# COMPUTER SCIENCE

## PAPER 1

## (THEORY)

***Three hours***



*(Candidates are allowed additional 15 minutes for **only** reading the paper. They must NOT start writing during this time.)*

---

*Answer **all** questions in Part I (compulsory) and **seven** questions from Part-II, choosing **three** questions from Section-A, **two** from Section-B and **two** from Section-C.*

*All working, including rough work, should be done on the same sheet as the rest of the answer.*

*The intended marks for questions or parts of questions are given in brackets [ ].*

---

## PART I

*Answer **all** questions.*

***While answering questions in this Part, indicate briefly your working and reasoning, wherever required.***

**Question 1**

(a) Simplify: $(A + C)\cdot(A + A\cdot D) + A\cdot C + C$ **[2]**

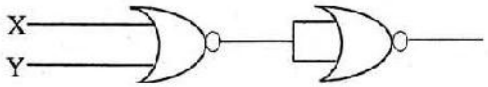
(b) Draw a logic circuit for $(A + B)\cdot(C + D)\cdot C$ **[2]**

(c) Verify the following proposition with the help of a truth table: **[2]**

$$P \vee (\sim P \wedge Q) = P \vee Q$$

(d) State De Morgan's law and verify it, using a truth table. **[2]**

(e) Answer the questions related to the circuit given below: **[2]**




(i) Give the output if, X=1 and Y=0

(ii) Name the basic gate represented by the above diagram.

---

**Question 13**

(a) A linked list is formed from the objects of the class: **[4]**

```
class Nodes
{
        int num;
        Nodes next;
}
```

Write an *Algorithm* **OR** a *Method* to print the sum of nodes that contains only odd integers of an existing linked list.

The method declaration is as follows:

**void NodesCount( Nodes starPtr )**

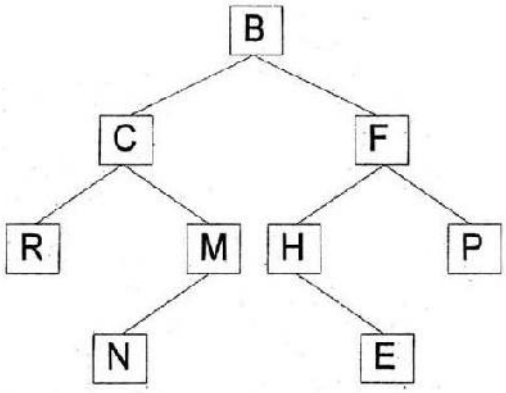
(b) (i) Give the meaning of the following common expression in Big O notation: **[1]**

O(N)

$O(N^2)$

(ii) List *any two* cases to analyse algorithm complexities. **[1]**

(c) Answer the following questions from the diagram of a Binary Tree given below:




(i) Name the leaf nodes of the right sub-tree. **[1]**

(ii) Write postorder traversal of the left sub-tree of node B including itself. **[1]**

(iii) State the level number of nodes R and M when the root is at level 0. **[1]**

(iv) Name the internal nodes of the tree. **[1]**

**Question 2**

(a) Define *computational complexity*. Calculate the complexity using Big 'O' notation for the following code segment: **[2]**

```
for(int k=0;k<n;k++)
    s+=k;
```

(b) Convert the following infix notation into postfix form: **[2]**

X + ( Y – Z ) + ( ( W + E ) * F ) / J

(c) Differentiate between *this keyword* and *super keyword*. **[2]**

(d) The array D[-2...10][3...8] contains double type elements. If the base address is 4110, find the address of D[4][5], when the array is stored in **Column Major Wise**. **[2]**

(e) State *any two* characteristics of a Binary tree. **[2]**

**Question 3**

(a) The following function is a part of some class. Assume 'x' and 'y' are positive integers, greater than 0. Answer the given questions along with dry run / working.

```
void someFun(int x, int y)
{
    if(x>1)
    {   if(x%y==0)
        {   System.out.print(y+" ");
            someFun(x/y, y);
        }
        else
            someFun(x, y+1);
    }
}
```

(i) What will be returned by **someFun(24,2)** ? **[2]**

(ii) What will be returned by **someFun(84,2)** ? **[2]**

(iii) State in one line what does the function **someFun( )** do, apart from recursion? **[1]**

(b) The following is a function of some class which checks if a positive integer is an Armstrong number by returning true or false. (*A number is said to be Armstrong if the sum of the cubes of all its digits is equal to the original number.*) The function does not use modulus (%) operator to extract digit. There are some places in the code marked by **?1?, ?2?, ?3?, ?4?, ?5?** which may be replaced by a statement / expression so that the function works properly.

```
boolean  ArmstrongNum( int N )
{
        int sum= ?1?;
        int num=N;
        while( num>0)
        {
           int f= num/10;
           int s =  ?2?;
           int digit = num−s;
           sum+=  ?3?;
           num = ?4?;
        }
        if(  ?5? )
           return true;
        else
           return false;
}
```

(i) What is the statement or expression at **?1?** [1]

(ii) What is the statement or expression at **?2?** [1]

(iii) What is the statement or expression at **?3?** [1]

(iv) What is the statement or expression at **?4?** [1]

(v) What is the statement or expression at **?5?** [1]

## PART – II

*Answer **seven** questions in this part, choosing **three** questions from Section A, **two** from Section B and **two** from Section C.*

## SECTION - A

*Answer any **three** questions.*

**Question 4**

(a) Given the Boolean function **F(A, B, C, D) = π (0,1,2,3,5,7,8,9,10,11).**

(i) Reduce the above expression by using 4-variable Karnaugh map, showing the various groups (i.e. octal, quads and pairs). [4]

(ii) Draw the logic gate diagram for the reduced expression. Assume that the variables and their complements are available as inputs. [1]

(b) Given the Boolean function:

**P(A, B, C, D) = ABC'D' + A'BC'D' + A'BC'D + ABC'D + A'BCD + ABCD**

(i) Reduce the above expression by using 4-variable Karnaugh map, showing the various groups (i.e. octal, quads and pairs). [4]

(ii) Draw the logic gate diagram for the reduced expression. Assume that the variables and their complements are available as inputs. [1]

**Question 5**

A person is allowed to travel in a reserved coach of the train, if he/she satisfies the criteria given below: **[10]**

- The person has a valid reservation ticket and a valid ID proof.

**OR**

- The person does not have a valid reservation ticket, but holds a valid pass issued by the Railway department with a valid ID proof.

**OR**

- The person is a disabled person and holds a valid pass issued by the Railway department along with a valid ID proof.

The inputs are:

| INPUTS | |
|---|---|
| **R** | The person has a valid reservation ticket. |
| **P** | The person holds a valid pass issued by the Railway department. |
| **D** | The person has a valid ID proof. |
| **H** | The person is a disabled person. |

(In all the above cases 1 indicates yes and 0 indicates no).

Output : T – Denotes allowed to travel (1 indicates yes and 0 indicates no in all the cases)

(a) Draw the truth table for the inputs and outputs given above and write the **POS** expression for **T(R, P, D, H).** **[5]**

(b) Reduce **T(R, P, D, H)** using Karnaugh map. **[5]**

Draw the logic gate diagram for the reduced **POS** expression for **T(R, P, D, H)** using only **NOR** gates. You may use gates with two or more inputs. Assume that the variable and their complements are available as inputs.

**Question 6**

(a) Draw the truth table and logic gate diagram for an Octal to Binary encoder. **[4]**

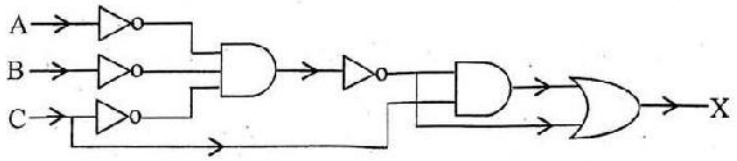
(b) What is a Multiplexer? State an application of a Multiplexer. Also, draw the logic diagram of a 4:1 Multiplexer. **[4]**

(c) Verify the following expression using Boolean laws. Also, mention the law used at each step of simplification. **[2]**

$$X \cdot Y \cdot Z + X \cdot Y' \cdot Z + X \cdot Y \cdot Z' = X \cdot ( Y + Z )$$

**Question 7**

(a) Derive a Boolean expression for the logic circuit given below and reduce the derived expression, using Boolean laws: **[3]**




(b) What are universal gates? Construct a logic circuit using NAND gates only for the expression: $A \cdot (B + C)$ **[3]**

(c) Define *Half Adders*. Draw the circuit diagram and the truth table for a Half Adder. **[4]**

## SECTION – B

*Answer any two questions.*

*Each program should be written in such a way that it clearly depicts the logic of the problem.*

*This can be achieved by using mnemonic names and comments in the program.*

(Flowcharts and Algorithms are **not** required.)

**The programs must be written in Java.**

**Question 8**

A class **Admission** contains the admission numbers of 100 students. Some of the data members / member functions are given below: **[10]**

| | | |
|---|---|---|
| **Class name** | : | **Admission** |
| **Data member/instance variable:** | | |
| Adno[ ] | : | integer array to store admission numbers |
| **Member functions/methods:** | | |
| Admission( ) | : | constructor to initialize the array elements |
| void fillArray( ) | : | to accept the elements of the array in ascending order |
| int binSearch(int l, int u, int v) | : | to search for a particular admission number (v) using **binary search** and **recursive technique** and returns 1 if found otherwise returns -1 |

Specify the class **Admission** giving details of the **constructor, void fillArray( )** and **int binSearch(int, int, int)** . Define the **main( )** function to create an object and call the functions accordingly to enable the task.

**Question 9**

A class **Merger** concatenates two positive integers that are greater than 0 and produces a new merged integer. **[10]**

**Example**: If the first number is **23** and the second is **764**, then the concatenated number will be **23764**.

Some of the members of the class are given below:

| | | |
|---|---|---|
| **Class name** | : | **Merger** |
| **Data members/instance variables:** | | |
| n1 | : | long integer to store first number |
| n2 | : | long integer to store second number |
| mergNum | : | long integer to store the merged number |
| **Member functions:** | | |
| Merger( ) | : | constructor to initialize the data members |
| void readNum( ) | : | to accept the values of the data members n1 and n2 |
| void JoinNum( ) | : | to concatenate the numbers n1 and n2 and store it in mergNum |
| void show( ) | : | to display the original numbers and the merged number with appropriate messages |

Specify the class **Merger**, giving the details of **the constructor, void readNum( ), void JoinNum( )** and **void show( )**. Define the **main( )** function to create an object and call the functions accordingly to enable the task.

**Question 10**

A class **TheString** accepts a string of a maximum of 100 characters with only one blank space between the words. **[10]**

Some of the members of the class are as follows:

| | | |
|---|---|---|
| **Class name** | : | **TheString** |
| **Data member/instance variable:** | | |
| str | : | to store a string |
| len | : | integer to store the length of the string |
| wordcount | : | integer to store the number of words |
| cons | : | integer to store the number of consonants |
| **Member functions/methods:** | | |
| TheString( ) | : | default constructor to initialize the data members |
| TheString( String ds) | : | parameterized constructor to assign str=ds |

| | | |
|---|---|---|
| void countFreq( ) | : | to count the number of words and the number of consonants and store them in wordcount and cons respectively |
| void Display( ) | : | to display the original string, along with the number of words and the number of consonants |

Specify the class **TheString** giving the details of the **constructors, void countFreq( )** and **void Display( )**. Define the **main( )** function to create an object and call the functions accordingly to enable the task.

## SECTION – C

*Answer any **two** questions.*

*Each program should be written in such a way that it clearly depicts the logic of the problem stepwise.*

*This can be achieved by using comments in the program and mnemonic names or pseudo codes for algorithms. The programs must be written in Java and the algorithms must be written in general / standard form, wherever required / specified.*

(Flowcharts are **not** required.)

**Question 11**

**WordPile** is an entity which can hold maximum of 20 characters. The restriction is that a character can be added or removed from one end only.

Some of the members of classes are given below:

| | | |
|---|---|---|
| **Class name** | : | **WordPile** |
| **Data members/instance variables:** | | |
| ch[ ] | : | character array to hold the character elements |
| capacity | : | integer variable to store the maximum capacity |
| top | : | to point to the index of the topmost element |
| **Methods/Member functions:** | | |
| WordPile( int cap) | : | constructor to initialise the data member capacity = cap, top = -1 and create the WordPile |
| void pushChar( char v) | : | adds the character to the top of WordPile if possible, otherwise output a message "WordPile is full" |
| char popChar() | : | returns the deleted character from the top of the WordPile if possible, otherwise it returns '\\' |

(a) Specify the class **WordPile** giving the details of the **constructor, void pushChar(char)** and **char popChar( )**.
**The main function and algorithm need not be written.** [8]

(b) What is the name of the entity described above and state *one* of its applications. [2]

**Question 12**

A line on a plane can be represented by coordinates of the two-end points p1 and p2 as p1(x1, y1) and p2(x2, y2). **[10]**

A super class **Plane** is defined to represent a line and a sub class **Circle** to find the length of the radius and the area of circle by using the required data members of super class.
Some of the members of both the classes are given below:

| | | |
|---|---|---|
| **Class name** | : | **Plane** |
| **Data members/instance variables:** | | |
| x1 | : | to store the x-coordinate of the first end point |
| y1 | : | to store the y-coordinate of the first end point |
| **Member functions/methods:** | | |
| Plane( int nx, int ny ) | : | parameterized constructor to assign the data members x1=nx and y1=ny |
| void Show( ) | : | to display the coordinates |
| **Class name** | : | **Circle** |
| **Data members/instance variables:** | | |
| x2 | : | to store the x-coordinate of the second end point |
| y2 | : | to store the y-coordinate of the second end point |
| radius | : | double variable to store the radius of the circle |
| area | : | double variable to store the area of the circle |
| **Member functions / methods** | | |
| Circle(...) | : | parameterized constructor to assign values to data members of both the classes |
| void findRadius( ) | : | to calculate the length of radius using the formula: $(\sqrt{(x2-x1)^2+(y2-y1)^2})/2$ assuming that x1, x2, y1, y2 are the coordinates of the two ends of the diameter of a circle |
| void findArea( ) | : | to find the area of circle using formula: $\pi r^2$. The value of pie ($\pi$) is 22/7 or 3.14 |
| void Show( ) | : | to display both the coordinates along with the length of the radius and area of the of the circle |

Specify the class **Plane** giving details of the **constructor** and **void Show( )**. Using the **concept of inheritance**, specify the class **Circle** giving details of the **constructor, void findRadius( ), void findArea( )** and **void Show( ).**
**The main function and algorithm need not be written.**

---