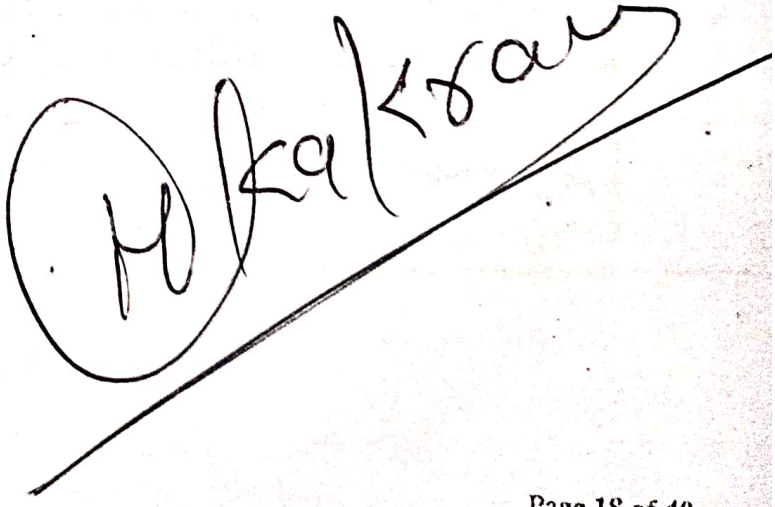
Course Outcome:

Course Outcome (CO)		Bloom's Level
At the End of Course , the Student will be Able to Understand		
CO 1	To Develop Simple Algorithms for Arithmetic and Logical Problems.	K_2, K_3
CO 2	To Translate the Algorithms to Programs & Execution (in C Language).	K_3
CO 3	To Implement Conditional Branching, Iteration and Recursion.	K_3
CO 4	To Decompose a Problem into Functions and Synthesize a Complete Program Using Divide and Conquer Approach.	K_4
CO 5	To Use Arrays, Pointers and Structures to Develop Algorithms and Programs.	K_2, K_3

K_1- Remember, K_2- Understand, K_3- Apply, K_4- Analyze , K_5- Evaluate , K_6- Create

Text Books:

1. Schaum's Outline of Programming with C by Byron Gottfried , McGraw-Hill
2. The C programming by Kernighan Brain W. and Ritchie Dennis M., Pearson Education .
3. Computer Basics and C Programming by V.Rajaraman , PHI Learning Pvt. Limited, 2015.
4. Computer Concepts and Programming in C, E Balaguruswami, McGraw Hill
5. Computer Science- A Structured Programming Approach Using C, by Behrouz A. Forouzan, Richard F. Gilberg, Thomson, Third Edition , Cengage Learning - 2007.
6. Let Us C By Yashwant P. Kanetkar.
7. Problem Solving and Program Design in C, by Jeri R. Hanly, Elliot B. Koffman, Pearson Addison-Wesley, 2006.
8. Programming in C by Kochan Stephen G. Pearson Education – 2015.
9. Computer Concepts and Programming in C by D.S. Yadav and Rajeev Khanna, New Age International Publication.
10. Computer Concepts and Programming by Anami, Angadi and Manvi, PHI Publication
11. Computer Concepts and Programming in C by Vikas Gupta, Wiley India Publication
12. Computer Fundamentals and Programming in C. Reema Thareja, Oxford Publication



B.Tech First Year: Regular Course Lecture Plan Session 2022-23

Subject Name	Programming for Problem Solving

Unit No.	Unit Name	Syllabus Topics	Lecture No
1	Introduction to Programming: Introduction to components of a computer system:	Memory, processor, I/O Devices, storage	1
		Operating system and it's type, Introduction to low level & high level languages	2
		IDE role for development , Concept of assembler, compiler, Interpreter, loader and linker	3
	Idea of Algorithm:	Representation of Algorithm, Flowchart	4
		Pseudo code with examples	5
	Programming Basics:	Structure of C program: writing and executing the first C program, types of errors	6
		Components of C language: Standard I/O in C	7
		Fundamental data types, Variables and memory locations	8
2	Arithmetic expressions & Conditional Branching: Arithmetic expressions and precedence:	Operators and expression using numeric and relational operators, mixed operands	9
		Logical operators, bit operations, assignment operator	10
		Operator precedence and associatively, Implicit and explicit type conversion	11
	Conditional Branching:	Applying if	12
		Nesting if else	13
		Else if ladder	14
		Switch statements, use of break and default with switch.	15
3	Loops & Functions: Iteration and loops:	Use of while and for loops	16
		Concept of do while loop, break and continue	17
		Nested Loop and multiple loop variables	18
	Functions:	Introduction to function and types of functions	19
		Call by value and call by reference	20
		Storage Class	21
		Recursive functions	22

B.Tech First Year: Regular Course Lecture Plan Session 2022-23

Subject Name	Programming for Problem Solving

Unit No.	Unit Name	Syllabus Topics	Lecture No
4	Arrays & Basic Algorithms: Arrays:	Array notation and representation, manipulating array elements	23
		Using multi dimensional arrays	24
		Functions with array Passing arrays to functions	25
		Character arrays and strings	26
		Structure & Union	27
		Enumerated data types	28
	Basic Algorithms:	Searching : Linear Search	29
		Binary Search	30
		Basic Sorting Algorithms : Bubble Sort	31
		Selection Sort	32
		Insertion Sort, Notion of order of complexity	33
5	Pointer& File Handling: Pointers:	Introduction, declaration, applications of pointer	34
		Introduction to dynamic memory allocation:- malloc, calloc, realloc and free	35
		Linked List: Use of pointers in self-referential structures notion of linked list (no implementation)	36
	File handling:	File I/O functions	37
		File Programs	38
		Standard C preprocessors, Types of Preprocessor Directives	39
		Command-line arguments	40

Signature	
Name of Subject Head	Ms. Radhika Jindal

Unit → 4

Que1 What is an array ? Give an example ? (2016-17)

Ans An array is the collection of similar data type elements stored at contigous memory location.

Syntax:- data-type array-Name[array-size];

Ex!- int marks[5];

Que2 What are subscript ? How they are specified ? (2017-18)

Ans! Array subscript is an integer type constant or variable name whose value range from 0 to size-1, where size is the total number of element in the array.

Syntax:- Array-Name[subscript]

Que3 What is array ? An which situation array is advantageous over linked list ? (2018-19)

Ans An array is the collection of similar data type elements stored at contiguous memory location.

Syntax: data-type Array-Name[array-size]

Ex: int marks[5];

Advantage of array over linked list:-

1. Array allow random acess to elements contained within it, where as link list only allows sequential acess.
2. Array takes less storage than link list for storing same number of element.
3. Array we is better when the maximum no. of list that can be present are known.

Que 4. Write an algorithm to find second largest element in an array. (2019-20)

Ans
① Start
② Initialize variable first with value 0
③ Start traversing the array from array[1]
④ a) If the current element in the array, say array[i] is greater than first. Then update first and second as:-

```
second = first
first = array[i]
```

b) If the current element is in between first and second, then update second to store the value of current variable as:

```
second = array[i]
```

④ Return the value stored in second.
⑤ Stop.

Que 5. Write a program to find odd and even number from the array elements and its count. (2020-21)

Ans

```
#include <stdio.h>
int main()
{
  int a[100];
  int n, i, even=0, odd=0;
  printf("Enter no of elements\n");
  scanf("%d", &n);
  for(i=0; i<n; i++)
  {
```

```
printf("Enter %d element\n", i+1);
scanf("%d", &a[i]);
}
for(i=0; i<n; i++)
{
if (a[i]%2==0)
{
printf("%d is even\n", a[i]);
even++;
}
else
{
printf("%d is odd\n", a[i]);
odd++;
}
}
printf("No of odd numbers are %d\n", odd);
printf("No of even numbers are %d\n", even);
return 0;
}
```


Output:
```
Enter no of element
5
Enter 1 element
5
Enter 2 element
6
Enter 3 element
4
Enter 4 element
9
Enter 5 element
3
```

5 is odd
6 is even
4 is even
9 is odd
3 is odd

No of odd numbers are 3
No of even numbers are 2

Multidimensional array

Qu6- write a C program to check whether given square matrix is symmetric or not. (2015-16)

```
#include <stdio.h>
  int main()
   {
    int n, c, d, matrix[10][10], transpose[10][10];
    printf("Enter the number of rows & column of matrix\n");
    scanf("%d", &n);
   printf("Enter elements of the matrix\n");
    for(c=0; c<n; c++)
     {
      for(d=0; d<n; d++)
       {
        scanf("%d", &matrix[c][d]);
       }
     }
      for(c=0; c<n; c++)
         {
           for(d=0; d<n; d++)
            {
              transpose[d][c] = matrix[c][d];
            }
         }
```

```
for(c=0; c<n; c++)
  {
    for(d=0; d<n; d++)
     {
       if(matrix[c][d] = transpose[c][d]
        break;
     }
       if(d != n)
          break;
       if(c==n)
        printf("The matrix is symmetric\n");
       else
        printf("matrix is not symmetric\n");
       return 0;
     }
```

Output: Enter the number of rows & columns of matrix

2

Enter elements of the matrix

2

4

5

3

matrix isn't symmetric

imp

Q7 write a c program to multiply two matrices. Take the size of elements of matrices from the keyboard. (2015-16)

'or'

write a c program to multiply two matrices and display the resultant matrix (2016-17)

write a program to multiply two matrices read size and number of elements of matrices from the keyboard. (2017-18)

'or'

write a program to multiply 2 matrice of size mxm. (2015-16)

```
#include <stdio.h>
int main()
  {
    int a[10][10], b[10][10], c[10][10], i, j, k, m, p, q
    printf("enter the order of first matrix");
    scanf("%d%d",&m,&n);
    printf("enter the order of second matrix");
    scanf("%d%d",&p,&q);
    if(n!=p)
    {
     printf("multiplication not possible");
    }
  else
    {
     printf("enter elements of first matrix\n");
      for(i=0;i<m;i++)
      {
        for(j=0;j<n;j++)
      {
    scanf("%d",&a[i][j]);
    }
  }
printf("enter element of second matrix\n");
for(i=0;i<p;i++)
  {
```

```
for(j=0; j<q; j++)
  {
   scanf("%d", &b[i][j]);
  }
}
for(i=0; i<m; i++)
  {
   for(j=0; j<q; j++)
     {
       c[i][j]=0
       for(k=0; k<n; k++)
         {
           c[i][j]= c[i][j]+ a[i][k] * b[k][j];
         }
     }
  }
printf("Product of matrices:\n");
for(i=0; i<m; i++)
{
 for(j=0; j<q; j++)
 {
  printf("%d\t", c[i][j]);
 }
 printf("\n");
}
return 0;
}
```

output: Enter the order of first matrix 3 3
Enter the order of second matrix 3 3
Enter element of first matrix
1 2 0 0 1 1 2 0 1
enter element of second matrix
1 1 2 2 1 1 1 2 1
Product of matrices:
5 3 4
3 3 2
3 4 5

Q. Write a program to multiply two matrices of dimension 3×3 and store the result in another matrix (write comments also at appropriate places) (2018-19)

Soln

```
#include<stdio.h>
int main()
{
  a[3][3], b[3][3], c[3][3], i, j
  printf("Enter elements of first matrix \n");
  for( i=0; i<3; i++)          // Take first matrix from user
  {
    for(j=0; j<3; j++)
    {
      scanf("%d", &a[i][j])
    }
  }
printf("enter elements of second matrix\n")
for(i=0; i<3; i++)             // Take second matrix from user
{
  for(j=0; j<3; j++)
  {
```

```
scanf ("%d", &b[i][j]);
 }
}
for(i=0; i<3; i++)      /* Calculating product of matrices*/
{
 for(j=0; j<3; j++)
 {
  c[i][j]=0
  for(k=0; k<3; k++)
  {
   c[i][j] = c[i][j] + a[i][k] * b[k][j];
  }
 }
printf("output matrix:\n"); //Printing output matrix
 for(i=0; i<3; i++)
 {
  for(j=0; j<3; j++)
  {
   printf("%d", c[i][j]);
  }
  printf("\n");
 }
```

Output

```
Enter element of first matrix
1 2 0 0 1 1 2 0 1
enter element of second matrix
1 1 2 2 1 1 1 2 1
output matrix:
5 3 4
3 3 2
3 4 5
```

V.imp

Q.9. write a program in C for matrix addition. (2016-17)

Sol^n

```
#include <stdio.h>
int main()
 {
 int a[10][10], b[10][10], sum[10][10], i, j, m, n
printf("Enter rows and column of matrix");
scanf("%d%d", &m, &n);
printf("\n enter first matrix");
      for(i=0; i<m; i++)
       {
        for(j=0; j<n; j++)
        {
          scanf("%d", &a[i][j]);
        }
       }
printf("\n enter second matrix");
     for(i=0; i<m; i++)
      {
        for(j=0; j<n; j++)
        {
          scanf("%d", &b[i][j]);
        }
      }
printf("output matrix: \n");
 for(i=0; i<m; i++)
  {
    for(j=0; j<n; j++)
    {sum[i][j] = a[i][j] + b[i][j];
     printf("%d", sum[i][j]);
    }
      printf("\n");
  }
```

Output:- Enter rows & columns of matrix 2 2
Enter first matrix 2 2 2 2
Enter second matrix 3 3 3 3
Output matrix :
5 5
5 5

Q10 Write a program in C to find the largest number of a 4 * 4 matrix. (2017-18, 2016-17)

```
#include<stdio.h>
int main()
{
int a[4][4], largest, i, j
printf("Enter elements of 4*4 matrix");
for(i=0; i<4; i++)
   {
     for(j=0; j<4; j++)
     {
       scanf("%d", &a[i][j]);
     }
   }
largest=a[0][0];
for(i=0; i<4; i++)
 {
   for(j=0; j<4; j++)
   { if(a[i][j] > largest
       largest = a[i][j];
   }
 }
printf("\n largest element is %d", largest);
return 0;
}
```

Output:- Enter element of 4*4 matrix 1 2 3 4 5 1 2 3 4 5 1 2 3
1126
largest element is 6.

Q11. Write a program to add two matrices of dimension 3x3 and store the result in another matrix. (2017-18)

Soln

```
#include <stdio.h>
int main()
{
int a[3][3], b[3][3], sum[3][3], i, j;
printf("enter elements of first matrix\n");
for(i=0; i<3; i++)
{
  for(j=0; j<3; j++)
  {
    scanf("%d", &a[i][j])
  }
}
printf("enter elements of second matrix\n");
for(i=0; i<3; i++)
  {
    for(j=0; j<3; j++)
      {scanf("%d", &b[i][j]);
  }
}
printf("output matrix is:\n");
for(i=0; i<3; i++)
  {
    for(j=0; j<3; j++)
    {
      sum[i][j] = a[i][j] + b[i][j];
```

```
        printf ("%d \t", sum[i][j]);
      }
    }
```

output:- Enter elements of first matrix
1 1 1 2 2 2 3 3 3
Enter elements of second matrix
1 1 1 1 1 1 1 1 1
output matrix is :

2 2 2
3 3 3
4 4 4

Q.12 What are advantages of using array? (20-21)
'or' write a program for matrix multiplication of two matrix elements. (refer Que-7 for solution)

Ans ① In an array, accessing an element is very easy by using the index number.

② The search process can be applied on any array very easily.

③ Matrix can be represented as 2D array.

④ If a user wishes to store multiple value of similar data type then array can be use.

Q Explain passing array to functions in detail;

Ans Passing array to function:- There are various problems in which we wish to pass more than one variable to a function - Ex:- In a function which sorts the 10

elements in attending order. To such function we can pass an array of 10 elements instead of passing 10 variable as actual parameter. There are different ways to pass the array as argument to the function.

① Passing single element of array to function: We can pass single element of an array as actual argument to the function.

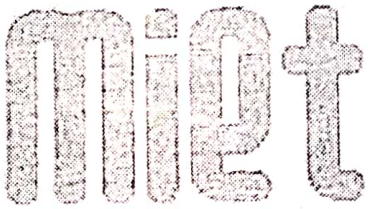
Ex:-
```
#include<stdio.h>
void display(int age)
{
    printf("%d", age);
}
int main()
{
    int ageArray[] = {2, 3, 4};
    display(ageArray[2]);
    return 0;
}
```

Output:- 4

② Passing an entire array to function: An entire array can be passed to function as argument.

Ex:
```
#include<stdio.h>
float average(float age[]);
int main()
{
    float avg, age[] = {23.4, 55, 22.6, 3, 40.5, 18};
    avg = average(age);
```

```
printf("Average age = %f", avg);
return 0;
}
float average( float age[])
{
int i;
float avg, sum = 0.0;
for(i=0; i<6; i++)
{
sum += age[i];
}
avg = (sum/6);
return avg;
}
```



Output:-

Average age = 27.080000

③ Passing multidimensional array to function :- we can also pass multidimensional array to function.

ex:

```
#include <stdio.h>
void displayNumbers(int num[2][2]);
int main()
{
int num[2][2], i, j;
printf("enter 4 numbers : \n");
for(i=0; i<2; i++)
for(j=0; j<2; j++)
scanf("%d", &num[i][j]);
```

```
    displayNumbers(num);
    return 0;
}
void displayNumbers(int num[2][2])
{
    int i, j;
    printf("Displaying:\n");
    for(i=0; i<2; i++)
    for(j=0; j<2; j++)
        printf("%d\t", num[i][j]);
}
```

Output: Enter 4 numbers: 2 3 4 5
Displaying: 2 3 4 5

Que13 What is string? (2015-16)

Ans A string is a sequence of character terminated with a null character '\0'

Ex:- char C[] = "Hello world";

When the compiler encounters a sequence of character enclosed in the double quotation marks, it appends a null character at the end by default.

Que14 Explain in brief the purpose of 'strcmp' function. (2015-16)

Ans This strcmp() function is used to compare two strings. The strcmp() function returns 0 if both the strings are same.

Syntax:- int i = strcmp(first_string, second_string);

Ex:- char first [ ] = "Hello" ;
char second [ ] = "Hello" ;
int i = strcmp (first , second);
⇒ i° = 0

Q15- write a program to reverse a string by using pointer. (2017-18)

Ans
```
#include <stdio.h>
# include <conio.h>
  void strrev1 (char*) ;
void main()
 {
  char name[100] ;
  int i ;
  clrscr() ;
  printf ( " Enter a string \n" ) ;
   gets (name) ;
  strrev1(name) ;
  printf ( " After reverse string is : \n" ) ;
   puts ( name ) ;
   getch() ;
 }
void strrev1 ( char *s )
  { int i=0 ;
    char *j , t ;
      j = s ;
    while ( *j != '\0' )
     { i++ ;
       j++ ;
     } j-- ;
```

```
        i = i/2;
        while(i>0)
        {
          t= *s;
          *s = *d;
          *d = t;
          s++;
          d--;
          i--;
        }
    }
```

Output:- Enter a string
RAM
After reverse string is:
MAR

V.imp

Qu16. Explain the significance of null character in string

Ans Null character is used to represent (2018-19)
the end of a string of character.

Ex: char s[] = "Test" can be written as
char s[] = {'T', 'e', 's', 't', '\0'};

String library function:

① Strlen() :- It counts number of character present in string.

```
Ex:- #include <stdio.h>
     #include <conio.h>
     #include <string.h>
     void main()
     {
       char s[10];
```

```
int l ;
clrscr ( ) ;
printf ( "enter string") ;
gets (s) ;
l = strlen (s) ;
printf ( "string length = %d ", l) ;
getch ( ) ;
}
```

Output:- Enter string RAM
String length = 3

② strcpy (s2, s1) :- It copies the content of string s1 into string s2.

```
Ex:- # include <stdio.h>
     # include <conio.h>
     # include <string.h>
       void main ( )
       {
        char s1 [10], s2 [10] ;
        clrscr ( ) ;
        printf ( "enter string") ;
         gets (s1) ;
        strcpy (s2, s1) ;
       printf ( "\n string copied is \n") ;
       puts (s2) ;
        getch ( )
       }
```

③ strcat (s2, s1) :- It join the content of string s1 with string s2.

Ex:-
```
#include <stdio.h>
#include <conio.h>
#include <string.h>
void main()
{
 char s1[10], s2[10];
 clrscr();
 printf("enter string 1 \n");
  gets(s1);
 printf("\n enter string 2 \n");
  gets(s2);
 strcat(s2, s1);
 printf("\n string after join \n");
 puts(s2);
 getch();
}
```

Output:-
```
Enter string 1
RAM
Enter string 2
RAJ
RAJRAM
```

⑨ strrev(s):- This function is use to reverse the string.

Ex:-
```
#include <stdio.h>
#include <conio.h>
#include <string.h>
void main()
{
 char s[10];
 clrscr();
```

```
printf("Enter string");
  gets(s);
 strrev(s);
printf("string reverse is \n");
 puts(s);
  getch();
}
```

Output:- Enter string
ABC
String reverse is
CBA

V. imp

Ques Differentiate between structure and union. (2015-16, 2018-19, 2019-20, 2020-21)

Ans

Structure	Union
① Keyword struct is use to define a structure.	① Keyword union is use to define a union.
② Every member within structure is assigned a unique memory location.	② Memory location is shared by all data members.
③ Changing the value of one data member will not affect other data member in structure.	③ Changing the value of one data member will change the value of other data member of union.
④ The total size of the structure is the sum of the size of every data member.	④ The total size of the union is the size of the largest data member.
⑤ We can retrieve any member at a time	⑤ You can access one member at a time in the union.

⑥ Syntax:

```
struct student{
        char name[20];
        int age;
        char address[50];
} std;
```

⑥ Syntax:

```
union student{
        char name[20];
        int age;
        char address[50]
} u;
```

Q.18 Describe structure. Differentiate between structure and array. Define structure data type called time_structure containing three members integer hour, integer minute and integer second.

or

write a program in c that would assign value to the individual members and display the time in the following form: 16:40:52

Ans A structure is a user defined data type which contains number of similar and dissimilar data types group together.

Syntax

```
struct [structure tag]
{
 member defination;
 member defination;
  =
  =
 member defination;
}
[one or more structure
        variable]
```

Example

```
struct books
{
 char title[50];
 char author[50];
 char subject[50];
 int book_id;
} book;
```

Que: Explain structure with suitable example (2015-16)
Que: Difference between structure & array (2015-16, 2016-17)

Ans

Array	Structure
① Array is the collection of same data type elements	① Structure is the collection of variable of different data type.
② Syntax:- type array-Name[size]	② Syntax: struct struct-name { type element 1; type element 2; ! ! } structure-variable;
③ Array elements are stored in contigyous memory location.	③ Structure element may not be stored in a contigyous memory location.
④ Array elements are accessed by their index number.	④ Structure elements are accessed by their names.
⑤ Array declaration and elements accessing operator is "[]" (square) bracket.	⑤ Structure element accessing operator is "." (Dot operator)
⑥ Every element in array is of same size.	⑥ Every element in a structure is of different data type.
⑦ No keyword is to declare array.	⑦ Struct keyword is used

```
#include <stdio.h>
  struct time_struct
  {
    int hour;
    int minute;
    int second;
  } t;
  int main()
   {
    printf("\n Enter hour:");
    scanf("%d", &t.hour);
    printf("\n Enter minute:");
    scanf("%d", &t.minute);
    printf("\n Enter second:");
    scanf("%d", &t.second);
  printf("%d:%d:%d", t.hour, t.minute, t.second);
   return 0;
  }
```

Output:-
```
Enter hour 16
Enter minute 40
Enter second 51
16:40:51
```

Que20- write a C program to store the employee details using structure. (2015-16)

Ans
```
#include <stdio.h>
  struct EMPL
{
  int employee_n[];
  char employee-name[20];
  char employee_edu[10];
  } emp[50];
int main()
    {
      int i;
printf("\n Enter details of 50 employees");
for(i=0; i<50; i++)
{
  scanf("%d %s %s", &emp[i].employee_no, emp[i].
        employee-name, emp[i], employee_edu);
  }
   for(i=0; ic50; i++)
  {
   printf("employee %d details are:\n", i+1);
   printf("EMP No = %d, Emp Name = %s, EMP Education=%s
        , emp[i].employee-no, emp[i].employee_
          name, emp[i].employee-edu);
    }
   return 0;
  }
```

output:- Enter details of 50 employees

20 Ram B.tech

30 Raj MCA

⋮

40 Suresh B.tech

Employee 1 details are:

20 Ram B.tech

Employee 2 details are:

30 Raj MCA

⋮

Employee 50 details are:

40 Suresh B.tech

Que21- Describe the relation between structure & pointer. (2015-16)

Ans — Pointer variable can be accessed along with structure.

A pointer variable of structure can be created as follows:

```
struct name {
        member1;
        member2;
        ⋮
        };
struct name *ptr;
```

Here, the pointer variable of type struct name is created.

Example to access structure member using pointer —

```
#include <stdio.h>
  struct name
      {
       int a;
       float b;
      };
int main()
    {
      struct name *ptr, b;
        ptr = &b
      printf("enter integer:");
      scanf("%d", &(*ptr).a);
      printf("enter number:");
      scanf("%f", &(*ptr).b);
      printf("Displaying:");
      printf("%d%d", (*ptr).a, (*ptr).b);
       return 0;
    }
```

In this example, the pointer variable of type struct name is referenced to the address of P. Then, only the structure member can also be accessed using → operator (*ptr).a is same as ptr→a.

Que22. Write a program in C to create a database of fifty students to store personal details such as roll No, name & marks, print all the details of students whose name is entered by the user. (2016-17, 2017-18)

Sol:

```
#include <string.h>
#include <stdio.h>
#include <conio.h>
  struct student
  {
    int roll_no;
    int marks;
    char name[20];
   } s[50];
  void main()
   {
     int i
     char c = 'y'
     char sname[20];
     clrscr();
   for(i=0; i<50; i++)
   {
     printf("Enter roll no, name & marks \n");
     scanf("%d %s %d", &s[i].roll_no, s[i].
           name, &s[i].marks);
   }
     while(c=='y')
     {
   printf("Enter name of students whose record
           you want to display:");
   scanf("%s", &sname);
   for(i=0; i<50; i++)
  { if(strcmp(s[i].name, sname)==0)
```

```
{
  printf("\n %d \t %s \t %d", s[i].roll_no,
          s[i].name, s[i].marks);
}
  printf("Enter y for continue");
  fflush(stdin);
  scanf("%c", &c);
  }
getch();
}
```

Output:- Enter roll no, name & marks

```
12  RAM   50
15  Shyam 70
 .
 .
20  Sita  81
```

Enter name of student whose record you want to display

RAM

```
12   RAM   50
```

Q. Write a C program to store the employee information using structure and search a particular employee using employee number.
Take the structure EMPL and its member employeeNo. employee Name, employee Edu.

Sol.

```
#include <stdio.h>
#include <string.h>
#include <conio.h>
```

```
struct EMPL {
  int employee No
  char employee Name [20];
  char employee Edu [10];
   }
     emp[50];
  void main()
   {
     int i, eno; char ch='y';
      for( i=0; i<50; i++)
     {
      printf("enter employee no, employee name and
                    employee education \n");
      printf("enter employee No");
      scanf("%d", &emp[i].employee No);
      printf("\n enter employee Name");
      scanf("%s", &emp[i].employee Name);
      printf("\n enter employee Education");
      scanf("%s", emp[i].employee Edu);
     }
    while( ch=='y')
     {
      printf("\n enter employee no you want to search");
       scanf("%d", &eno);
    for(i=0; i<50; i++)
   { if( eno==emp[i].employee No)
    {
      printf("\n %d \t %s", emp[i].employeeNo,
          emp[i].employee Name, emp[i].employee Edu);
```

```
    }
printf("Enter Y for continue");
    fflush(stdin);
    scanf("%c", &ch);
    }
 getch();
}
```

Output:- Enter employee no, employee name and employee education.

Enter employee No 12
Enter employee Name suresh
Enter employee Education B.Tech
!
!
Enter employer no, employee name and employee education
Enter employee No 14
Enter employee Name Ramesh
Enter employee Education MCA

Enter employee no you want to search
12

12 suresh B.Tech

Enter Y for continue
n

imp
Que:- Write a program that compares two given dates.

To store data we structure say date that contains three members namely date, month & year. If dates are equal then display message as "Equal" otherwise "Unequal"

Sol:

```
#include <stdio.h>
struct Date
        {
          int date
          int month
          int year
        } d1, d2;
int main()
    {
      printf("\n Enter first date (dd/mm/yyyy):");
      scanf("%d %d %d", &d1.date, &d1.month,
                        d1.year);
      printf("\n");
     printf("Enter second date (dd/mm/yyyy):");
     scanf("%d %d %d", &d2.day, &d2.month, &
                d2.year);
 if((d1.day==d2.day)&&(d1.month==d2.mont)&&
                (d1.year==d2.year))
  printf("Equal");
 else
   printf("unequal");
  return 0;
 }
```

Output:-

```
Enter first date (dd/mm/yyyy):
22 11 1996
enter second date (dd/mm/yyyy):
22 11 1996
Equal.
```

Qw25 Explain structure in C programming. Write a program in C using structure to enter and print the record of 10 books available in your library. Following fields may be included in the record : book_title, book_price and number of pages.

Ans
```
#include <stdio.h>
struct book {
        char book_title[50];
        float book_price;
        int number of pages;
    } b[10];
int main()
{
  int i
  for(i=0; i<10; i++)
  {
   printf("Enter title, price & no of pages of book
          %d ", i+1);
   scanf("%s %f %d", b[i].book_title, &b[i].book_
        price, &b[i].number_of_pages);
   printf("\n");
  }
  for( i=0; i<10; i++)
  {
    printf("\n Book %d details: %s %f %d",
     b[i].book_title, b[i].book_price, b[i].
     number_of_pages);
  }
```

```
return 0;
}
```

Output:- Enter title, price & no of pages of book 1

```
Book 1   100.0   100
!
Enter title, price & no. of pages of book 10
Book 10   900.0   120
Book 1 details : Book1   100.0   100
!
!
Book 10 details : Book 10   900.0   120
```

Ques6 Create a suitable structure in C language for keeping the records of the employees of an organization about their code, Name, Designation, salary, department, city of posting. Also write a program in C to enter the records of 100 employees and display the name of those who earn more than 20,000. (2019-20)

Soln

```
#include <stdio.h>
Struct emp
        {
          int code;
          char Name[20];
          char Designation[10];
          float salary;
          char Department[10];
          char city-of-posting[20];
        } e[100];
```

```
int main()
  {
   printf("Enter details of 100 employees:");
    for(i=0; i<100; i++)
   {
    scanf("%d %s %s %f %s %s", &e[i].code,
          e[i].Name, e[i].Designation, &e[i].salary,
          e[i].Department, e[i].city-of-posting);
   }
  printf("Name of employees whose salary is above 20k");
   for(i=0; i<100; i++)
   {
    if(e[i].salary > 20000)
      printf("\n%s", e[i].name);
   }
  return 0;
}
```

Output:- Enter detail of 100 employees:

1 RAM AP 10000 MGMT DELHI
2 RAJ AP 21000 FINANCE MUMBAI
⋮

Name of employees whose salry is above 20k

RAJ

Q7 Write short notes on ① union
imp ② Enumerated data types.
(2016-17, 2018-19).

Ans Union: - A union is a special data type available in C that enables us to store different data types in the same memory location.

We can define a union with many members, but only one member can contain a value at a time. Union provide an efficient way of using the same memory location for multiple purpose.

Syntax:-

```
union union_name
{
 union_member 1
 union_member 2
}
union_variable;
```

Example:-

```
union Data {
    int i;
    float f;
    char str[20];
} data;
```

Enumerated data type: A enumeration is a user defined data type consists of integral constants and each integer constant is given a name. Keyword enum is use to define enumerated data type.

Syntax:- enum type_name { value 1, value 2, - - -value N};

Example:-

```
enum week{sun, mon, tue, wed, Thu, fri, sat};
int main()
{
  enum week today;
   today = wed;
  printf("%d day", today+1);
  return 0;
}
```

Output:- '4 day'

Que 28: Explain linear search and binary search technique for searching an item in a given array. Also write the complexity for each searching algorithm. (2018-19)

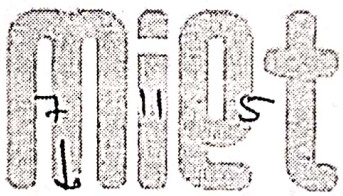
Ans ① Linear search: In this method each element of the list checked in a sequential manner until desired element is found.

Example:- Suppose a list is given as and we want to

23 25 7 11 5 9 2

search element 7 in the list.

① First we compare 23 with 7 then 25 then with 7, here match has found therefore this process return either message element found or element found at location 3.



23 25 7 11 5 9 12

↓ ↓

7 7 7

(not matched) (not matched) (matched)

(return location where element found)

Complexity:- O(n), where n is the number of element in which we are searching for a particular element.

② Binary search: The basic requirement for the binary search algorithm to be applied is that input array must be arranged in either ascending or descending order.

In this technique the approximate middle entry of the array is located, and its key is examined. If key is greater than the value we want to search then in next iteration we discard the second half of the array & then we apply same procedure on first half.

If key is less than the value we want to search then in next iteration we discard 1st half of the array & then we apply same procedure on second half.

If key is equal to value we want to search for then return the index/ location where found element.

	0	1	2	3	4	5	6	
List	2	5	7	9	12	15	19	— index

middle entry

search for 15 (9 is element at middle index)

∴ discard array portion from index 0 to 3 & apply same procedure to 2nd half of array.

4	5	6
12	15	19

↓ middle entry

Here element present at middle index = 15 and also we want to search for element 15, ∴ return index of this element + 1 (6 in this case) as location of element we want to search.

Complexity :- O(log n)

Qu29. Differentiate between linear search and binary search. (2018-19)

Ans

Basics	Linear search	Binary search
Time complexity	O(n)	O($\log_2 n$)
Prerequisite for an array	Not required	Array must be in sorted order.
Best case time	First element O(1)	Center element O(1)
Worst case for n number of elements	n comparisions are required	Can conclude after only $\log_2 n$ comparision.
Can be implemented on	Array & linked list	Can't be directly implement on linked list
Insert operation	New element can be easily inserted at the end of the list	Processing is required to insert element at its proper place in sorted list
Usefulness	Easy to use	Tricky algorithm
Lines of code	Less	More

imp.
Que30- WAP for Linear Search.

```
#include <stdio.h>
#include <conio.h>
void main()
{
int a[30], i, e, n;
printf("enter no. of elements");
scanf("%d", &n);
printf("enter array elements\n");
for(i=0; i<n; i++)
scanf("%d", &a[i]);
printf("enter element to be searched\n");
scanf("%d", &e);
for(i=0; i<n; i++)
  {
   if(a[i]==e)
     {
      printf("%d is present at loc=%d", e, i+1);
      break;
     }
  }
if(i==n)
 printf("%d is not present", e);
}
```

imp.

Que:-31 WAP for Binary Search.

```
#include <stdio.h>
#include <conio.h>
void main()
{
 int a[30], n, i, l, h, mid, e;
 printf("enter no. of elements");
 scanf("%d", &n);
 printf("enter array elements in sorted order");
 for(i=0; i<n; i++)
 scanf("%d", &a[i]);
 printf("enter search element");
 scanf("%d", &e);
 l = 0;
 h = n-1;
 while(l<=h)
  {
    mid = (l+h)/2;
    if(e == a[mid])
        {
          printf("%d is present at %d", e, mid+1);
          break;
        }
```

```
else if ( e > a [mid])
    l = mid + 1;
else
    h = mid - 1;
}
if (l > h)
printf ("%d is not present ", e);
getch();
}
```

Output :-

```
enter no. of elements
5
enter array element
2
3
6
10
15
enter search element
3
3 is present at 2
```

Q1) What do you mean by sorting? write program to sort the given n positive integers. Also give the flowchart for the same. (2015-16, 2016-17 (I sem))

'or'

Write a program in C for sorting N positive integers in ascending order (2016-17 (II sem))

'or'

Write a C program to sort set of integers in ascending order by using bubble sort technique. (2017-18)

'or'

Write the importance of sorting in problem solving write a program in C using bubble sort technique to sort 10 numbers entered by the user. (2018-19)

'or'

Ans What do you mean by sorting? write a program in C to sort 'n' positive integers using bubble sort. Also draw the flowchart for the same. (2019-20)

or

Define sorting algorithm with example. (2020-21)

Ans **Sorting**:- It is the process of arranging data in some given sequence or order (in increasing) or decreasing order.

Ex:- Suppose we have an array of 10 elements.

10, 3, 6, 12, 4, 17, 5, 9, 25, 24

After sorting in ascending order values will be

3 4 5 6 9 10 12 17 24 25

Sorting algorithm:- ① Bubble sort
② selection sort
③ Insertion sort

imp.

Program:-

```
# Include <stdio.h>
# Include <conio.h>
  void main()
  {
    int i, j, t, a[25], n;
    printf("How many elements");
    scanf("%d", &n);
    printf("enter %d elements:", n);
     for(i=0; i<n; i++)
       {
       scanf("%d", &a[i]);
       }
  for(i=0; i<n-1; i++)
     {
      for(j=0; j<n-1-i; j++)
      {
        if(a[j] > a[j+1])
        {
          t=a[j];
          a[j]=a[j+1];
          a[j+1]=t;
          }
      }
  }
```

```
printf("\n sorted array is \n");
for(i=0 ; i<n ; i++)
    {
      printf("%d \t", a[i]);
      }
    getch();
  }
```

output:- How many elements 10

Enter 10 elements

10 5 6 9 12 1 7 13 19 16

Sorted array

1 5 6 7 9 10 12 13 16 19

Example :- If n=4 & numbers are 4 3 2 1

i=0

4	3	2	1	compare a[0], a[1]
3	4	2	1	compare a[1], a[2]
3	2	4	1	compare a[2], a[3]
3	2	1	4	3 compare (n-1-i)

i=1

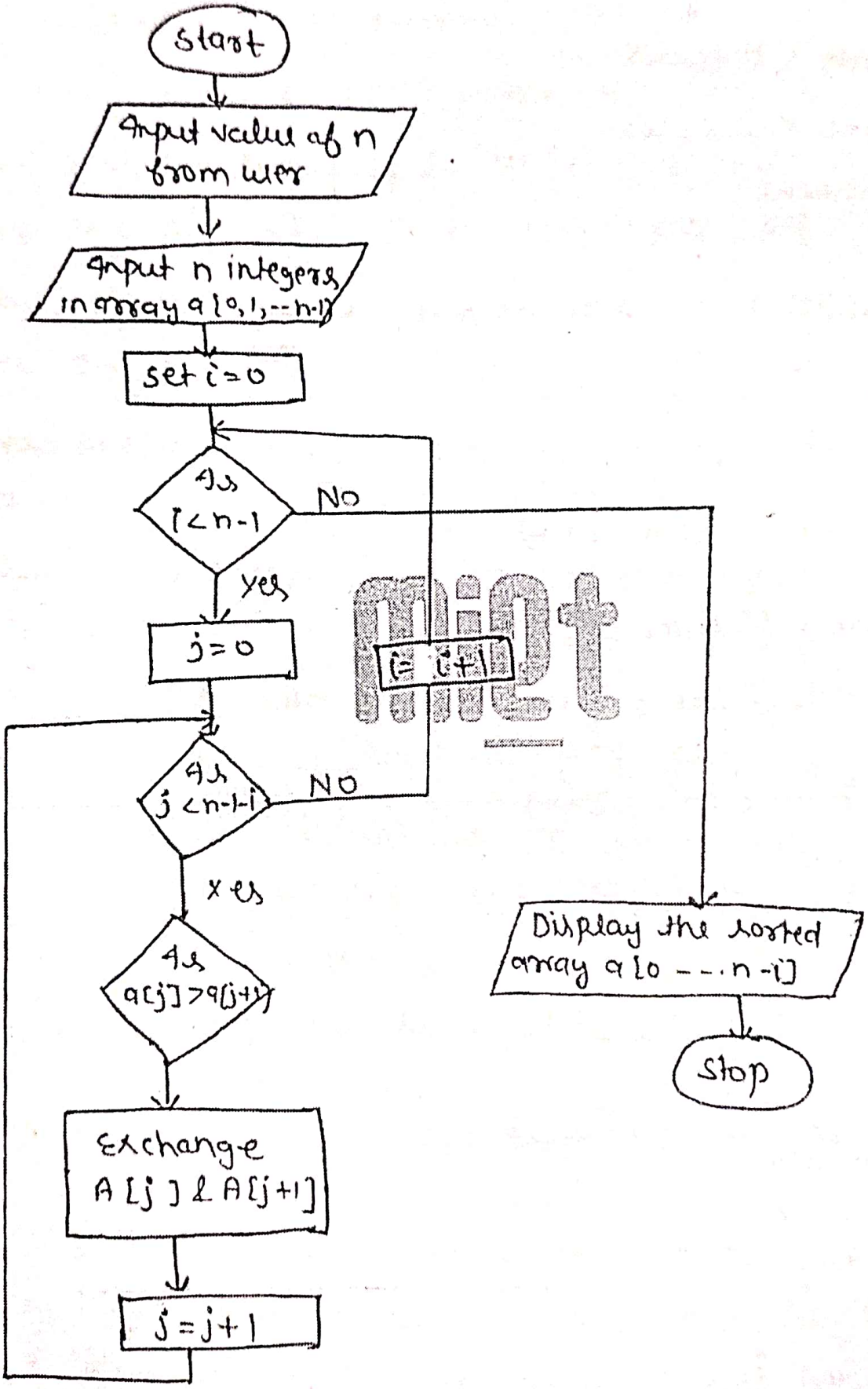
3	2	1	4	compare a[0], a[1]
2	3	1	4	compare a[1], a[2]
2	1	3	4	2compare (n-1-i)

2	1	3	4	compare a[0], a[1]
1	2	3	4	1compare (n-1-i)

For each pass we need n-1-i comparision

complexity:- $O(n^2)$

Flowchart Bubble sort :-



Que with :- WAP for selection Sort..

```
#include <stdio.h>
#include <conio.h>
void main()
{
int a[25], i, min, n, p, t, j;
printf ("No: of elements");
scanf ("%d", &n);
for (i=0; i<n; i++)
  {
    printf ("enter values");
    scanf ("%d", &a[i]);
  }
for (i=0; i<n-1; i++)
  {
    min = a[i];
     p = i;
   for (j=i+1; j<n; j++)
     {
       if (a[j] < min)
         {
           min = a[j];
            p = j;
         }
     }
```

```
    t = a[i];
    a[i] = a[p];
    a[p] = t;
  }
printf ("Sorted array");
for (i=0; i<n; i++)
  printf ("%d", a[i]);
getch();
}
```

Output

```
No. of elements
5
enter value
5 2 7 3 1
Sorted array
1 2 3 5 7
```

Q. Explain selection sort technique for sorting problem. Write an algorithm for selection sort. Sort the following number using selection sort technique. (2018-19)

'or'

Explain selection sort technique with example. (2020-21)

Ans Selection sort is a simple sorting algorithm. In this the list is divided into two parts, the sorted part at the left and unsorted part at the right side. Initially sorted part is empty & unsorted part is entire list. The smallest element is selected from the unsorted array & swapped with the leftmost element, and that element becomes a part of sorted list. The process continues moving unsorted array boundary by one element by one element to the right.

Time complexity = $O(n^2)$

Algorithm:
1. Start
2. Repeat step 3,4,5,6 for i = 0 to n - 2
3. Set min = a[i] and loc = i
4. Repeat step 5 for j = i+1 to n - 1
5. If min > a[j] then min = a[j] and loc = j
6. Interchange a[i] and a[loc] by
   temp = a[i], a[i] = a[loc] and a[loc] = temp
7. Stop.

Q Sort the following number using selection sort.

(26) 54 93 (17) 77 31 44 55 20

Smallest is 17 so exchange with 26

17 (54) 93 26 77 31 44 55 (20)

Now smallest element is 20 so exchange with 54

17 20 (93) (26) 77 31 44 55 54

Now smallest element is 26 so exchange with 93

17 20 26 (93) 77 (31) 44 55 54

Now smallest elment is 31 so exchange with 93.

17 20 26 31 (77) 93 (44) 55 54

Now smallest element is 44 so exchange with 44

17 20 26 31 44 (93) 77 55 (54)

Now smallest element is 54 so exchange with 93

17 20 26 31 44 54 (77) (55) 93

Now smallest element is 55 so exchange 55 & 77

17 20 26 31 44 54 (77) 93

Now 77 is smallest and its already in proper place

17 20 26 31 44 54 55 77 93

Ques35 What is insertion sort. Explain it with algorithm & example?

Ans — This algorithm places an unsorted element at its suitable place in each iteration. Insertion sort works similarly as we sort cards in our hand in a card game. We assume that first card is sorted then

we select unsorted card. If unsorted card is greater than the card in hand, it is placed on the right. otherwise to the left. In the same way, other unsorted cards are taken and put in their right place.

Similar approach is used by insertion sort.

Algorithm:- ① Start

② Iterate from arr[1] to arr[n] over the array.

③ Compare the current element (key) to its predecessor.

④ If the key element is smaller than its predecessor, compare it to the element before. Move the greater elements one position up to make space for the swapped element.

Example:- 12 11 13 5 6

i = 1 to 4

Key = 11 and key < 12 move 12 & insert 11 before 12

11 12 13 5 6

Now key is 13 and 13 > 12. so no need to move element

11 12 13 5 6

Now key is 5 and 5 < 13 & 5 < 12 & 5 < 11 so move 13, 12, 11 one place right and place 5 at first place (index 0)

5 11 12 13 6

Now key is 6 and 13 > 6 & 12 > 6 & 11 > 6

Therefore move 13, 12 & 11 one place toward right and place 6 at position 2 in array (index 1)

imp

Program (Insertion sort)

```
# include <stdio.h>
# include <conio.h>
  void main()
  {
    int i, j, a[25], n, key
    printf("how many elements:");
    scanf("%d", &n);
    printf("enter values:");
     for(i=0; i<n; i++)
      {
        scanf("%d", &a[i]);
      }
     for(i=1; i<n; i++)
     {
       key=a[i];
     for(j=i-1; j>=0; j--)
        {
          if(a[j]>key)
        {
          a[j+1]=a[j];
        }
          else
              break;
        }
     a[j+1]=key
  }
  printf("\n sorted array is \n");
    for(i=0; i<n; i++)
    {
      printf("%d",a[i]);
    }
```

```
getch();
}
```

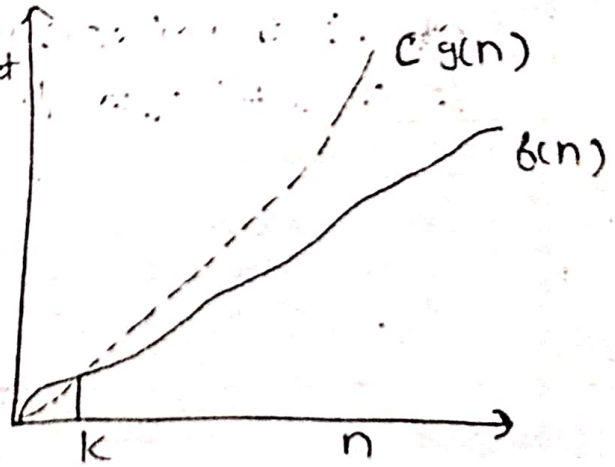
output:- how many elements:
6
Enter values:
5 6 3 2 1 4
sorted array is
1 2 3 4 5 6

imp.
Q.3 What do you mean by order of complexity? Explain various notation to represent order of complexity with diagram. (2018-19)

Ans Order of complexity of an algorithm is a way of predicting / saying how execution time of a program and space / memory occupied by it's change with in change in input size.

Asymptotic notations are mathematical tools to represent time complexity of algorithm for asymptotic analysis.

• Big Oh 'Notation':



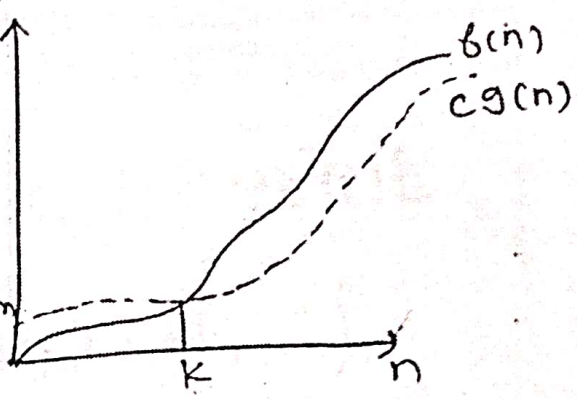
Big oh(o) notation is formal way to expess the upper bound of an algorithm running time. It measure the worst case time complexity or the longest amount of time an algorithm can possibly take to complete.

Ex:- $O(g(n)) = \{ f(n)$

There exist positive constants c and k such that

$0 <= f(n) <= c * g(n)$ for all $n >= k \}$

② Omega Notation:- The omega notation is the formal way to express lower bound of an algorithm's running time. It measures the best case time complexity or the best amount of time an algorithm can possibly take to complete



Ex:- for a function $f(n)$

$\Omega(g(n)) = \{ f(n)$: there exists positive constant c and k such that

$0 <= c * g(n) <= f(n)$ for all $n >= k \}$

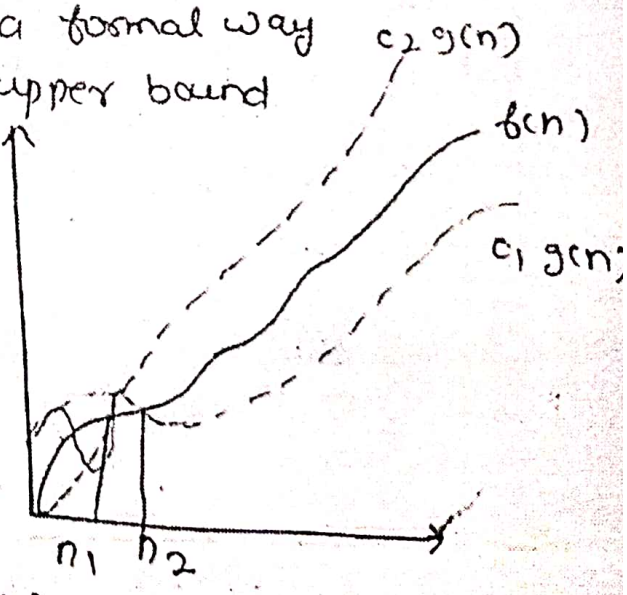
③ Theta Notation, Θ :- This is a formal way to express both the lower & upper bound of an algorithms running time. It is represented as follows:

Ex:- $\Theta(g(n)) = \{ f(n)$.

There exists positive constant c_1, c_2, k such that

$0 <= c_1 * g(n) <= f(n) <= c_2 * g(n)$

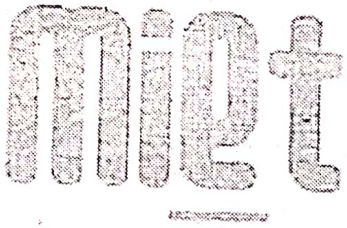
for all $n >= k \}$



5 Year's

University Paper Questions

(AKTU Question Bank)



5 Years AKTU University Examination Questions		Unit-4	
S. No	Questions	Session	Lecture No
1	What is an array? Give an example.	16-17 (Even)	23-33
2	What are subscripts? How they are specified?	17-18 (Odd)	23-33
3	What is array? In which situation array is advantageous over linked list?	18-19 (Odd)	23-33
4	Write an algorithm to find second largest element in an array.	19-20 (Odd)	23-33
5	Write a program to find odd and even number from the array elements and its count.	20-21 (Odd)	23-33
6	Write a C program to check whether given square matrix is symmetric or not .	15-16 (Odd)	23-33
7	Write a C program to multiply two matrices. Take the size of elements of matrices from the keyboard.	15-16 (Odd)	23-33
8	Write a program to multiply the 2 matrices of size M*N	15-16 (Even)	23-33
9	Write a program in C for matrix addition.	16-17 (Odd)	23-33
10	Write a program to multiply two matrices of dimension 3*3 and store the result in another matrix	16-17 (Odd)	23-33
11	Write a C program to multiply two matrices and display the resultant matrix.	16-17 (Even)	23-33
12	Write a program in C to find the largest number of a 4*4 matrix.	17-18 (Odd)	23-33
13	Write a program to add two matrices of dimension 3*3 and store the result in another matrix	17-18 (Odd)	23-33
14	Write a program to multiply two matrices(read size and number of elements of matrices from the keyboard	17-18 (Even)	23-33
15	Write a program in C to input two 3*3 matrix from the user and print multiplication as the result in matrix form.(Write comments also at appropriate places in the program)	18-19 (Even)	23-33
16	What are advantages of using array? Write a program for matrix multiplication of two matrix elements.	20-21 (Odd)	23-33
17	What is a String?	15-16 (Odd)	23-33
18	Explain in brief the purpose of 'strcmp' function.	15-16 (Odd)	23-33
19	Write a program in C to reverse a string by using pointer	17-18 (Odd)	23-33
20	Explain the significance of null character in string.	18-19 (Odd)	23-33
21	Differentiate between structure and Union.	15-16 (Odd)	23-33
22	Describe structure. Differentiate between structure and array. Define structure data type called time_structure containing three members's integer hour,integer minute and integer second.Write a progrom in C that would assign values to the individual members and display the time in the following form: 16:40:52	15-16 (Odd)	23-33
23	Explain struct with suitable example.	15-16 (Odd)	23-33.

24	Differentiate between structure and union in 'C'. Write a 'C' program to store the employee details using structure.	15-16 (Odd)	23-33
25	Explain with example ,the syntax and usage of the following in C programs I)Array and Structure	15-16(Odd) 15-16 (Even)	23-33
26	Describe the relation between structures and pointer.	2015 (Even)	23-33
27	What is the difference between array and structure? Write a program in C to find the largest number of a 4*4 matrix.	16-17 (Odd)	23-33
28	Define a structure? Write a program in C to create a data base of fifty students to store personal details such as roll no, name and marks .Print all the details of students whose name is entered by the user.	16-17 (Odd),17-18 (Odd)	23-33
29	Write a C program to store the employee information using structure and search a particular employee using employee number. Take the structure EMPL and its member employee No, employee Name , employee edu.	16-17 (Even)	23-33
30	Differentiate between structure and array.	17-18 (Odd)	23-33
31	Define structure with syntax. Also write a program that compares two given dates. To store data use structure say date that contains three members namely date, month and year. If the dates are equal then display message as "Equal" otherwise "Unequal"	17-18 (Even)	23-33
32	What is the difference between structure and union?	18-19 (Odd)	23-33
33	Explain the importance of structure in C programming. Write a program in C using Structure to enter and print the record of 10 books available in your library. Following fields may be included in the record: book_title , book_price and number_of_pages.	18-19 (Even)	23-33
34	Create a suitable structure in C language for keeping the records of the employees of an organization about their code , Name , Designation , salary , Department , City of posting. Also Write a program in C to enter the records of 100 employees and displays the name of those who earn more than 20,000	19-20 (Odd)	23-33
35	Differentiate structure and union.	19-20 (Odd) ,20-21 (Odd)	23-33
36	Write short notes on Union and Enumerated data type.	16-17 (Odd)	23-33
37	Write short note on following 1.Enumerated Data Type 2.String	18-19 (Odd)	23-33
38	Explain the following: i)Enumerated Data Type ii)Union	18-19 (Even)	23-33
39	Explain and write a C program to search a number in given 10 numbers	16-17 (Even)	23-33
40	Explain linear search and binary search technique for searching and item in a given array .Also write the complexity for each sorting algorithm.	18-19 (Odd)	23-33
41	Differentiate between linear search and binary search.	18-19 (Even)	23-33
42	What do you mean by sorting? Write program to sort the given n positive integers. Also give the flowchart for the same.	15-16 (Odd)	23-33

		16-17 (Odd)	
43	Write a program in C for sorting N positive integers in Ascending order	16-17 (Even)	23-33
44	Write a C program to sort set of integers in ascending order by using bubble sort technique.	17-18 (Even)	23-33
45	Write the importance of sorting in problem solving .Write a program in C using bubble sort technique to sort 10 numbers entered by the user.	18-19 (Even)	23-33
46	What do you mean by sorting ?Write a program in C to sort 'n' positive integers using bubble sort.Also draw the flow chart for the same.	19-20 (Odd)	23-33
47	Define sorting algorithm with example.	20-21 (Odd)	23-33
48	Explain selection sort technique for sorting problem. Write an algorithm for selection sort. Sort the following numbers using selection sort technique 26,54,93,17,77,31,44,55,20	18-19 (Odd)	23-33
49	Explain Selection sort with example.	20-21 (Odd)	23-33
50	What do you mean by order of complexity? Explain various notions to represent order of complexity with diagram.	18-19 (Odd)	23-33