Module 1

1) What is the difference between script and program?

→ A "program" in general, is a ~~gener~~ sequence of instructions written, so that a computer can perform a certain task.

A "script" is code written in a scripting language. A scripting language is nothing but a type of programming language in which we can write code to control another software application.

2) What are the basic components of a digital computer?

→ A typical digital computer system has four basic functional elements :
i) input-output equipment, ii) Main Memory, iii) Control Unit and iv) Arithmetic Logic Unit.

3) Difference between compiler and interpreter.

→

Compiler	Interpreter
i) Compiler scans the entire program and translates the whole of it into machine code at once.	i) Interpreter translates just one statement of the program at a time into machine code.
ii) A compiler takes a lot of time to analyze the source code.	ii) An interpreter takes very less time to analyze the source code.

4) Is RAM absolutely volatile?

→ Volatile Memory is the memory that can keep the information only during the time it is powered up. In other words, volatile memory requires power to maintain the information.
RAM is considered volatile because it loses all its content when it turns off.

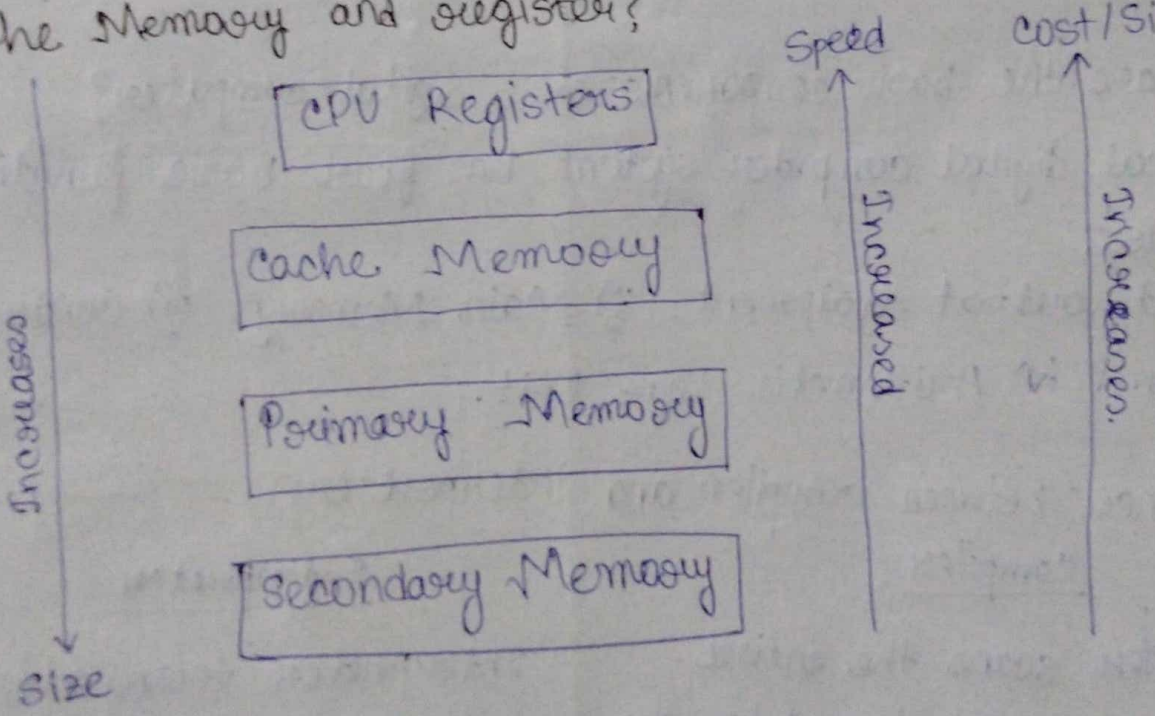
So, RAM is absolutely volatile.

5) What is the basic role of an Operating system?

→ An operating system is the most important software that runs on a computer. It manages the computer's memory and processes, as well as its software and hardware. It also allows you to communicate with the computer without even knowing how to speak the computer's language.

6) Differentiate between Secondary Memory, Primary memory Cache Memory and register?

→



Module 2

1) Convert the Hexadecimal number 9AC2 to the corresponding decimal number.

→ $(9AC2)_{16}$

A = 10
C = 12

$= (16^3 \times 9) + (16^2 \times A) + (16^1 \times C) + (16^0 \times 2)$

$= (16^3 \times 9) + (16^2 \times 10) + (16^1 \times 12) + (16^0 \times 2)$

$= 36864 + 2560 + 192 + 2$

$= (39618)_{10}$

2) Explain r's compliment and $(r-1)^{th}$'s compliment with example.

→ **r's compliment**

r's compliment = $r^m - N$, where r = base of the given Number
N = the given number
m = No. of digits in N

e.g → $(74)_{10}$

N = 74, r = 10, m = 2

∴ r's compliment of $(74)_{10} = 10^2 - 74 = 100 - 74 = (26)_{10}$

(r-1)'s compliment

(r-1)'s compliment = $r^m - N - 1$, where N = given number
r = base of the given number
m = No. of digits in N.

eg → $(23)_8$

N = 23, r = 8, m = 2

(r-1)'s compliment of $(23)_8 = 8^2 - 23 - 1 = 64 - 23 - 1$
$= (40)_8$

3) Subtract $(1011)_2$ from $(101)_2$ using 1's complement method.

→ ~~$(101)_2 - (1101)_2 = (111)_2$~~

Let, $P = (101)_2$ and $Q = (1101)_2$

1's complement of Q is $-(0010)_2$

$\therefore P + (-Q) = 0101$

$+0010$

$(0111)_2$

~~$\therefore (101)_2 - (1101)_2 = (111)_2$~~

As the result is coming negative, we have to take 1's complement of the final result.

$\therefore$ The final result is $(1000)_2$

Subtract $(101)_2$ from $(1011)_2$ using 1's complement method.

→ Let, $P = (1011)_2$ and $Q = (101)_2$

1's complement of Q is $-(010)_2$

$P + (-Q) = 1011$

$+0010$

1101

4. Subtract $(1011)_2$ from $(101)_2$ using 2's complement method.

→ Let, $P = (101)_2$ and $Q = (1011)_2$

1's complement of $Q = 0100$

2's complement of $Q = 0100 + 1 = (0101)_2$

$P + (-Q) = 0101$

$+0101$

1010

no extra carry bit —

2's complement of the result = 1's compliment + 1

$= 0101 + 1$

$= (0110)_2$

Subtract $(101)_2$ from $(1011)_2$ using 2's complement method.

→ Let, $P = (1011)_2$ and $Q = (101)_2$

1's complement of $Q = (010)_2$

2's " " $Q = 010 + 1 = (011)_2$

$P + (-Q) = 1011$

$+0010$

1101

no extra bit.

2's complement of the result = 1's complement + 1

$= 0010 + 1$

$= (0011)_2$

5) Express -7.5 in IEEE 754 single-precision format.

→ $(-7.5)_{10} = (-111.1)_2$

$= -1.111 \times 10^2 = -1.111 \times 2^2$

M = 111..... 0

E = 2

E' = E + 127 = 2 + 127 = 129 = $(10000001)_2$

S = 1

1	10000001	111.....0
S	E'	M

S = Sign bit

E' = 8 bit Exponent

M = 23 bit Mantissa.

6) Express 6.25 in IEEE 754 double-precision format.

→ $(6.25)_{10} = (110.01)_2$

$= 1.1001 \times 2^2$

M = 1001..... 0

E = 2

E' = E + 1023 = 2 + 1023 = 1025 = $(10000000001)_2$

S = 1

1	10000000001	1001.....0
S	E'	M

S = Sign bit

E' = 11 bit Exponent

M = 52 bit Mantissa

7) What are the drawbacks of signed magnitude representation and signed 1's complement representation?

→ Drawbacks of Signed Magnitude Representation

i) Has the problem of double representing the 0 (0 and +0).

ii) Complicates the design of the logic circuits that handle signed-numbers arithmetics.

iii) Each of the sign and magnitude parts has to be processed separately.

Drawbacks of Signed 1's Complement Representation

i) 1's complement notation is not very simple to understand because it is very much different from the conventional way of representing signed numbers.

ii) The other disadvantage is that there are two notations for 0 (0000 and 1111), which is very inconvenient when the computer wants to test for a 0 result.

8) In IEEE 754 single and double precision format how 0, +infinity and -infinity are represented?

→

9) Subtract $(85)_{10}$ from $(27)_{10}$ using both 10's complement and 9's complement.

→ $(85)_{10}$ —

10's complement $= r^m - N = 10^2 - 85 = (15)_{10}$

9's complement $= r^m - N - 1 = 10^2 - 85 - 1 = (14)_{10}$

$(27)_{10}$ —

10's complement $= r^m - N = 10^2 - 27 = (73)_{10}$

9's complement $= r^m - N - 1 = 10^2 - 27 - 1 = (72)_{10}$

10) Convert decimal number 17.75 to the corresponding binary number.

→

32	16	8	4	2	1
1	0	0	0	1	

$0.75 \times 2 = 1.50 \rightarrow 1$ ↓

$0.50 \times 2 = 1.00 \rightarrow 1$ ↓

$\therefore (17.75)_{10} = (10001.11)_2$

11) A floating-point number system uses 16-bit bits to represent a number. MSB is the sign bit. The least significant bits are mantissa and the remaining 6 bits are exponent in excess 31 formats.

i) Represent -1.5×10^{-2} in this format.

→ $-1.5 \times 10^{-2} = -150$ sign bit = 1

$150 = 10010110 = 1.0010110 \times 2^7 = 1.00101100$

$E = 7$, $E' = E + 31 = 38 = 100110$

1	100110	1.0010 11000
S	6 bit Exponent	9 bit Mantissa

ii) What is the value represented by 1 001110 11000000?

→ $E' = 001110 = 14$ $\therefore E = E' - 31 = 14 - 31 = -17$

$M = 1.110000\ldots 0$

$N = M \times 2^E$

$N = 1.11000000..0 = 1.11 \times 2^{-17}$

$2.35 \times 10^{-19} = 0.0000\ldots 235$

$N = 0.$

12) 7's complement of 375 will be —

→ 7's complement of $(375)_8$ is $= r^m - N - 1$

$= 8^3 - 375 - 1$

$= (136)_8$

13) 8's complement of 375 will be —

→ 8's complement of $(375)_8$ is $= r^m - N = 8^3 - 375$

$= (137)_8$

14) 8's complement of 370 will be —

→ 8's complement of $(370)_8$ will be $= r^m - N = 8^3 - 370$

$= (142)_8$

15) What will be the value of x, if $x-1$ is 377 in octal?

→

Module 3 and 4

1) State and Prove De-Morgan's Theorem.

→ **De Morgan's Theorem**

The complement of the sum of two or more variables is equal to the product of the complement of the variables.

The complement of the product of two or more variables is equal to the sum of complements of variables.

Prove

$A=0 \quad B=1$

$\overline{A+B} = \overline{0+1} = \overline{1} = 0$

$\bar{A}.\bar{B} = \bar{0}.\bar{1} = 1.0 = 0$

Hence $\overline{A+B} = \bar{A}.\bar{B}$

2) What do you mean by Universal gate?

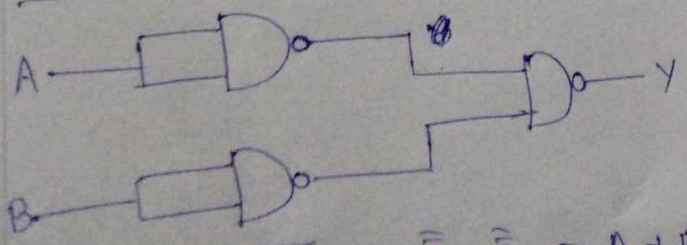
→ A Universal gate is a gate which can implement any boolean function without need to use any other gate type.

NAND and NOR gates are universal gates.

3) Explain why NAND and NOR gates are known as universal gates.

→ Because they can be combined to produce any basic logic gates like OR, AND and NOT.

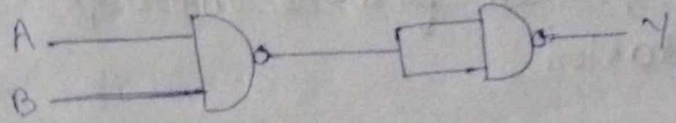
NAND → OR



$Y = \overline{\bar{A}.\bar{B}} = \bar{\bar{A}} + \bar{\bar{B}} = A + B$

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

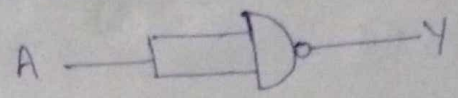
NAND → AND



$Y = \overline{\overline{A \cdot B}} = AB$

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

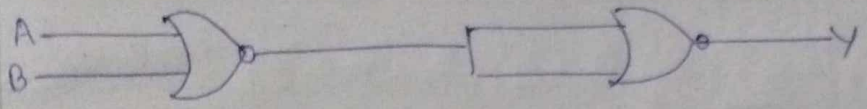
NAND → NOT



$Y = \overline{A}$

A	Y
0	1
1	0

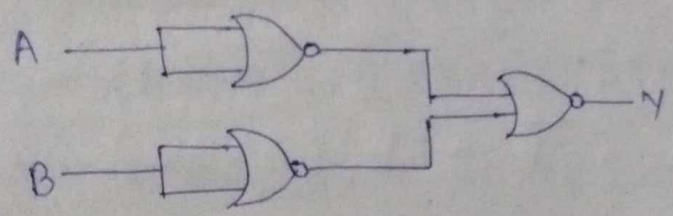
NOR → OR



$Y = \overline{\overline{A+B}} = A+B$

A	B	Y
0	0	0
0	1	1
1	0	1
1	1	1

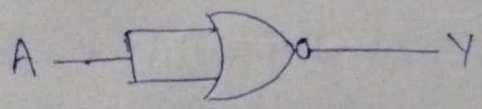
NOR → AND



$Y = \overline{A} \cdot \overline{B} = A \cdot B$

A	B	Y
0	0	0
1	0	0
0	1	0
1	1	1

NOR → NOT



$Y = \overline{A}$

A	B	Y
0	0	1
1	1	0

For a 4 variable Boolean function if the Min term representation is $\Sigma 2, 5, 6, 7$ then find the corresponding Max term representation.

5) Find the Min Term representation and the max term representation of the Boolean function $f(A,B,C) = AB+BC+CA$.

→

A	B	C	$Y_1 = AB$	$Y_2 = BC$	$Y_3 = CA$	$Y = Y_1 + Y_2 + Y_3$
0	0	0	0	0	0	0
0	0	1	0	0	0	0
0	1	0	0	0	0	0
0	1	1	0	1	0	1
1	0	0	0	0	0	0
1	0	1	0	0	1	1
1	1	0	1	0	0	1
1	1	1	1	1	1	1

Min terms are — Σ 3, 5, 6, 7

$\therefore Y' = (A+\bar{B}+\bar{C})(\bar{A}+B+\bar{C})(\bar{A}+\bar{B}+C)(\bar{A}+\bar{B}+\bar{C})$

Max terms are — Π 0, 1, 2, 4

6) What is a Truth Table?

→ The truth table of a logic system describes the output(s) of the system for given input(s). The input(s) and output(s) are used to label the columns of a truth table, with the rows representing all possible inputs to the circuit and the corresponding outputs.

7) For the N variable Boolean Function, what is the total number of rows/tuples in a truth Table? How many literals are possible?

→ For the N variable Boolean function —

Total number of rows in the truth table = 2^N

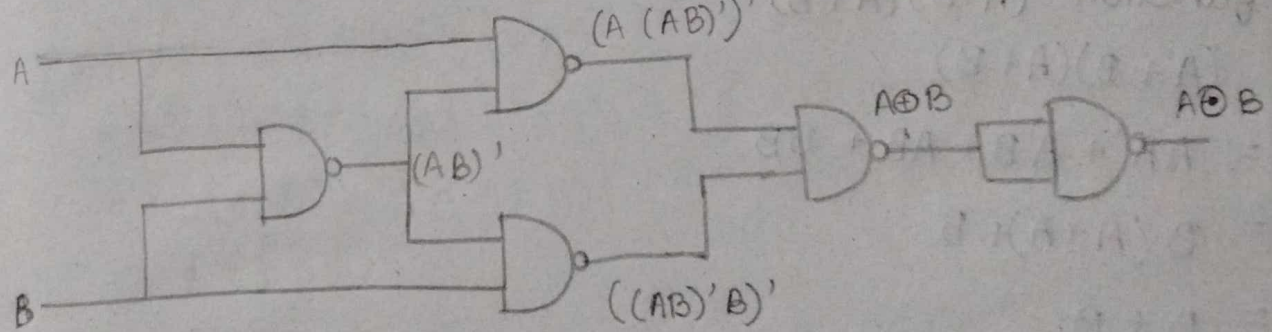
Possible literals are =

8) For 1024 literals how many variables are there?

→

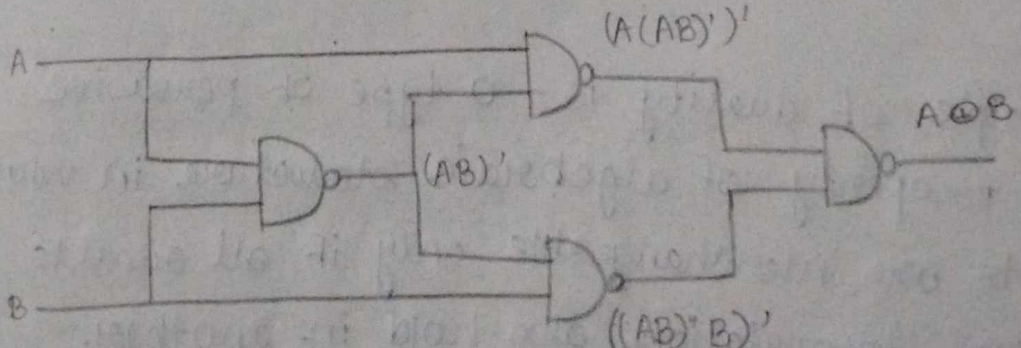
9) Design XNOR gate using only NAND gate.

→

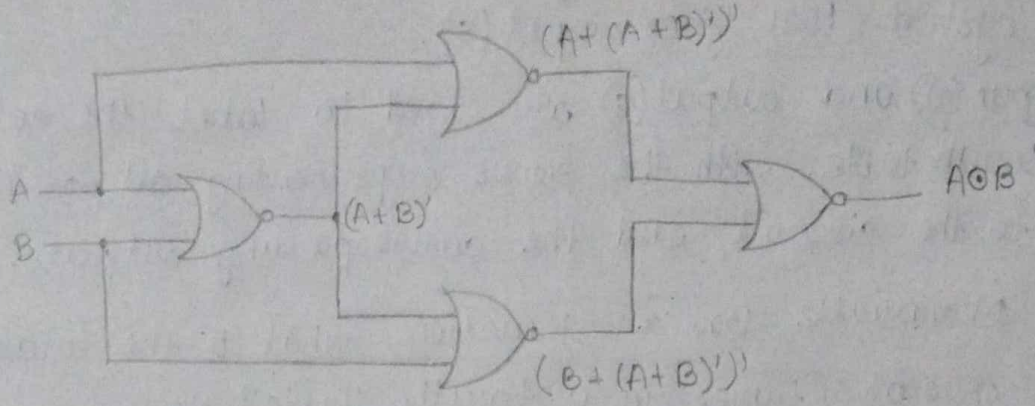


10) Design XOR gate using only ~~nand~~ NAND gate.

→

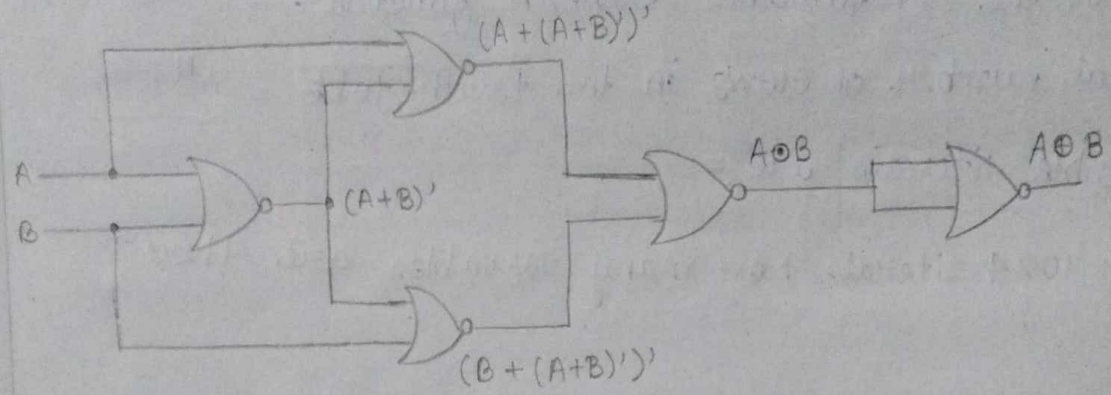


Design XNOR gate using only NOR gate.



2) Design XOR gate using only NOR gate.

→



13) What will be the answer if you reduced logic gate function (A'+B)(A+B)

→ (A'+B)(A+B)

= AA' + A'B + AB + BB

= B(A+A') + B

= B + B

= B

14) What is the principle of duality related to Boolean Algebra?

→ The principle of duality is - a type of pervasive property property of algebraic structure in which two concepts are interchangeable only if all results held in one formulation also hold in another. The concept is known as dual formulation.

15) Explain absorption law and consensus theorem in Boolean Algebra. Obtain the alternative expression for both of these using the principle of duality.

→ Absorption Law

this law states that —

i) $X + XY = X$ and ii) $X(X+Y) = X$

X	Y	XY	X+XY
0	0	0	0
0	1	0	0
1	0	0	1
1	1	1	1

X	Y	X+Y	X(X+Y)
0	0	0	0
0	1	1	0
1	0	1	1
1	1	1	1

Consensus Theorem

This law states that —

$AB + A'C + BC = AB + A'C$

$\therefore Y = AB + A'C + BC$

$= AB + A'C + BC.1$

$= AB + A'C + BC(A + A')$

$= AB + A'C + ABC + A'BC$

$= AB(1 + C) + A'C(1 + B)$

$= AB + A'C$

Module 5

1) Differentiate between algorithm, program and flowchart. Give a suitable example to explain your answer.

→ **Algorithm**

An algorithm is a set of instructions for solving a problem or accomplishing a task.

Example → One common example of an algorithm is a recipe, which consists of specific instructions for preparing a dish or meal.

Recipe for Making Potato Bondas –

Ingredients :

Potatoes 250 grams, chopped onions 3 (small), finely chopped green chillies 5, gram flour 100 gram, oil for frying, water for batter, salt 2 teaspoons.

Method :

Step 1 – Boil potatoes till cooked, peel and mash them until they are soft.

Step 2 – Mix fried onions, green chillies and salt with the mashed potatoes.

Step 3 – Take little portions of the mixture and make small balls.

Step 4 – Mix gram flour, water and a little salt and beat well till a smooth and creamy batter is obtained.

Step 5 – Dip the potato balls in the batters. Take out and deep fry in oil on a low fire.

Step 6 – Take out when the balls are fried to a golden brown colour.

Result :

A dozen potato bondas ready to be served hot with tomato sauce.

Program

A program is a set of instructions that a computer follows in order to perform a particular task.

Example →

Input :

~~a = int(input("Enter~~

a = 10

b = 15

s = a + b

print (s)

Output :

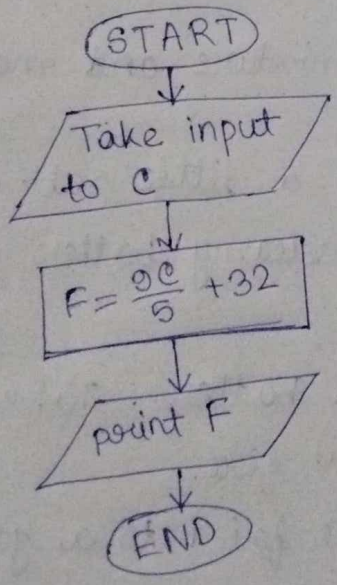
25

Flowchart

Flowchart is a diagram that shows step-by-step progression through a procedure or system especially using connecting lines and a set of conventional symbols.

Example →

Flowchart to convert Celsius temparature to Farenheit temperature –



$$\frac{C}{5} = \frac{F-32}{9}$$

$$\Rightarrow 9C = 5(F-32)$$

$$\Rightarrow F-32 = \frac{9C}{5}$$

$$\Rightarrow F = \frac{9C}{5} + 32$$

2) Define algorithm and its properties.

→ An algorithm is a set of instructions for solving a problem or accomplishing a task.

Properties →

i) An algorithm begins with instruction(s) to accept inputs. These inputs are processed by the subsequent sequence of intructions in the algorithm.

ii) The sequence of instructions specified in the algorithm must be precise and unambiguous.

iii) Each instruction must be sufficiently basic such that it can, it principle, be carried out by a person with paper and pencil.

iv) The total time to carry out all the steps in the algorithm must be finite. As algorithms may contain instructions to repetitively carry out a group of instructions, this requirement implies that the number of repetitions must be finite.

v) An algorithm must produce one or more outputs.

3) Define Time Complexity and Space Complexity with example.

→ • Time Complexity

Time complexity is a concept in computer science that deals with the quantification of the amount of time taken by a set of code or algorithm to process or run as a function of the amount of input.

In other words, time complexity is essentially efficiency, or how long a program function takes to process a given input.

Examples:

$O(n^2)$, $O(n)$, $O(\log n)$

• Space Complexity

Space complexity is the total amount of memory space used by an algorithm/program including the space of input values for execution.

So to find space-complexity, it is enough to calculate the space occupied by the variables used in an algorithm/program.

Example :

O(n)

Module 7

1) Difference between 5/2 with 5//2.

→ 5/2 ⇒ it divides 5 by 2. The output will be 2.5.

5//2 ⇒ it gives the floor value from the division. The result is 2.

2) Difference b/w relational operators with logical operators.

→ Relational operators compare values and return either TRUE or FALSE. Logical operators perform logical operations on TRUE and FALSE. Value used with a logical operator are converted into booleans prior to being evaluated.

Module 8

1) Differentiate break, continue and pass with examples

→ **Break**

The break statement in Python terminates the current loop and resumes execution at the next statement.

Input

```
x=10
while (x>0):
    print("x = ", x)
    x = x-1
    if (x==5):
        break
```

Output

```
x = 10
x = 9
x = 8
x = 7
x = 6
```

Continue

The continue statement in Python returns the control to the beginning of the while loop. The continue statement rejects all the remaining statements in the current iteration of the loop and moves the control back to the top of the loop.

Input

```
x = 10
while (x > 0):
    if (x == 5):
        continue
    print ("x = ", x)
    x = x - 1
```

Output

```
x = 10
x = 9
x = 8
x = 7
x = 6
```

Pass

The pass statement in Python is used when a statement is required syntactically but you do not want any command or code to execute.

Input

```
x = 'nivakshi'
for letter in 'nivakshi':
    if(letter == 'h'):
        pass
    print('name', letter)
print('nivakshi')
```

Output

```
name n
name i
name v
name a
name k
name s
name h
name i
nivakshi
```

3) What is if-else statement in python? Give a suitable example.

→ The if-else statement is used to execute both the true part and the false part of a given condition. If the condition is true, the if block code is executed and if the condition is false, the else block code is execute.

Syntax

```
if(condition):
#Executes this block if the condition is true
else:
#Executes this block in the condition is false
```

Example:

Input

```
n=20
if(i%.2==0):
    print("n is an even number")
else:
    print("n is an odd number")
```

Output

n is an even number.

4) Differentiate b/w for and while loop with examples.

→ • For loop allows a programmer to execute a sequence of statements several times, it abbreviates the code which helps to manage loop variables.

• While loop ~~always~~ allows a programmer to repeat a single statement or a group of statements for the TRUE conditions.

	FOR	WHILE
Declaration	for (initialization ; condition; iteration) {// body of 'for' loop}	while (condition) {statements ; //body of loop}
Format	It allows initialization, condition checking and iteration statements are	It allows initialization and condition checking on top of the loop.

	written on the top of the loop.	
Condition	It iterates infinite time, if no condition is given.	It shows an error if the no condition is given.
Initialization	If initialization is once done if "for" loop, it never be repeated.	If initialization is done while condition checking then it is required each time when the loop iterates itself.
Iteration Statement	As iteration is statement is written at the top, it will execute only after all the statements will be executed.	The iteration statement can be placed anywhere in the syntax of the loop.

Example:

'for' loop

```
n = int(input("Enter a no. : "))
r=0
for i in range (1, n+1):
    if (i % 2==0):
        r=r+i
```

'while' loop

```
n= int(input("Enter a no. : "))
i=1
sum=0
while (i<=n):
    if (i%2==0):
        sum = sum + i
    i=i+2
print(sum)
```

1) What is the difference b/w script and program?

→

Script	Program
• Conversion from HLL to MLL is done using interpreter.	• The conversion from HLL to MLL is done using compiler.
• Execution time is more than program.	• Since compiler checks the entire program at once, the execution time is less.

2) Is RAM completely volatile?

→ RAM is partially volatile. In RAM, a program known as Boot-strap loader is present which isn't deleted once the system is powered off. So, the role of bootstrap loader is to initiate transfer of Operating System from secondary memory to primary memory.

3) XOR using NAND.

→

XOR →

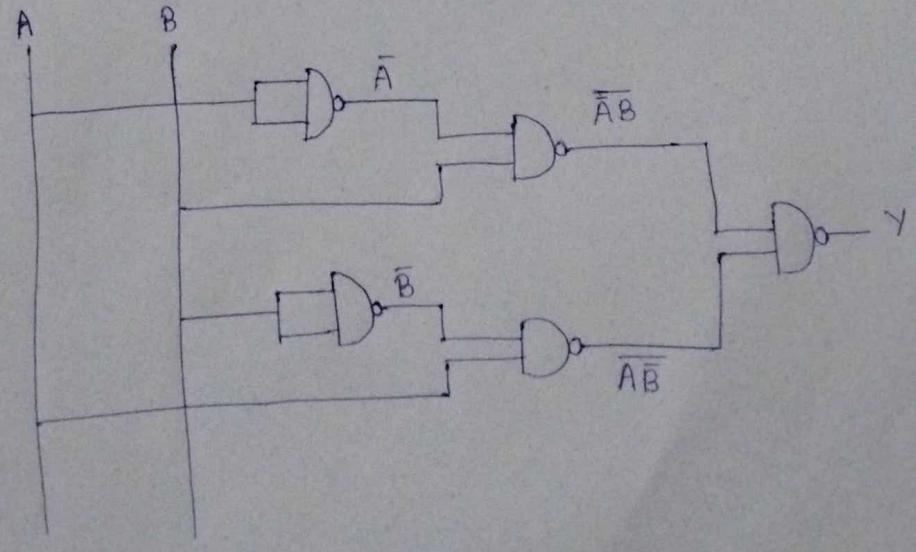
A	B	Y
0	0	0
0	1	1
1	0	1
1	1	0

NAND →

A	B	Y
0	0	1
0	1	1
1	0	1
1	1	0

$Y = \bar{A}B + A\bar{B}$

$Y = \bar{A}\bar{B} + \bar{A}B + A\bar{B}$



Module 2

5) -14.75 in IEEE single precision.

→ | S | Biased Exponent | 1. (Hidden) | 23 bit Mantissa |

8 bit
Excess 127

14 = 1110 14.75 = 1110.11

$= 1.11011 \times 2^3$ → Exponent

Mantissa

$= 1.11011000\ldots0 \times 2^3$

Mantissa

E = 3

∴ E' = 3 + 127 = 130 = 10000010

| 1 | 10000010 | 1.(Hidden 1) 1101100....0 |

8bit exponent 23 bit Mantissa.

6) 27.625 into IEEE 754 Double precision format.

→

S	Exponent	Mantissa
1 bit	11 bit	52 bits.
	Excess 1023 bit	

$27.625 = 11011.101 = 1.1011101 \times 2^4$

$= 1.10111010\ldots0 \times 2^4 \rightarrow E$

Mantissa

$E' = 4$

$E = 4 + 1023 = 1027 = 10000000011$

Hidden 1

0	10000000011	1.10111010....0
Sign bit	Exponent	Mantissa.

Module 1

6) Speed (S.M) ≤ Speed (P.M) ≤ Speed (C.M) ≤ Speed (CPU Registers)

Size (CPU Registers) ≤ Size (C.M) ≤ Size (P.M) ≤ Size (S.M.)

Cost (S.M) ≤ Cost (P.M) ≤ Cost (C.M) ≤ Cost (CPU Registers)

Module 2

15) $x - 1 = (377)_8$

$\Rightarrow x = (377)_8 + (1)_8$

$= (377)_8 + (1)_8$

$(400)_8$ (Ans.)

$(7)_8 = (7)_{10}$

$(1)_8 = (1)_{10}$

$(8)_{10}$

$= (10)_8$

8 | 8
8 | 1 — 0 ↑
0 — 1

$(377)_8 = 3 \times 8^2 + 7 \times 8^1 + 7 \times 8^0 = 192 + 56 + 7$

$= (255)_{10}$

$(1)_8 = (1)_{10}$

$x - 1_0 = (255)_{10}$

$\Rightarrow x = 255 + 1 = (256)_{10}$

$8 \mid 256$
$8 \mid 32 \quad - 0 \uparrow$
$8 \mid 4 \quad - 0 \;|$
$0 \quad - 4 \;|$

$= (400)_8$ (Ans.)

16) $x + 1 = (2340)_5$

$\Rightarrow x = (2340)_5 - (1)_5 = (2334)_4$

$(2340)_5 = 2 \times 5^3 + 3 \times 5^2 + 4 \times 5^1 + 0 \times 5^0$

$= 250 + 75 + 20$

$= (345)_{10}$

06 Sunday $\therefore x = (345)_{10} - 1 = (344)_{10}$

$5 \mid 344$
$5 \mid 68 \quad R = 4 \uparrow$
$5 \mid 13 \quad R = 3$
$5 \mid 2 \quad R = 3$
$0 \quad R = 2$

$(2334)_5$

Module 3 & 4

A) $\Sigma 2, 5, 6, 7$ → 4 variables.
↳ SOP

x_3	x_2	x_1	x_0	f
0	0	0	0	0
0	0	0	1	0
0	0	1	0	1
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
				:

Minterms are the input combination for which output is 1.

$\Sigma 2,5,6,7 = \bar{x}_3 \bar{x}_2 x_1 \bar{x}_0 + \bar{x}_3 x_2 \bar{x}_1 x_0 + \bar{x}_3 x_2 x_1 \bar{x}_0 + \bar{x}_3 x_2 x_1 x_0$
(2) (5) (6) (7)

$\pi(0,1,3,4,8,9,10,11,12,13,14,15)$

$= (x_3+x_2+x_1+x_0)(x_3+x_2+x_1+\bar{x_0})$ *(

Module 8

2) L1 = [[1, 2] [3, 4] [5, 6], [7, 8]]

L2 = L1

print (L2) => [[1, 2], [3, 4], [5, 6], [7, 8]]

print(L1) => "

L1[2][0] = 500

print (L1) => [[1, 2], [3, 4], [500, 6], [7, 8]]

print (L2) => "

Import copy as copy

L1 = [[1, 2], [3, 4], [5, 6], [7, 8]]

L3 = copy.copy(L1) [Shallow copy]

L4 = copy.deepcopy(L1) [Deep Copy]

L3[3][0] = 400

print(L1) => [[1,2],[3,4],[5,6],[400,8]]

print(L3) => [[1,2],[3,4],[5,6],[400,8]]

L4[0][0] = 700

print(L1) =>

print(L4) => [[1,2],[700,4],[5,6],[7,8]]

```
x = 5
y = 3
p = x + y
a = 3.5
b = 4
c = a + b
print(p)        => 8
print(c)        => 7.5
x = float(x+y)  Explicit type
print(x)
```

```
Import array as arr.
L = [5, 7, 8, 9]
for i in range (0, len(L)):
    L[i] = float (L[i])
a = arr. array(L)
Import array as np.
L = [1, 2, 3, 4]
a = np. array (L)
b = a. astype (float)
```

Module 9

A) python 3 prog1.py 10 30

argu[0] argu[1] argu[2]

```
x = int( sys. argu [1])
y = int (sys. argu [2])
z = x + y
print (z)
```