Subject Name : C programming for problem Solving

Subject Code : 18CPS13/23 Q.P : Jan/Feb 2021

Faculty Name : Prof. Ravindra. T. Patil

---

1 a. Describe the various types of computers. 10 marks

According to purpose :

1) Analog computers: These are used to read & measure the physical phenomena such as Electrical, mechanical or hydraulic quantities. Ex : Thermometer.

2) Digital computers: These perform calculations & logical operations with quantities represented by digits usually in binary system. Ex: laptop, PCs.

c) Hybrid computers: A combination of both Analog and digital Computer, which perform complex simulation Ex: robot and super computers.

According to size :

1) Micro computer :

* It can be defined as a computer that has a microprocessor as its CPU.
* It can perform only basic operations.
* Used for small, individual tasks

Ex: Desktop machine. PC's in labs.

2) Mini computers:

* The computer whose capacity lies between mainframe & micro computer are called mini computers.
* Used in small & mid size busines organizations.

Ex: PDP-11, VAX-11

3) Mainframe computers:

* These are larger & expensive compared to microcomputers.

* Used as large servers and for intensive business applications.

* These computers act as stations.

Ex: IBM ES/9000.

4) Supercomputers:

* These are most powerful, Expensive, fastest comp available at present.

* Used for scientific appl which need high level of processing. Ex: weather forecasting

* Typically Employ parallel architecture.

Ex: Cray Supercomputer.

2. What is Printer. Explain the different types of printers. 8 marks.

printers produce hard copies of output which include text and graphics. There are 2 ~~types~~ category of printers

1) An impact printer is one where the print head will be in physical contact with paper.

2) In a non impact printer, print head will have no physical contact with paper.

Different impact printers are.

1) Dot matrix printer:

* print head has ~~Either~~ Either 9 or 24 pins which are fired in multiple combinations to generate letters, numbers, symbols.
* The ribbon is impregnated with ink and when the pins fire against the ribbon, an impression is created on the paper behind it.
* The paper itself is moved forward by a drum after one line of printing has been completed.

2) Line printers:

* For heavy printing, the line printer is still the best.
* This technology uses a print chain containing the characters.
* The chain rotates continuously in front of the paper
* Hammers strike the paper in the normal manner, but the print speeds attained with technology are as high as 1200 lpm (lines per minute).
* line printer is Extremely noisy, so they have to be kept in a separate room.

non impact printer.

1) Laser printers:

* It works like photocopier.
* A laser beam creates an image of the page to be printed on a light-sensitive drum.
* The charged area attract black magnetic powder called the toner.
* The image, which is created in the form of dots, is then transferred from the drum to the paper by actual contact.
* A separate roller heats up the paper to melt the toner which then gets fused onto the paper. Color laser printer use colored toner.

2) Ink-jet printer:

* It is an affordable non-impact printer whose print quality is considerably outperforms a dot matrix printer but underperforms a laser printer.
* A printhead sprays tiny drops of ink at high pressure as it moves along the paper.
* The ink stored in a replaceable catridge, passes through a matrix comprising a number of tiny nozzles.
* Like with the pins of a dot matrix printer, characters are formed by choosing the nozzles that how to be activated

1. c Define Software. Name the different types of Software. 2 marks.

→ Software is a collection of programs.

Different types of Softwares are

1) Application Software

2) System Software

2. a Define operators. Illustrate all the operators used in C language.

→ An operator is a symbol that specifies which operation to be carried out on operands.

C operators can be classified in to number of types Some of them are as follows.

1. Arithmetic operators:

The operators that are used to perform arithmetic operation such as addition, substraction, multiplication, division & motulus operation are called arithmetic operators.

Ex:

1) + : a+b , H+2

2) - : a-b , H-2

3) * : a*b , H*2

4) / : a/b , H/2

5) % : a%b H%2.

2) Relational Operators:

These are used to perform comparison of 2 operands. Result of relational operators is Either true or False.

Operator		
1)	<	→ H < 5 → TRUE
2)	<=	→ H <= 5 → TRUE
3)	>	→ H > 5 → FALSE.
4)	>=	→ H >= 5 → TRUE
5)	==	→ H == H → TRUE
6)	!=	→ H != 5 → TRUE

3) Logical operators:

These are used to combine 2 or more relational Expressions. The result of logical operators will be Either TRUE or FALSE.

Operator	
!	logical not
&&	logical and
||	logical or

Ex: ! 0 → 1

1 && 1 → 1

1 || 0 → 1

4) Increment or Decrement operators:

Increment operator (++) increases the value by 1 and Decrement operator (--) decreases the value of variable by 1.

Ex: post increment : a++ → operand value is used 1st & then value is incremented by 1.

pre increment : ++a → operand value is incremented by 1 bit and then used.

Ex:

```
a = 1
b = a++;  // b = 1

a = 2
b = ++a;  // b = 3.

a = 1
b = a--;  // b = 1

a = 2
b = --a;  // b = 1
```

5) Bitwise operator: Bitwise operators are used to manipulate data at the bit level.

operator	meaning.
~	Bitwise negate
<<	shift left.
>>	shift right
&	Bitwise AND
^	Bitwise OR
|	Bitwise Exclusive OR.

Special operators

1) comma operator : It is used to link the related Expressions together. Combines 2 or more statements in to a single statement.

2) Size of operator: It is used with operand to return the number of bytes of memory occupied by the operand.

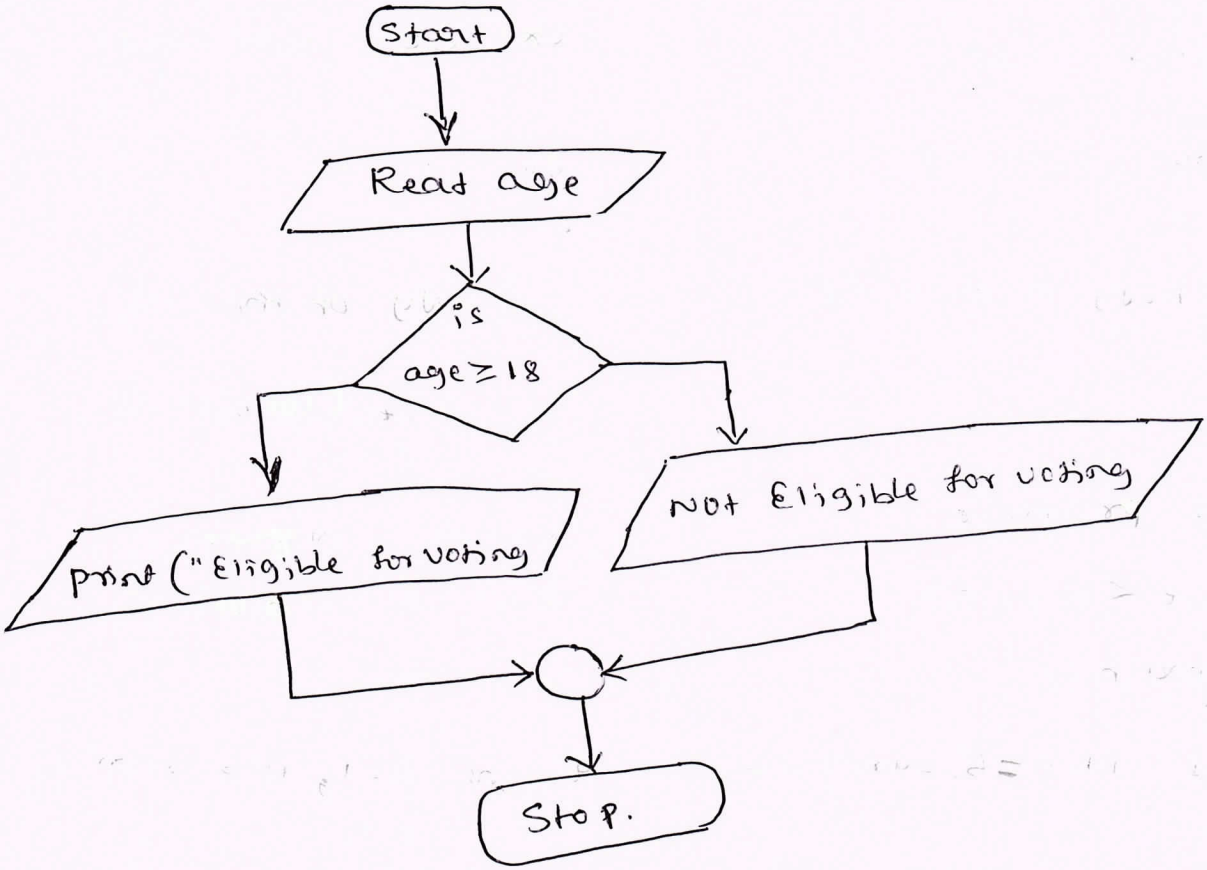
b. Write a C program to find the Eligibility for voting. Draw the flow chart for the same.

→

```
#include <stdio.h>
void main()
{
    int age;
    printf ("Enter age \n");
    scanf ("%d", &age);
    if (age ≥ 18)
    printf (" You are Eligible for voting \n");

    else
    printf (" Not Eligible for voting \n");

}
```

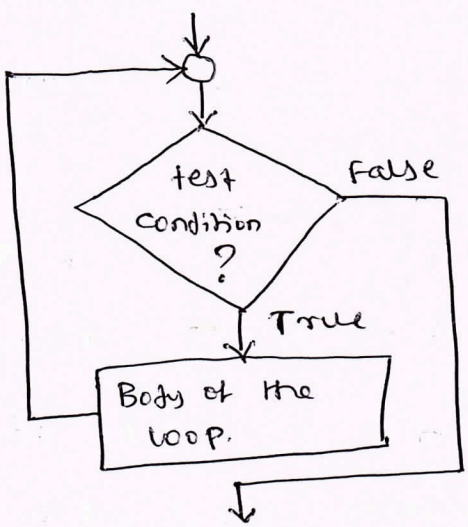


Module - II

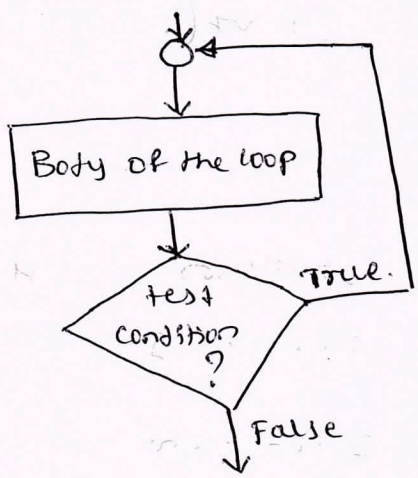
3. a Differentiate between Entry Control loop and Exit Control loop. Explain with Syntax & Example 10M.

→

* Entry control loop



Exit controlled loop.



* First condition is tested then if it is true Body of the loop Executes Else body of the loop will not be Executed

First Body of the loop is Executes then condition will be tested.

* Minimum one time body of the loop is not Executes when condition tests False

Minimum one time body of the loop Executes Even Condition results to false

* Ex: while loop

```
while (condition)
{
   // body of the loop
}
```

* Example program

```
#include <stdio.h>
void main()
{ int i=1, n=5, sum=0;
  while (i<=5)
  { sum=sum+i;
    i++;
  }
  printf("sum=%d", sum);
}
```

Ex: do-while loop.

```
do
{
  // body of the loop
} while (condition);
```

* Example program

```
#include <stdio.h>
void main()
{ int i=1, n=5, sum=0;
  do
  { sum=sum+i;
    i=i+1;
  } while(i<=n);
  printf("sum=%d", sum);
}
```

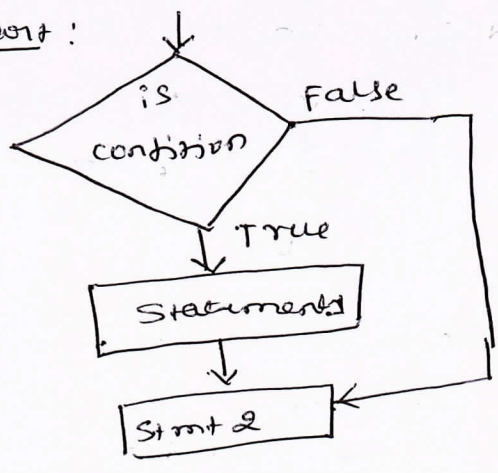
3. C Explain with Syntax, Flowchart Simple IF statement. 2m.

→ Simple IF Statement is used to execute a set of statement if the condition is true. If the condition is false it skips executing those set of statements.

Syntax:

```
if (condition)
{
  // statements
}
```

Flowchart:



3 b. Develop a C program to find the reverse of a positive integer and check for palindrome or not. 10m.

```
#include <stdio.h>
void main()
{
    int n, reverse=0, temp;
    printf("Enter a number\n");
    scanf("%d", &n);
    temp=n;
    while (temp!=0)
    {
        reverse = reverse*10;
        reverse = reverse + temp%10;
        temp = temp/10;
    }
    if (n==reverse)
    printf("%d is a palindrome\n", n);
    else
    printf("%d is not a palindrome", n);
}
```

4 a. Why conditional branching statements are needed in C program. Illustrate 5 types of branching statements in C program.

→ In some context it becomes inevitable to skip certain set of statements in order to execute

another set of statements. This type of selective Execution is achieves using conditional branch Statement.

Different conditional branching statements are

1 If Statement: It is used Execute a set of Statements if the condition is true.

Syntax:

```
if (condition)
{
  // statements
}
```

2. If else statement

Syntax:

```
if (condition)
{ S1;
  S2;
}
else
{ S3;
  S4;
}
```

It is used to Execute any one set of 2 set of Statements at a time. If condition is true it Executes one set of statements otherwise it executes another set of statements

3) Nested if else:

It is a multi decision statement which consist of if else control statement within another if or else control statement.

Syntax:

```
if (c1)
{
  if (c2)
    s1;
  else
    s2;
}
else
{
  if (c3)
    s3;
  else
    sH;
}
```

4) Cascaded if else (else if ladder): It is a multipath decision statement which consist of chain of if else where in the nesting take place only for else block.

Syntax:

```
if (c1)
    s1;
else if (c2)
    s2;
else if (c3)
    s3;
else
    sH;
```

SY switch statement:

It is a multiway decision making control statement used to make a selection between many alternatives.

Syntax:

```
switch (choice / expr)
{   case value1 : block1;
                  break;
        .
        .
        .
    case valuen : blockn;
                  break;
    default : block
}.
```

8M.

4 b. Write a C program to plot Pascal's triangle

```
#include <stdio.h>
void main()
{ int i, j, n, c[10][10];
  printf(" Enter the value of n\n");
  scanf (" %d", &n);
  for ( i=0 ; i<=n; i++)
    { printf ("\n");
      for ( j = 0 ; j <= n-i ; j++)
        printf (" ");
      for (j=0 ; j<=i ; j++)
    { if (j==0)
      c[i][j]=1;
      else if (i==j)
      c[i][j]=1;
```

```
        else
            c[i][j] = c[i-1][j-1] + c[i-1][j];
            printf("%d\t", c[i][j]);
        }
    }
}
```

4-C Explain loop control statement in C 2m.

→ loop control statements are used to execute a set of statement repeatedly.

Ex: for loop statement.

Module - III

5.a Define String. list all string manipulation functions Explain any 2 with example.

→ String is a ~~sequence~~ set of characters Enclosed within double quotes.

String manipulation functions are

1) Strcat() 2) Strlen() 3) Strcpy() 4) Strcpy()
5) Strncat() 6) Strncpy().

I) Strlen(): It is used to determine the number of characters present in a string.

Syntax: Strlen(Str)

```
char a[10] = "VTU";
len = Strlen(a);   // len = 3.
```

Ex:

II Strcpy(): It is used to copy the contents of one string to another.

Syntax: Strcpy(dest, source);

Ex: char a[10] = "Vsit";

char b[10];

Strcpy (b, a);

Here all the characters in source string will ge copied to destination string.

5. b Write a C program to count vowels and consonants in a string. &M.

→

```
#include <stdio.h>
void main()
{
    char s[20], ch;
    int i, v=0, c=0;
    printf("Enter a String\n");
    gets(s);
    for(i=0; i<strlen(s); i++)
    {  if (isalpha(s[i])
       { ch = tolower(s[i]);
         if (ch=='a' || ch=='e' || ch=='i' ||
             ch=='o' || ch=='u')
            v++;
         else
            c++;
       }
    }
    printf("Number of vowels = %d", v);
    printf("Number of Consonants = %d", c);
}
```

5 c Explain I/O functions for Strings. 2 M.

- gets() : This function is used to read String.

puts() : This function is used to print String.

OR

6.a Define array. Write the syntax for Declaring and initializing 1D and 2D Array with Suitable Example

→ Array is a set of Data elements of similar Data type.

Declaration of 1D array:

Datatype array_name[size];

Ex: int a[10];

Initialization of 1D array.

1. compile time initialization: Initialization done during declaration.

Syntax: Datatype array_name = {list of values};

Ex: int a[3] = {1, 2, 3};

int a[] = {1, 2}; // Array size can be omitted.

Array can be partially initialized

int a[10] = {1, 2};

char city[10] = {'a'};

2. Run time Initialization: Initialization done during program Execution.

```
int a[100];
for(i=0; i<10; i++)
scanf("%d", &a[i]);
```

Declaration 2D Array: (row, column)

Datatype array_name [size1] [size2];

Ex: int a [10] [10];

Initialization of 2D.

Datatype array_name [row] [col]

= { { a1, a2, a3 },
{ b1, b2, b3 },
- - - - - - - - -
- - - - - - - - -
{ c1, c2, c3 }
};

Ex: int a [2] [3] = { { 0, 0, 0 },
{ 1, 1, 1 }
};

2D array can be initialized partially.

int a [2] [3] = { { 1, 1 },
{ 2 }
};

6.b. Write a C program to find sum of diagonal Elements of matrix 5M.

```
#include <stdio.h>
void main ()
{
    int a [10] [10], i, sum=0, m, n;
```

```
printf(" Enter the value of m and n\n");
scanf(" %d %d", &m, &n);
printf(" Enter matrix A Elements\n");
for(i=0; i<m; i++)
    for(j=0; j<n; j++)
        scanf(" %d", &a[i][j]);

for(i=0; i<m; i++)
    sum = sum + a[i][i];

printf(" principle diagonal sum = %d", sum);
}
```

6. C write a C program to sort the numbers in ascending order using selection sort technique.

→
```
#include<stdio.h>
void main()
{ int num[20], n, i, j, temp;
  printf(" Enter number of elements\n");
  scanf(" %d", &n);
  printf(" Enter Elements\n");
  for(i=0; i<n; i++)
    scanf(" %d", &num[i]);
  for(i=0; i<=n-2; i++)
    { for(j=i+1; j<=n-1; j++)
      { if(num[i] < num[j])
        { temp = num[i];
          num[i] = num[j];
          num[j] = temp;
        }
    }
}
```

```
printf("Sorted elements are \n");
for (i=0; i<n; i++)
  printf("%d ", num[i]);
}
```

Module - 4.

7 a. What is function? Explain the different types of functions based on parameters. 10. M.

→ Function is a set of statements that performs specific task.

Different types of functions based on parameters are.

1. Function with parameter and return values.: In this category called function will receive values from calling function and when function returns a value calling function can receive a value from the called function.

```
#include <stdio.h>
int add(int p, int q);
void main()
{ int a=10, b=10, c;
c= add(a,b);
printf("%d\n", c);
}
```

```
int add(int p, int q)
{
  return(p+q);
}
```

2 Function with parameter and No return value:

Called function will receive value from calling function but calling function will not receive any value from called function.

```
#include <stdio.h>
void add(int p, int q);
void main()
{ int a=10, b=10;
  add(a,b);
}
```

```
void add(int p, int q)
{ int r;
  r = p+q;
  printf("%d\n", r);
}
```

3. Function with no parameter and return value:

```
#include <stdio.h>
int add();
void main()
{ int c;
  c = add();
  printf("%d\n", c);
}
```

```
int add()
{ int a=10, b=10;
  return(a+b);
}
```

Here only called function will send value to calling Function.

4] Function with no parameter and no return value.

In this category, both calling & called function will not exchange any values.

```
#include<stdio.h>
void print();
void main()
{
    print();
}
void print()
{
    printf("Hello world\n");
}
```

7.b. Explain recursion. Write a program to find factorial of a given number using recursive function. 10M.

→ The procen of function calling itself is called recursive. There are 2 types of recursion as given below 1) Direct recursion
2) Indirect recursion.

```
#include<stdio.h>
int factorial(int);
void main()
{
    int num, result;
    printf("Enter a number to find it's factorial\n");
    scanf("%d", &num);
    if (num<0)
        printf("Factorial of negative number is
                not possible\n");
```

```
else
{   result = factorial (num);
    printf (" The factorial of %d is %d", num, result);
}
}

int factorial (int num)
{
  if ((num == 0) || (num == 1))
    return 1;
    else
    return (num * factorial (num - 1));
}
```

OR

8a Write recursive functions for converting binary number to decimal number. 10m.

```
#include <stdio.h>
int BinToDec (int num);
void main()
{
    int bn, dn;
    printf (" Enter a binary number \n");
    scanf (" %d", &bn);
    dn = BinToDec (bn);
    printf (" Equivalent decimal number of
    %d is : %d ", bn, dn);
}
```

```
int BinToDec(int num)
{
    if (!(num/10))
        return (num);
    return (num % 10 + BinToDec(num/10)*2);
}
```

8. b Write a program to sort n numbers using bubble sort technique and using iterative function.

```
#include<stdio.h>
void bubble_sort();
void main()
{
  bubble_sort();
}
void bubble_sort()
{ int num[25], n, i, j, temp;
  printf("Enter the value of n\n");
  scanf("%d", &n);
  printf("Enter Element one by one\n");
  for(i=0; i<n; i++)
  scanf("%d", &num[i]);
  for(i=0; i<n; i++)
  { for(j=0; j<n-i-1; j++)
    { if(num[j] < num[j+1])
      { temp = num[j];
        num[j] = num[j+1];
      } } } num[j+1] = temp;
  printf("Sorted Elements are\n");
  for(i=0; i<n; i++)
}  printf("%d\n", num[i]);
```

Module - 5

9. a) Differentiate between structure and array. Explain the syntax of structure declaration in C with Example. 8M

→

Structure	Array
1) Structure is the collection of Heterogeneous data.	1) Array is collection of Homogeneous Data
2) Array Data are accessed using index Ex: a[2]	2) Structure Elements are access using . (DOT) Operator. Ex: S.name.
3) Individual Entries in a Structure are called members	3) Individual Entries in a array are called array Elements.
4) Keyword struct is used to declare Structure	4) No keyword are used.
5) A Structure is a programmer defined Datatype	5) Array is a Derived data type

Declaration of Structure: Keyword struct is used to declare structure.

Syntax:

```
struct struct_name
{
    Data_type member1;
    Data_type member2;
    Data_type member3;
};
```

Structure definition terminated with semicolon.

We can create the structure for a person as follow

```
struct person
{ int age;
  char name[50];
  float salary
};
```

9 b. Implement structure to read and write Book-title Book-author and Book-id for N books. 6M.

```
#include<stdio.h>
struct book_detail
{
    char book_title[50];
    char book_author[50];
    int book_id;
} b[20];

void main()
{   int n,
    printf("Enter the number of book details to read\n");
    scanf("%d", &n);
    for(i=0; i<n; i++)
    { printf("Enter%dbook_title, book_author, book id \n",i+1);
      scanf("%s %s %d", b[i].book_title,
          b[i].book_author, &b[i].book_id);
    }
    for(i=0; i<n; i++)
printf("Book %d title = %s\n author = %s\n
        book id = %d", i+1, b[i].book_title,
    b[i].book_author, b[i].book_id);
}
```

9. C Illustrate on 6M

i) Arrays within Structures.

ii) Arrays of Structure.

—>i) Arrays within structure.

It is possible to declare arrays within structure. In the following example character array is used for name.

```
Struct  Student
{   char  name[20]; // Array within Structure.
};
```

ii) Arrays of structures: In C we can have array of structures. For example suppose we want to store information of 10 students consisting of name. The structure definition can be written as

```
Struct  Student
{  char  name[20];
};
```

To store the information of 10 students the structure variable has to be declared as follows.

```
Struct  Student  S[10];
```

10 a what is preprocessor ? Explain types of preprocessor directives. 10 Marks

→ The preprocessor is a program that processes the source code before it passes through the Compiler.

There are 2 types of pre-processor directives.

1 Condition directives are used to instruct the preprocessor whether or not to include a chunk of code in the final token stream passed to the computer.

2 Unconditional directives performs well defined task.

The Unconditional directives are.

1) #define : It is used to define macros.

Syntax: #define Symbolic-name Symbolic-constant.

Ex: #define MAX 100

2) #include : It is used to inform preprocessor insert the Entire contents of Specified file into source program.

#include < file name >

Ex: #include <stdio.h>.

3) #line : This directive Enable's users to control the line numbers within the code files as well as the filename that appears when an Error takes place.

#line line-number filename.

2) conditional directives are as follows.

a) #if :

- It is used to control the compilation portions of source file.
- If specified condition has non-zero value, controlled text immediately following #if directive is executed

Syntax: #if condition controlled text

#endif.

b) #else :

- It is used within controlled text of #if directive to provide alternative text to be used if condition is false.

Syntax: #if condition

controlled text1

#else

controlled text2

#endif

c) #elif :

* It is used when there are more than 2 possible alternatives.

Syntax: #if condition

controlled text1

#elif new_condition

controlled text2

#else

controlled text3

#endif.

10M.

10 b. Develop a program using pointers to compute the sum and average of all Elements in an array.

```
#include <stdio.h>
#include <math.h>
void main()
{
    float a[10], *ptr, mean, sum=0;
    int n, i;
    printf("Enter the no of Elements \n");
    scanf("%d", &n);
    printf("Enter the array Elements \n");
    for(i=0; i<n; i++)
    scanf("%d", &a[i]);
    ptr = a;
    for(i=0; i<n; i++)
    { sum = sum + *ptr;
      ptr++;
    }
    mean = sum/n;
    printf("Sum of array Elements = %d", sum);
    Printf("\nAverage of array Elements = %d", mean);
}
```