USN

Internal Assessment Test 1 – September 2019

Sub:	C Programming for Problem Solving				Sub Code:	18CPS13	Branch:	1st Year (CSE)
Date:		Duration:	90 min's	Max Marks: 50	Sem / Sec:	I sem – Chem Cycle		OBE

Answer any FIVE FULL Questions

		MARKS	CO	RBT
1 (a)	Write 4 generations of computer with device, speed, size and example.	[5]	CO1	L2
(b)	Explain any 3 input and output devices with their use.	[5]	CO1	L1
2	Define an algorithm. Write an algorithm or flowchart & program to find greatest of three numbers using nested if statement.	[10]	CO2	L2
3 (a)	Write the basic structure of C program. Explain each sections briefly with suitable example.	[04]	CO2	L2
(b)	Evaluate each of the following expression independent of each other. The declaration & initialize statement is … int i=3,j=4,k=2; (a).i++ - j-- (b).++k % --j (c).j+1/i-1 (d).j++ / i--	[06]	CO2	L2
4	Write a C program to read 5 subject marks and find average. If the average is below 40 print fail,if average marks in between i) 40 to 50 print grade E ii) 50 to 60 print grade D iii) 60 to 70 print grade C iv) 70 to 80 print grade B v) 80 to 90 print grade A vi) 90 to 100 print grade S.	[10]	CO2	L3
5	Define branching statements. Explain if else, nested if, else if ladder statements syntax with examples.	[10]	CO2	L3
6 a)	Define a token? List different types of tokens available in C language? Explain	[5]	CO2	L3
b)	Evaluate the expression: a + 2 > b || !c && a = = d *a – 2 < = e Where a=11, b=6, c=0, d= 7 and e=5.	[5]	CO2	L4
7.	Write a C program to find the roots of the quadratic equation by accepting the coefficients.	[10]	CO2	L3
8.	a) Write a C program for swapping of two numbers. b) Write a C program to demonstrate size of() operator.	[10]	CO2	L3

C Programming for Problem Solving(18CPS13)
Scheme of Evaluation for IAT1

---

1 a) Write 4 generations of computer with device, speed, size and example. 5M

Computer Generations

Generation	Duration	Devices	Purpose	Size & Speed	Examples
First 1	1940-50	Vacuum Tubes & Plug boards	General Purpose Electromechanical Systems	Huge & Very Slow	ENIAC, UNIVAC-I
Second	1950-60	Transistors/Semiconductors	Batch processing, Punched cards, Magnetic Tape	Little Big & slow	IBM 1401, MARK III, UNIVAC 1107
Third	1960-70	IC(Integrated Circuits)	Parallel Processing, OS to manage Hardware, software & resources	Smaller & faster, Storage devices	IBM 360
Fourth	1970 onwards	LSI (Large Scale Integration)	GUI, Microprocessor, Networks & Internet	Small & very fast, Large storage capacity in MB	Intel C4004 8085, 8086, 80386 & 80486 Desktop PCs, Main Frame & Super computers
	Going on	VLSI	GUI & Network OS Ubuntu, , Android	Very small but Very-very Faster, Storage in Terabytes	Pentium I, II, III, IV, Dual core Laptops, Tablets, Smart Phones

b) Explain any 3 input and output devices with their use. 5M

The devices which are used to input the data and the programs in the computer are known as "Input Devices.

Output Device can produce the final product of machine processing into a form usable by humans. It provides man to machine communication.

Some of the I/O devices are explained below:

(1) **Keyboard** : Keyboard is used in the input phase of a computer-based information system. Keyboard is most common input device is used today. The data and instructions are input by typing on the keyboard.he message typed on the keyboard reaches the memory unit of a computer.

(2) **Mouse** : It's a pointing device. The mouse is rolled over the mouse pad, which in turn controls the movement of the cursor on the screen. We can click, double click or drag the mouse. Most of the mouse's have a ball beneath them, which rotates when the mouse moved. The ball has 2 wheels of the sides, which in turn mousse with the movement of the ball. The sensor notifies the speed of its movements to the computer, which in turn moves the cursor/pointer on the screen.

3) **Scanner** : Scanners are used to enter information directly into the computer's memory. This device works like a Xerox machine. The scanner converts any type of printed or written information including photographs into digital pulses, which can be manipulated by the computer.

4) Plotter : Plotter is an O/P device that is used to produce graphical O/P on papers. It uses a single color or multicolored pens to draw pictures as blue print etc.

5) **Digital Camera** : It converts graphics directly into digital form. It looks like an ordinary camera, but no film is used therein, instead a CCD (changed coupled Divide) Electronic chip in used. When light falls on the chip through the lens, it converts light waves into electrical waves.

6) **Printer:** Printers are used to display information directly from the computer's memory.

2.Define an algorithm. Write an algorithm or flowchart & program to find greatest of three numbers using nested if statement. 10M

Algorithm can be defined as Step by step procedure to solve any problem.

Algorithm and Flowchart

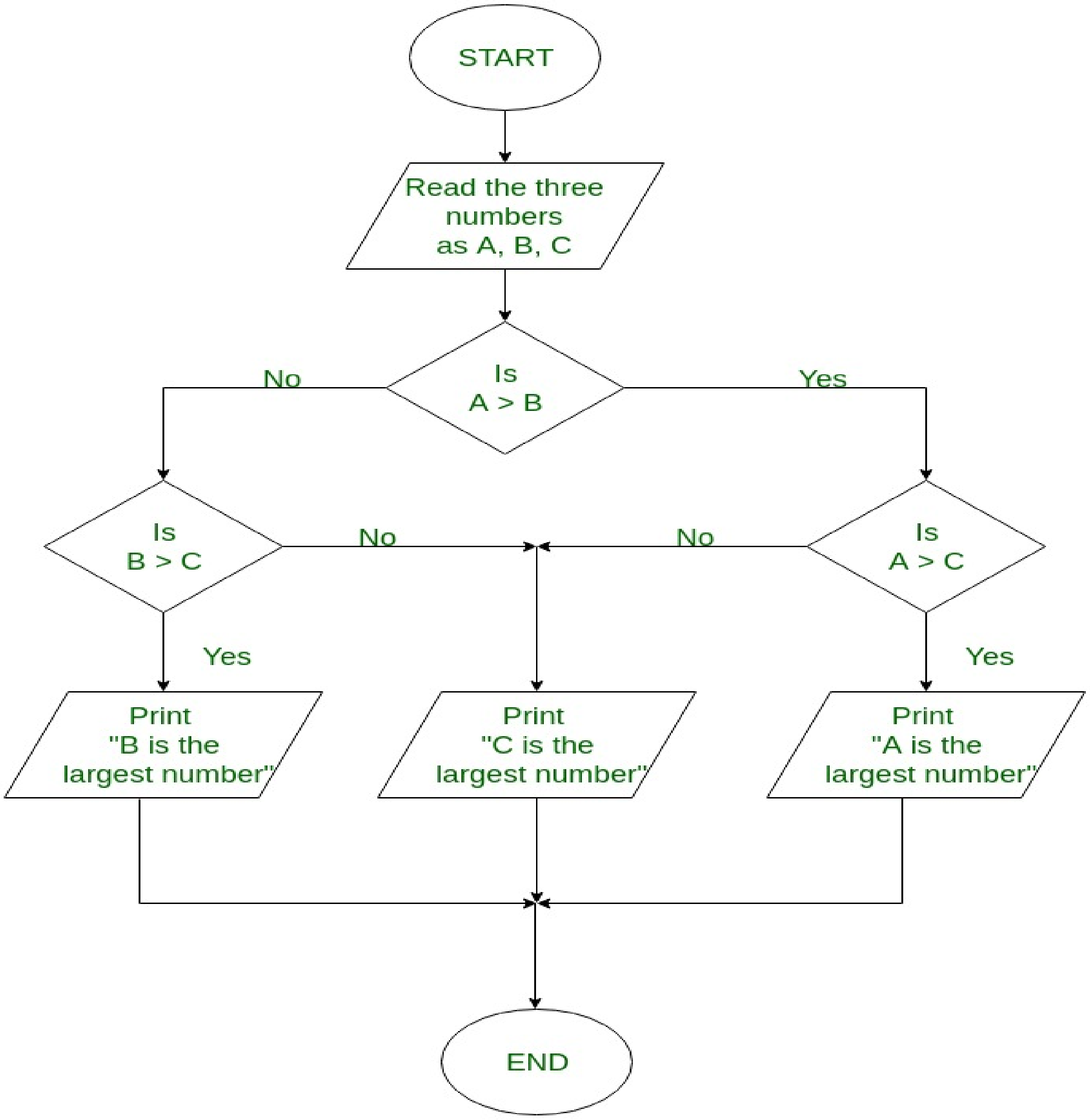
```
Step 1 : Start
Start 2 : Input a, b, c
Start 3 : if a > b goto step 4, otherwise goto step 5
Start 4 : if a > c goto step 6, otherwise goto step 8
Start 5 : if b > c goto step 7, otherwise goto step 8
```

Start 6 : Output "a is the largest", goto step 9

Start 7 : Output "b is the largest", goto step 9

Start 8 : Output " c is the largest", goto step 9

Start 9 : Stop


START
Read the three numbers as A, B, C
Is A > B
No
Yes
Is B > C
No
No
Is A > C
Yes
Yes
Print "B is the largest number"
Print "C is the largest number"
Print "A is the largest number"
END

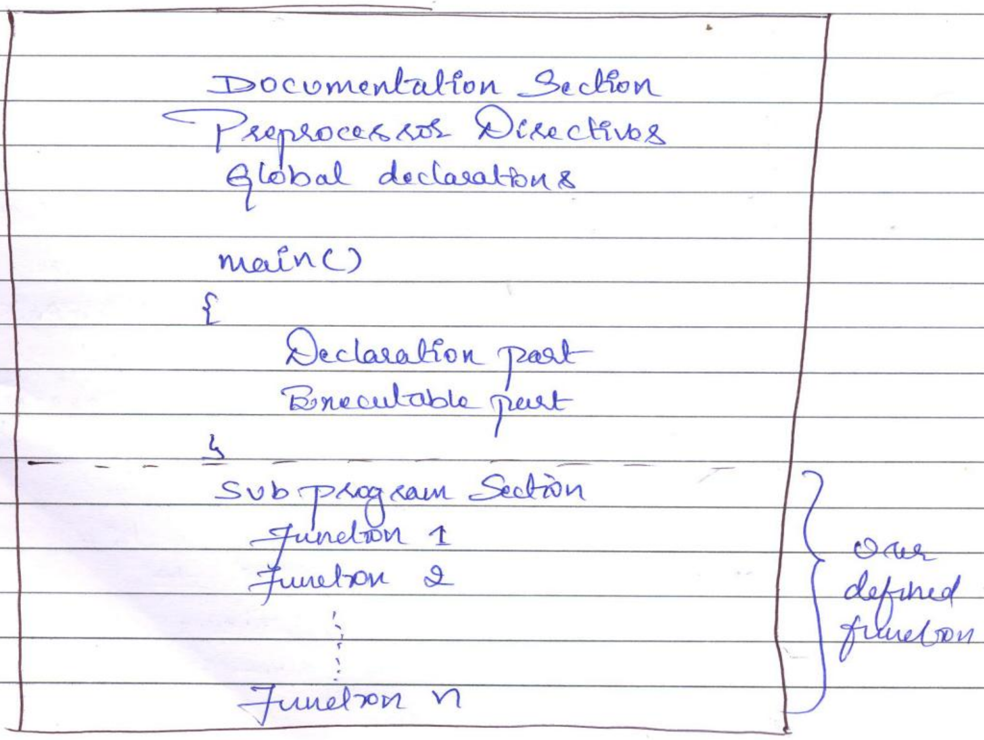
```
#include <stdio.h>
int main()
{
   int A, B, C;
   printf("Enter three numbers: ");
   scanf("%d %d %d", &A, &B, &C);
    if (A > B)
   {
      if (A > C)
         printf("%d is the largest number.", A);
      else
         printf("%d is the largest number.", C);
    }
   else
          {
      if (B > C)
         printf("%d is the largest number.", B);
      else
         printf("%d is the largest number.", C);
    }

   return 0;
}
```

3.a) Write the basic structure of C program. Explain each sections briefly with suitable example **6M**

8M Explain structure of C program along with an example?

X. **Structure of C Program**



Documentation Section:

Consists of Set of "Comment lines" giving the name of the program, Author and other details. They are not processed by the Compiler. two types ~~Preprocessor~~ of comment line

i) Single line Comment
2) multi line comment

Single line comment: starts with two forward slashes (//) and terminated with end of line.

Multi line comment: Starts with /* and terminates with */. A multi line comment is used when multiple lines of text are to be commented

Preprocessor Directive Section.

⇒ "# include" is known as preprocessor Directive which inform the compiler that will be using parts of the standard function library.

⇒ each function in a library performs one particular task

⇒ '#' is an instruction to the compiler, that to include Standard library function to the program

⇒ < > ⇒ tell the compiler the exact location of header file.

Completely #include < header file >
↓
Stdio.h

#include<stdio.h>

Global Declarations

* Variables which are declared globally can be used by or accessible from any functions present in a progra

main()

* Function where C program will start executing. Every program must have a main function since this is where the computer begins to execute the program. It shoud start with '{' and ending with }

Local Declaration

* Variables which are declared locally, can be used or accessible from that function.
  i.e Scope is limited within function.

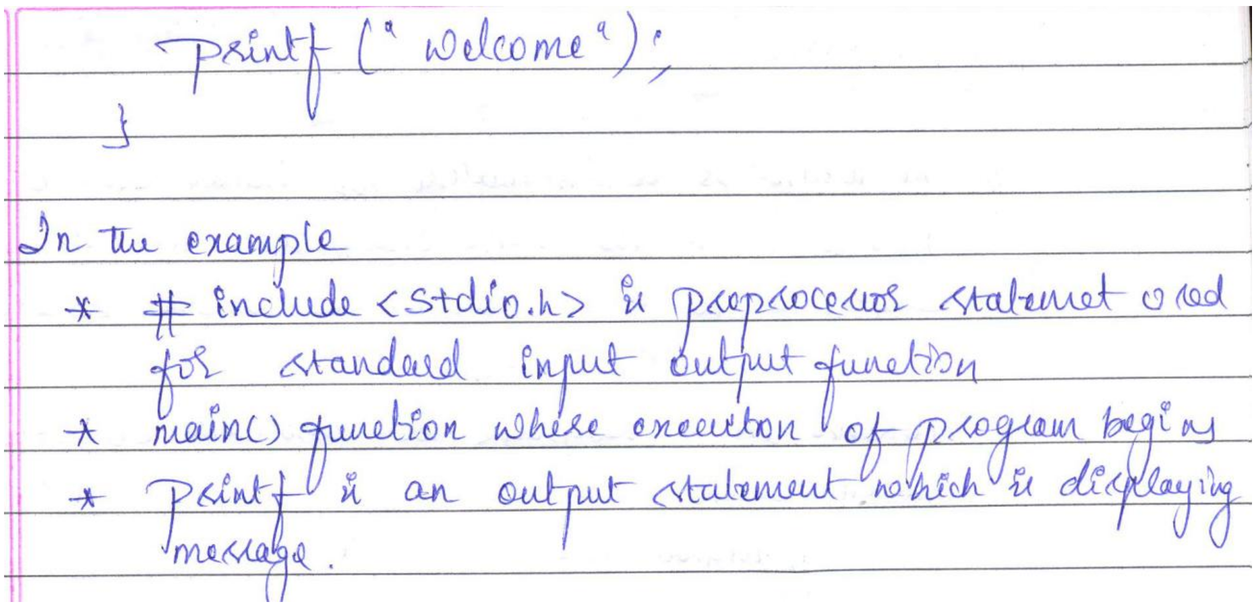
Executable part

* This part consisting set of statements, where the task to be achieved.

Subprogram Section

* Section which consist of user defined function which are defined by the user

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



```
    printf ("welcome");
}
```

In the example

* # include <stdio.h> is preprocessor statement used for standard input output function
* main() function where execution of program begins
* printf is an output statement which is displaying message.

b. Evaluate each of the following expression independent of each other.
The declaration & initialize statement is … int i=3,j=4,k=2; **4M**
(a).i++ - j-- **(b).++k % --j** **(c).j+1/i-1** **(d).j++ / i--**

a) i++ - j--
3++ - 4--
3++ - 4
3-4 = -1
i=4 j=3

b)
++k%--j
++2 % --4
++2 % 3
3%3 = 0
k=3 j=3

c) j+1/i-1
4+1/3-1
4+0-1
4-1 = 3

d) j++/i--
4++/3--
4++/3
4/3 = 1
j=5 i=2

4.Write a C program to read 5 subject marks and find average. If the average is below 40 print fail,if average marks in between i) 40 to 50 print grade E ii) 50 to 60 print grade D iii) 60 to 70 print grade C iv) 70 to 80 print grade B v) 80 to 90 print grade A vi) 90 to 100 print grade S. 10M

```
#include<stdio.h>
int main()
{
        int m1,m2,m3,m4,m5,avg;
        printf("enter 5subject marks:");
        scanf("%d%d%d%d%d",&m1 ,&m2,&m3,&m4,&m5);
        avg=(m1+m2+m3+m4+m5)/avg;
        if(avg>90 && avg <=100)
                printf("GRADE S");
        else if(avg>80 && avg <=90)
                printf("GRADE A");
        else if(avg>70 && avg <=80)
                printf("GRADE B");
        else if(avg>60 && avg <=70)
                printf("GRADE C");
        else if(avg>50 && avg <=60)
                printf("GRADE D");
        else if(avg>=40 && avg <=50)
                printf("GRADE E");
        else
                printf("Fail");
        return 0;
}
```

5. Define branching statements. Explain if else, nested if, else if ladder statements syntax with examples

10M

⇒ Branching Statements

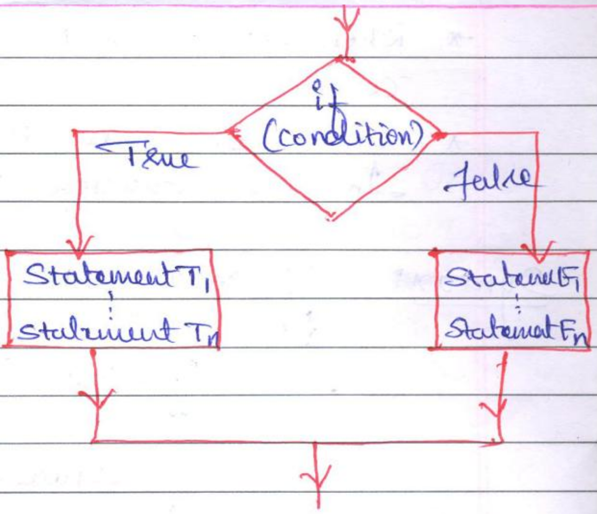
* Flow of control or program execution Order can be changed based on condition or without condition by using some set of statements. Such statements are known as Branching statements.

if-else

* It is also known as two-way decision statement.
* One set of statements has to executed when the condition is true and another set of activities have to be performed when the condition is false.

Syntax:

```
if (condition)
{
        Statement T1;
            :
        statement Tn;
}
else
{
        Statement F1;
            :
        Statement Fn;
}
```



where if is a keyword
condition can be a expression. If it is evaluated for TRUE, Statements from T1 -- Tn will be executed, if it is FALSE Statements from F1 --- Fn will be executed.

example: int age= 30

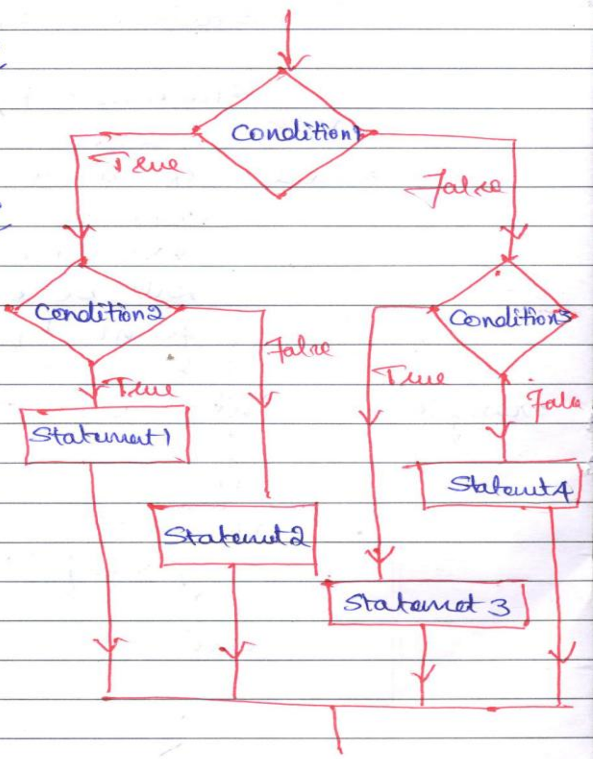
```
    if (age>= 18)
    {
        printf ("eligible to vote");
    }
    else
    {
        printf ('not eligible to vote");
    }
```

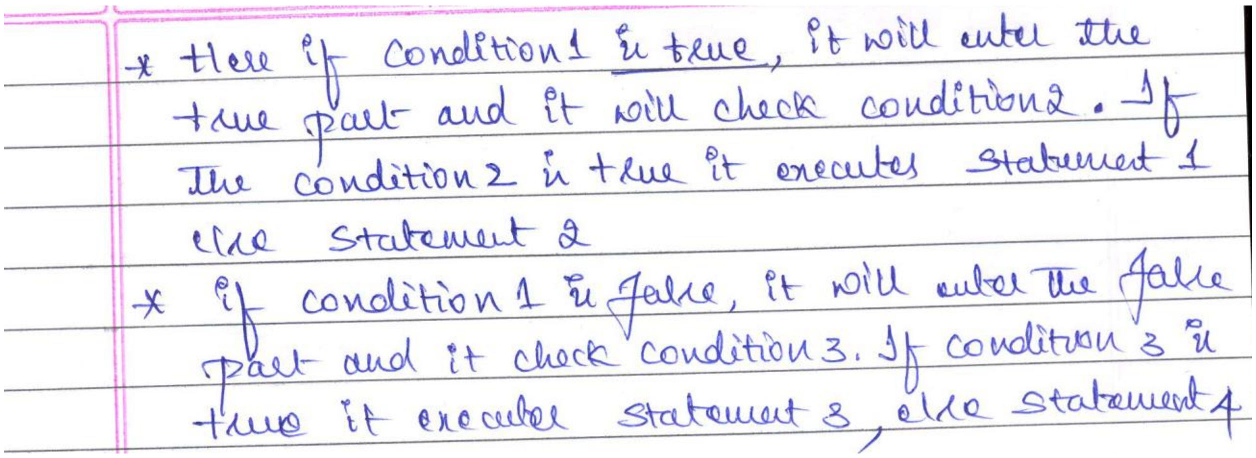
nested-if

* An if or if-else statement within another if or or if-else statement is called nested if statement.
* When the action have to performed more than one condition, then will use nested if statement.
* One condition is dependent on another condition. In such situations will use nested if.

yntax:

```
    if (condition1)
    {
        if (condition2)
        {
            Statement 1;
        }
        else
        {
            statement 2;
        }
    }
    else
    {
        if (condition3)
        {
            Statement 3;
        }
        else
        {
            Statement 4;
        }
    }
```





* Here if condition1 is true, it will enter the true part and it will check condition2. If The condition2 is true it executes statement 1 else statement 2
* if condition 1 is false, it will enter the false part and it check condition3. If condition 3 is true it executes statement 3, else statement 4

If – else ladder Statement

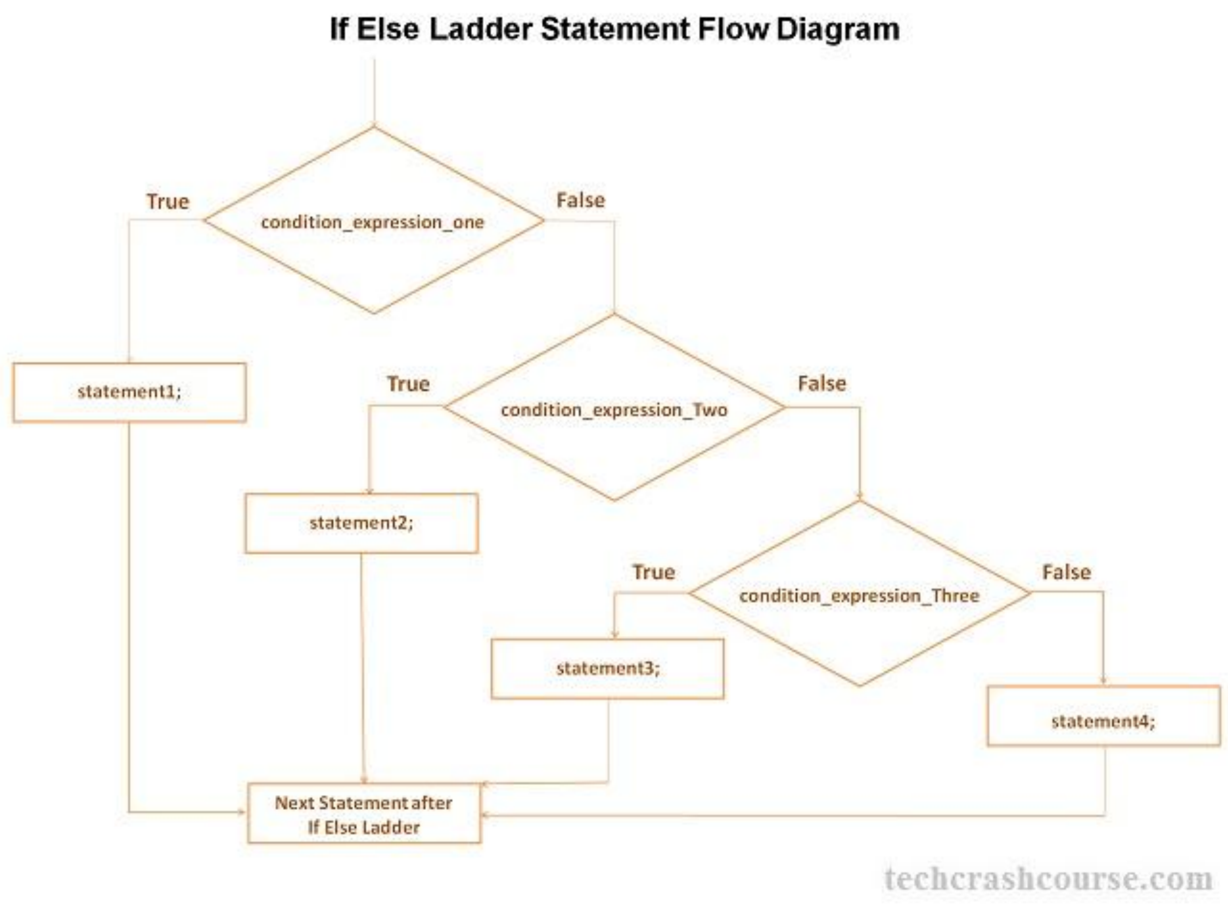
The if-else-if ladder statement executes one condition from multiple statements. The execution starts from top and checked for each if condition. The statement of if block will be executed which evaluates to be true. If none of the if condition evaluates to be true then the last else block is evaluated.

```
if(condition_expression_One) {
   statement1;
} else if (condition_expression_Two) {
   statement2;
} else if (condition_expression_Three) {
   statement3;
} else {
   statement4;
}
```

- First of all condition_expression_One is tested and if it is true then statement1 will be executed and control comes out out of whole if else ladder.
- If condition_expression_One is false then only condition_expression_Two is tested.

  Control will keep on flowing downward, If none of the conditional expression is true.

  The last else is the default block of code which will get executed if none of the conditional expression is true.



6 a) Define a token? List different types of tokens available in a C language? Explain 6M

A token is the smallest element of a program that is meaningful to the compiler. Tokens can be classified as follows:

1. Keywords
2. Identifiers
3. Constants
4. Special Symbols
5. Operators

Keyword: Keywords are pre-defined or reserved words in a programming language. Each keyword is meant to perform a specific function in a program. Since keywords are referred names for a compiler, they can’t be used as variable names because by doing so, we are trying to assign a new meaning to the keyword which is not allowed.

Identifiers: Identifiers are used as the general terminology for naming of variables, functions and arrays. These are user defined names consisting of arbitrarily long sequence of letters and digits with either a letter or the underscore(_) as a first character. Identifier names must differ in spelling and case from any keywords

Constants: Constants are also like normal variables. But, only difference is, their values can not be modified by the program once they are defined. Constants refer to fixed values. They are also called as literals.

Special Symbols: The following special symbols are used in C having some special meaning and thus, cannot be used for some other purpose.[] () {}, ; * = #

- **Brackets[]:** Opening and closing brackets are used as array element reference. These indicate single and multidimensional subscripts.
- **Parentheses():** These special symbols are used to indicate function calls and function parameters.
- **Braces{}:** These opening and ending curly braces marks the start and end of a block of code containing more than one executable statement.
- **comma (,):** It is used to separate more than one statements like for separating parameters in function calls.
- **semi colon :** It is an operator that essentially invokes something called an initialization list.
- **asterick (*):** It is used to create pointer variable

Operators: Operators are symbols that triggers an action when applied to C variables and other objects. The data items on which operators act upon are called operands.

Depending on the number of operands that an operator can act upon, operators can be classified as follows:

- **Unary Operators:** Those operators that require only single operand to act upon are known as unary operators.For Example increment and decrement operators
- **Binary Operators:** Those operators that require two operands to act upon are called binary operators. Binary operators are classified into :
    1. Arithmetic operators
    2. Relational Operators
    3. Logical Operators
    4. Assignment Operators
    5. Conditional Operators
    6. Bitwise Operators
    7. Ternary Operators

b) Evaluate the expression:

a + 2 > b || !c && a = = d *a – 2 < = e Where a=11, b=6, c=0, d= 7 and e=5

11+2>6 || !0 && 11==7 *11-2<=5

11+2>6 ||1 && 11==7 *11-2<=5

11+2>6 ||1 && 11==77-2<=5

13>6 ||1 && 11==77-2<=5

13>6 ||1 && 11==75<=5

1||1 && 11==75<=5

1||1 && 11==0

1||1 && 0

1||0

1 **4M**

7.**Write a C program to find the roots of the quadratic equation by accepting the coefficients.**
10M

```
#include<stdio.h>
    #include<math.h>
    int                                                                     main()
    {
            float                                        a,b,c,desc,r1,r2,realpart,imgpart;
            printf("Enter     the     coefficients     of     a,     b     and     c     :");
            scanf("%f%f%f",&a,&b,&c);

            if(a==0)
            {
                    printf("Coefficient     of     a     cannot     be     zero....\n");
                    printf("Please                    try                    again....\n");
                    return                                                             1;
            }

            desc=(b*b)-(4.0*a*c);

            if(desc==0)
            {
                    printf("The     roots     are     real     and     equal\n");
                    r1=r2=(-b)/(2.0*a);
                    printf("The     two     roots     are     r1=r2=%f\n",r1);
            }
            else                                                         if(desc>0)
            {
                    printf("The     roots     are     real     and     distinct\n");
                    r1=(-b+sqrt(desc))/(2.0*a);
                    r2=(-b-sqrt(desc))/(2.0*a);
                    printf("The     roots     are     r1=%f     and     r2=%f\n",r1,r2);
            }
            else
            {
                printf("The                roots              are               imaginary\n");
                realpart=(-b)/(2.0*a);
                imgpart=sqrt(-desc)/(2.0*a);
                printf("The    roots    are    r1=%f    +    i    %f\n",realpart,imgpart);
                printf("r2=%f            -            i            %f\n",realpart,imgpart);
        }
        return                                                                              0;
}
```

8.a) Write a C program for swapping of two numbers. **5M**

```
#include<stdio.h>
int main()
{
        int a,b,temp;
        printf("enter two numbers:");
        scanf("%d%d",&a,&b);
        temp=a;
        a=b;
        b=temp;
        printf("a=%d b=%d",a,b);
return 0;
}
```

b) Write a C program to demonstrate size of() operator. **5M**

```
#include<stdio.h>
int  main()
{
        int a;
        float b;
        char c;
        double d;
        printf("%d",sizeof(a));
        printf("%d",sizeof(b));
        printf("%d",sizeof(c));
        printf("%d",sizeof(d));
return 0;
}
```