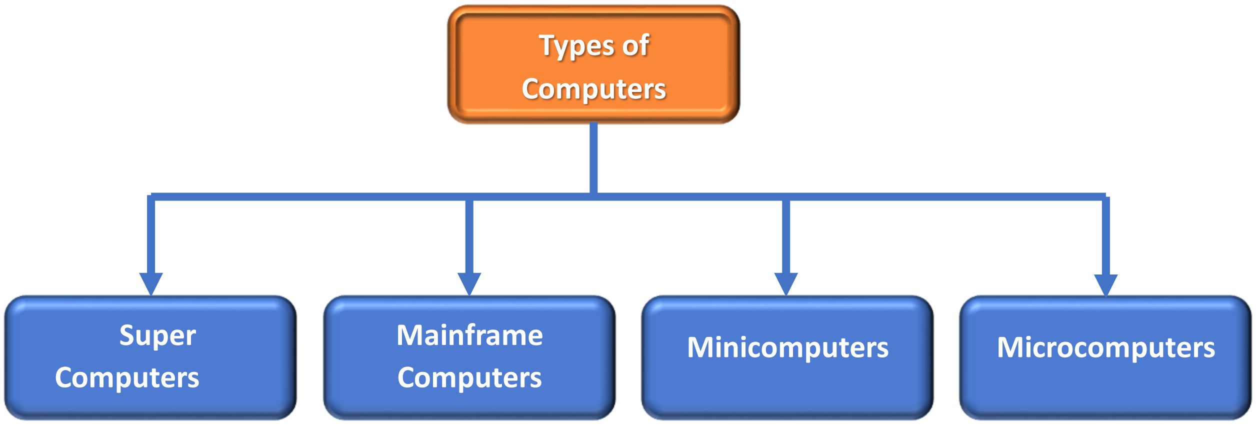
IAT-1 2021-22 EVEN SEM SCHEME (Problem Solving through Programming)

1.a) Describe various types of computers according to the purpose, size, and speed.

(5 Marks)

For listing out all types of computers (1 mark)



For comparison (2 Marks)

Factor\ Computer Type	**Super**	**Mainframe**	**Mini**	**Micro**
Purpose	Used for complex task such as: Weather forecasting, Scientific Simulation, Defence, Education, etc.	It is used is finance, large business, academics, healthcare, research.	It is mainly used in small organization.	Word processing, Accounting, Desktop publishing
Size	Large	Small compared to super and larger than mini and micro.	Small	Smaller than mini.
Speed	High	Low compared to Supercomputer	Low	Average

For Description of each type of computer (2 marks)

Supercomputer:

- ✓ A powerful computer capable of processing enormous amounts of data and performing complex calculations quickly.
- ✓ It consists of several computers that work in parallel as a single system.

Mainframe:

- ✓ It is called as enterprise servers.
- ✓ It occupies more area.
- ✓ Used for centralized computing.

Microcomputer (or) Personal computer:

- ✓ A small computer that can only be used by one person at a time.
- ✓ A digital computer that works on a microprocessor.
- ✓ It is used in homes and offices.

Minicomputer:

- ✓ A computer that falls between a mainframe and a microcomputer in terms of size, power, speed, storage capacity, and other factors.
- ✓ Minicomputers have been referred to as small or midsize servers in recent years (a server is a central computer that provides information to other computers).

1.b) Evaluate the below expression. (5 Marks)

a+2>b||!c&&a==d||a-2<=e   where a=11,b=6,c=0,d=7 and e=5.

Marks split-up (evaluate expression as per operator precedence-3 and answer-2 marks)

Step 1: Replace variables with values

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

Step 2: Out of ! and (+,-) , give priority to ! operator (!0=1)

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

Step 3: Perform arithmetic operations

11+2=13

11-2=9

13>6||1&&11==7||9<=5

Step 4: (<= is followed by >)

9<=5=0

13>6=1

1||1&&11==7||0

Step 5: (11==7)=0

1||1&&0||0

Step 6: (&& is followed by ||)

1&&0=0

1||0||0

1||0=1 //evaluate

Finally , we get

1||0=1

2.a) Describe the formatted input output functions with syntax and suitable example

(5 Marks)

User can interact with program using i/o operations.

Write about printf() and scanf() function **(3 marks)**

//how to read integer, float, char

// how to display output

Reading character and string using predefined function such as getchar(), putchar(),gets(), puts() method **(2 marks)**

2.b) Differentiate between Primary Memory and Secondary Memory. (5 Marks)

For each difference **0.5 marks**. Totally write 10 differences between primary and Secondary.

Primary Memory	**Secondary Memory**
It is temporary memory	It is permanent memory
It is volatile in nature	It is non-volatile in nature
The memory has fast access time	The memory has low access time
It is expensive	It is inexpensive
It has limited storage capacity	It has vast storage capacity
Data can be directly accessed by the CPU	Data cannot be directly accessed by the CPU
It contains data which is currently used by the CPU	It contains data which is not currently used by the CPU
Data transfer between CPU and primary memory is managed by the cache memory	Data transfer between CPU and Secondary memory is managed by the IO processor
Primary memories are semiconductor memories	Secondary memories are magnetic memories

It is used for processing data	It is used for storing data
It is also called as internal memory	It is also called as external memory
Example: RAM, ROM, etc.	Example: Hard Disk, Pen Drive

3.a) Define Token. Explain the keyword, identifiers, and constants in detail. (5 Marks)

Define token (mark)

Individual element of c program is called token.

Token can be a keyword, identifier and constant.

Definition and explanation of keyword (2 marks)

Keywords are predefined, reserved words used in programming that have special meanings to the compiler. Keywords are part of the syntax and they cannot be used as an identifier.

Totally 32 keywords in c language.

auto	double	int	struct
break	else	long	switch
case	enum	register	typedef
char	extern	return	union
continue	for	signed	void
do	if	static	while
default	goto	sizeof	volatile
const	float	short	unsigned

Definition and explanation of identifier (2 marks)

A name which is given to variable, function, structure and array is called identifier.

Rules for naming identifiers

- ✓ A valid identifier can have letters (both uppercase and lowercase letters), digits and underscores.
- ✓ The first letter of an identifier should be either a letter or an underscore.
- ✓ You cannot use keywords like int, while etc. as identifiers.
- ✓ There is no rule on how long an identifier can be. However, you may run into problems in some compilers if the identifier is longer than 31 characters.

For example:

int student;

float marks;

Here, student and marks are identifiers.

We have to remember that the identifier names must be different from keywords. We cannot use int as an identifier, because int is a keyword.

Consider another example for identifier.

int var1, var2;

float Avg;

function sum();

Here,

- ✓ int, float, function are all keywords.
- ✓ var1, var2, Sum, Avg, are the identifiers.

2ndsem	- Not valid //starting with number
$num1	-Not valid //starting with special symbol
Add	-Valid //starting with alphabet
a_2	-Valid // underscore allowed in between variable
for	-Not valid // for is the keyword
1st_paper_marks	-Not valid //should not start with number

Definition and explanation of constant (1 marks)

A constant is a name given to the variable whose values can't be altered or changed. A constant is very similar to variables in the C programming language, but it can hold only a single variable during the execution of a program.

Write about integer, floating-point and character constant.

For example

const int age=18;

const float PI=3.142;

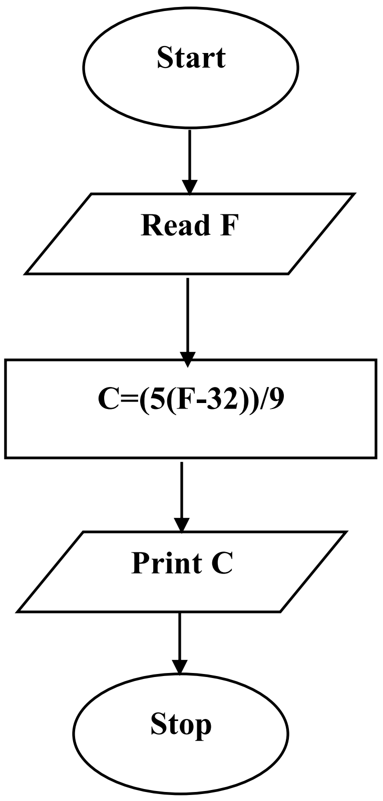
3.b) Write algorithm and flowchart to convert Fahrenheit temperature to Centigrade.

(5 Marks)

Algorithm: (2 Marks)

Step 1: Start.
Step 2: Read F.
Step 3: C=(5(F-32))/9.
Step 4: Print C.
Step 5: Stop.

Flowchart: (3 Marks)



4.a) Define two-way selection statement. Explain if, if_else, nested_if and ladder else_if with suitable syntax flowchart and example. (10 Marks)

Definition for two-way selection statement: (1 Marks)

The two-way selection is the basic decision statement for computers. The decision is based on resolving a binary expression, and then executing a set of commands depending on whether the response was true or false.

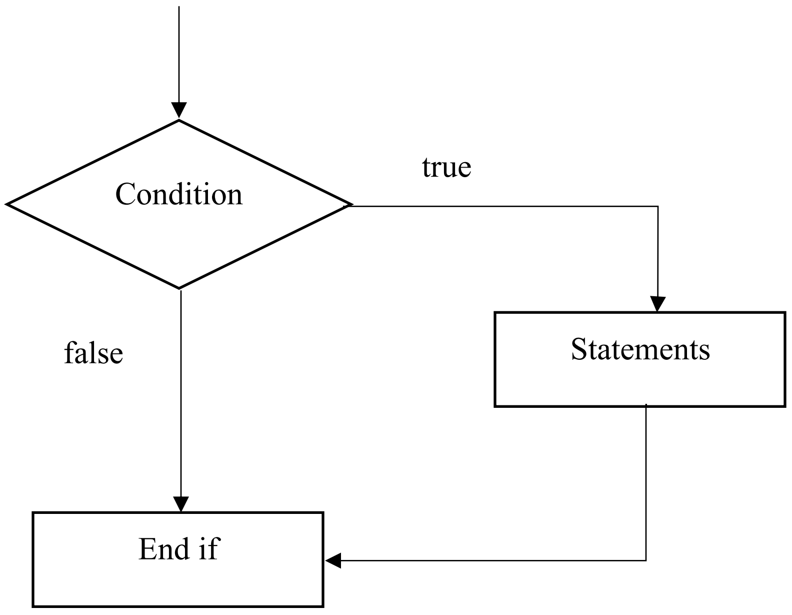
if (2 Marks)

The statement inside the if block is executed only when condition is true, otherwise not.

Syntax:

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

Flowchart:



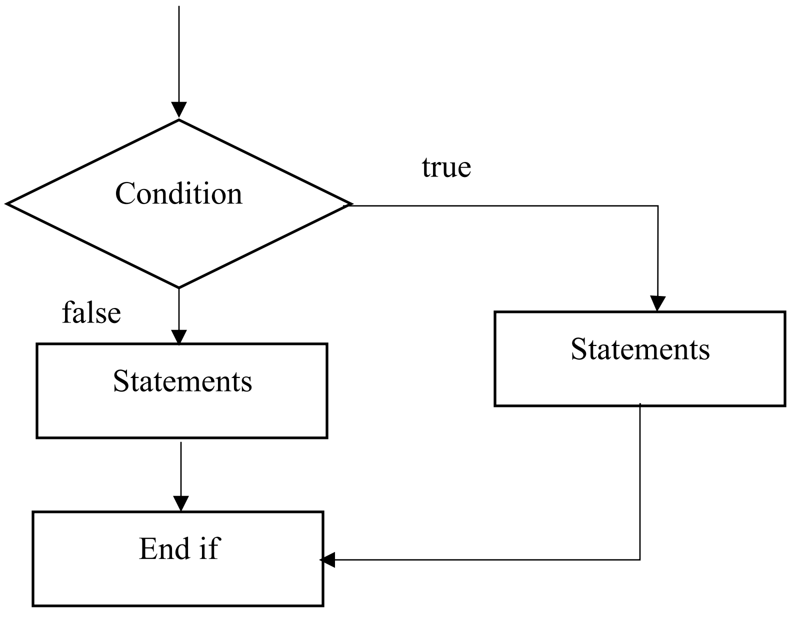
if else (2 Marks)

The statement inside the if block is executed only when condition is true, otherwise another set of statements will be executed.

Syntax:

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

Flowchart:



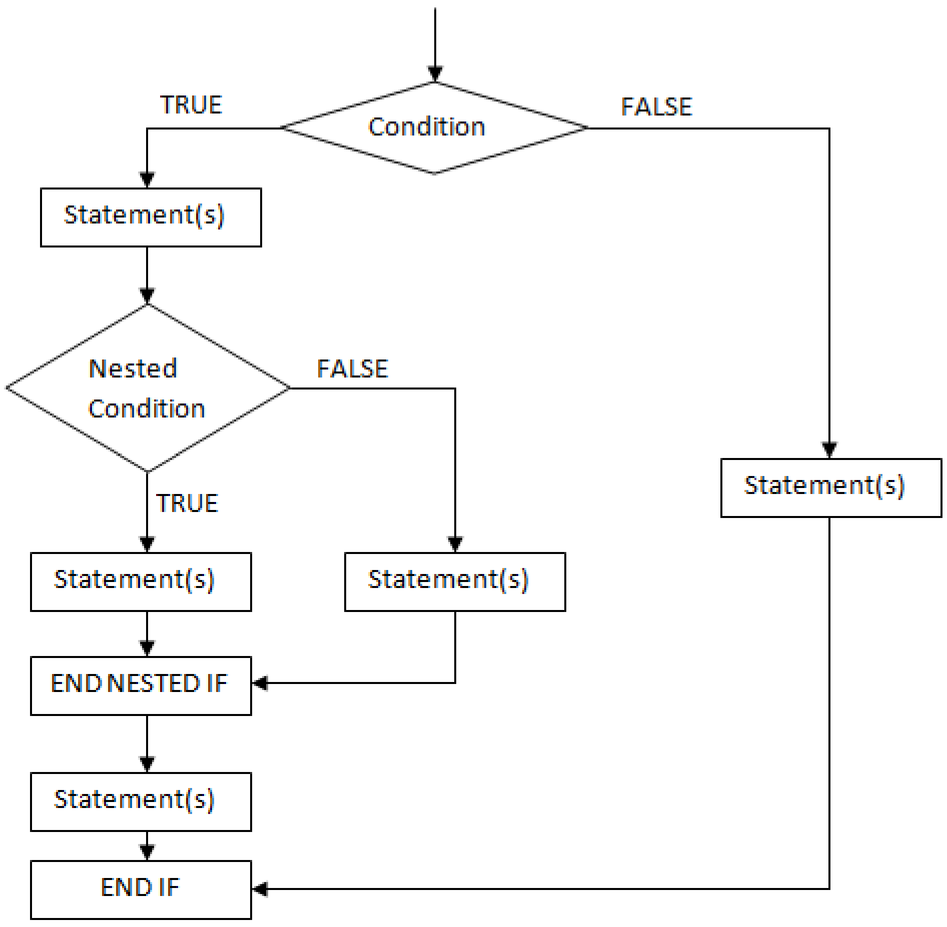
nested if (2 Marks)

When a if statement is kept inside another if statement, it is called nested if statement. Nested if statements are used if there is a sub condition to be tested. The depth of nested if statements depends upon the number of conditions to be checked.

Syntax:

```
if (condition 1)
{
   statements;
   if (sub condition 1)
   {
      statements;
   }
   statements;
}
else if (condition 2)
{
   statements;
   if (sub condition 2)
   {
      statements;
   }
   statements;
}
... ... ...
... ... ...
else
{
   statements;
   if (sub condition n)
   {
      statements;
   }
   statements;
}
```

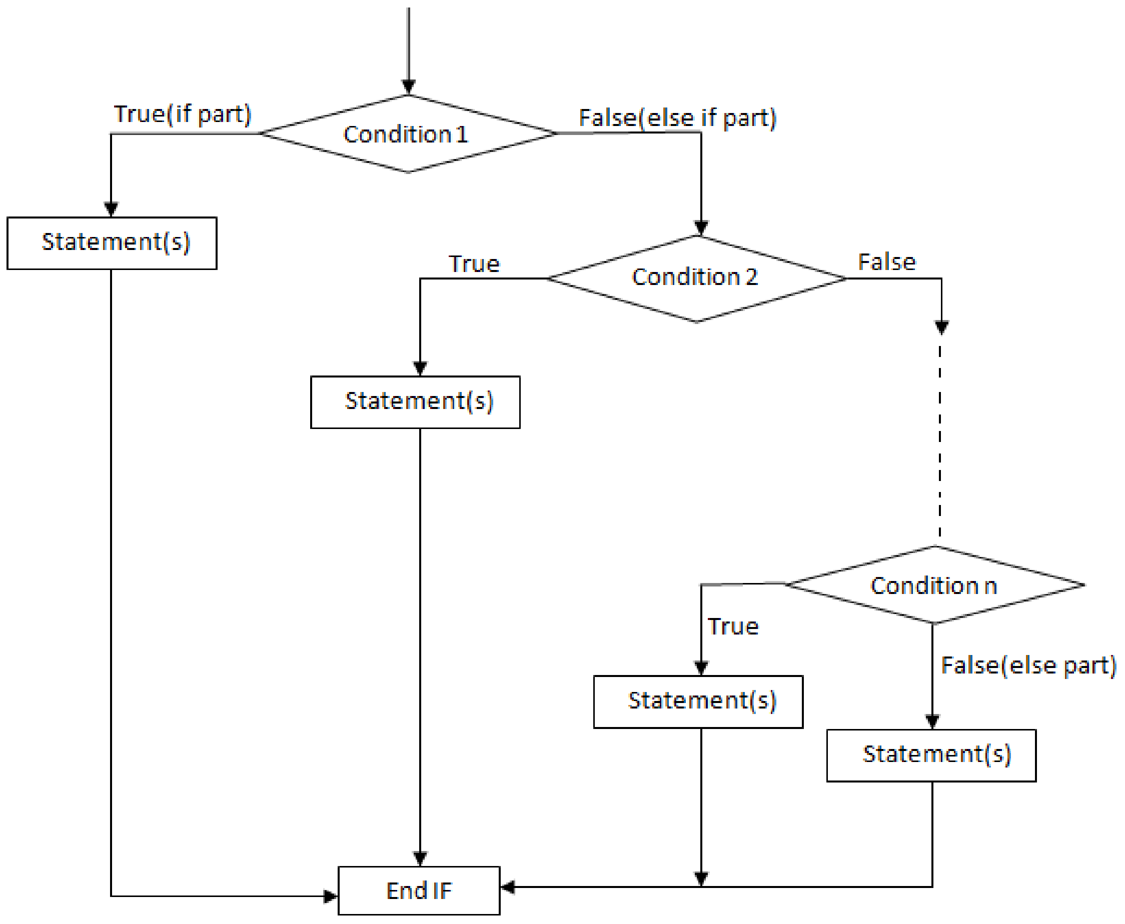
Flowchart:



ladder else if (3 Marks)

It is used when more than one condition is to be checked. A block of statement is enclosed inside if, else if and else part. Conditions are checked in each if and else if part. If the condition is true, the statements inside that block are executed. If none of the conditions are true, the statements inside else block are executed. A if … else if … else statement must have only one if block but can have as many else if block as required. Else part is optional and may be present or absent.

Flowchart:



Syntax:

```
if (condition 1)
{
   statements;
   … … …
}
else if (condition 2)
{
   statements;
   … … …
}
… … …
… … …
else if (condition n)
{
   statements;
   … … …
}
else
{
   statements;
   ... ... ...
}
```

5.a) Write a C program to input marks of five subjects Physics, Chemistry, Biology, Mathematics and Computer. Calculate sum and average percentage and grade according to following: (10 Marks)

Percentage >= 90% : Grade A
Percentage >= 80% : Grade B
Percentage >= 70% : Grade C
Percentage >= 60% : Grade D
Percentage >= 40% : Grade E
Percentage < 40% : Grade F

Program (6 Marks)

```
#include<stdio.h>
void main() //start of main method
{
  // how to read integer
 int m1,m2,m3,m4,m5;
 float total,per;
printf("Enter all subjects marks\n");
printf("Enter Physics Mark\n");
 scanf("%d",&m1);
printf("Enter Chemistry Mark\n");
 scanf("%d",&m2);
printf("Enter Biology Mark\n");
 scanf("%d",&m3);
printf("Enter Mathematics Mark\n");
 scanf("%d",&m4);
printf("Enter Computer Mark\n");
 scanf("%d",&m5);
//calculate total
 total=m1+m2+m3+m4+m5;
 //calculate percentage
 per=total/5;
printf("\nPercentage is=%.2f",per);
if(per>=90)
{
   printf("\nGrade-A");
}
if(per>=80 && per<90)
{
   printf("\nGrade-B");
}
if(per>=70 && per<80)
{
   printf("\nGrade-C");
}
if(per>=60 && per<70)
{
   printf("\nGrade-D");
}
if(per>=40 && per<60)
{
   printf("\nGrade-E");
}
if( per<40)
{
   printf("\nGrade-F");
}
}//end of main method
```

// For explanation 2 marks

Sample Output:(2 Marks)

```
Enter all subjects marks
Enter Physics Mark
23
Enter Chemistry Mark
21
Enter Biology Mark
25
Enter Mathematics Mark
26
Enter Computer Mark
24

Percentage is=23.80Grade-F
```

6.a) Write a C program to check given year is leap year or not. (5 Marks)
Program:

```
#include <stdio.h>
int main() {
  int year;
  printf("Enter a year: ");
  scanf("%d", &year);
  if (year % 400 == 0) {
    printf("%d is a leap year.", year);
  }
  else if (year % 100 == 0) {
    printf("%d is not a leap year.", year);
  }
  else if (year % 4 == 0) {
    printf("%d is a leap year.", year);
  }
  else {
    printf("%d is not a leap year.", year);
  }

  return 0;
}
```

For program and output (5 Marks)
Output:

```
Enter a year: 1990
1990 is not a leap year.
Enter a year: 1752
1752 is a leap year.
```

6.b) Write C program to check equality of 2 integer numbers without using relational operator (==,!=). (5 Marks)

For program and output (5 Marks)
Program

```
#include <stdio.h>
int main(void)
{
   int x = 10, y = 12;

if(!(x^y))
{
      printf ("x=%d is equal to y=%d\n", x, y);
   }
   else {
      printf ("x=%d is not equal to y=%d\n", x, y);
   }

   return 0;
}
```

Output:

```
x=10 is not equal to y=12
```

7.a) Explain 5 Generations of the computer development. (5 Marks)

For table (2 Marks)

Describe each generations in few lines (3 Marks)

7.b) Explain the following operators with example
a) Relational operators
b) Bitwise Operators.

Definition and list of relational operators (2.5 Marks)

<, <=, >, >= !=, ==

Definition and list of bitwise operators (2.5 Marks)
&, |, !

Generations of computers	**Generations timeline**	**Evolving hardware**
First generation	1940-1956	Vacuum tube based
Second generation	1956-1963	Transistor based
Third generation	1964-1971	Integrated circuit based
Fourth generation	1971-present	Microprocessor based
Fifth generation	The present and the future	Artificial intelligence based

8.a) Write c program to find the roots of the quadratic equation using else_if_ladder with suitable messages. (10 Marks)

For Source code 7 Marks

Output and Explanation 3 Marks

Program
```
#include<stdio.h>
#include<math.h>
void main()
{
  // how to read integer
 int a,b,c;
 float r1,r2,real,imag;
 int d;
 printf("Enter a,b,c");
 scanf("%d%d%d",&a,&b,&c);
 d=(b*b)-(4*a*c);
 if(d==0)
 {
    r1=-b/(2*a);
    printf("Root is%f",r1);
 }
 else if(d>0)
 {
    r1=(-b+sqrt(d))/(2*a);
     r2=(-b-sqrt(d))/(2*a);
     printf("Root is r1=%.2f\n",r1);
      printf("Root is r2=%.2f",r2);
 }
  else
  {
     real=-b/(2*a);
     imag=sqrt(-d)/(2*a);
      printf("root1 = %.2f+%.2fi \n", real, imag);
       printf(" root2 = %.2f-%.2fi", real, imag);
    }
}
```

Output:

```
Enter a,b,c
1
-1
-12
Root is r1=4.00
Root is r2=-3.00
Enter a,b,c
1
2
3
root1 = -1.00+1.41i
 root2 = -1.00-1.41i
```