ISC SEMESTER 2 EXAMINATION
SPECIMEN QUESTION PAPER
COMPUTER SCIENCE PAPER 1 (THEORY)

Maximum Marks: 35

Time allowed: One and a half hour

*Candidates are allowed an additional **10 minutes** for **only** reading the paper.*

*They must **NOT** start writing during this time.*

---

*Answer **all** questions in **Section A, Section B** and **Section C**.*

While answering questions in Sections A and B, working and reasoning may be indicated briefly.

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

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

SECTION A -7 MARKS

Question 1

(i) The keyword used by a class to acquire the properties of an interface is: **[1]**

(a) import

(b) implements

(c) extends

(d) include

(ii) The ability of an object to take many forms is known as: **[1]**

(a) inheritance

(b) data abstraction

(c) overriding

(d) polymorphism

(iii) int Toy(int n) **[1]**
{ return (n<=0)? 1: n%10 + Toy(n/10); }

With reference to the program code given above, what will the function **Toy()** return when the value of n=56 ?

(a) 65

(b) 12

(c) 651

(d) 11

(iv) Write the statement in Java to extract the word "MISS" from the word "SUBMISSION". **[1]**

(v) State the principle by which the stack data structure works. **[1]**

(vi) What is the output of the statement given below? **[1]**
System.out.print("FAN" + ("AUTOMATIC".charAt(5)));

(vii) Give one reason, why iteration is better than recursion. **[1]**

SECTION B - 8 MARKS

Question 2 **[2]**

Differentiate between *direct recursion* and *indirect recursion*.

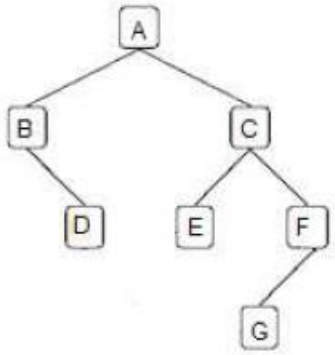
Question 3 **[2]**

Convert the following infix notation to postfix notation:

A * (B + C / D) – E / F

Question 4

Answer the following questions on the diagram of a Binary Tree given below:



(i) State the degree of the nodes C and G. Also, state the level of these nodes when the root is at level 0. **[2]**

(ii) Write the *pre order* and *post order* traversal of the above tree structure. **[2]**

SECTION C - 20 MARKS

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 5 **[6]**

(i) Design a class **Check** which checks whether a word is a palindrome or not. (Palindrome words are those which spell the same from either ends).

Example: MADAM, LEVEL etc.

The details of the members of the class are given below:

Class name	:	**Check**
Data members/instance variables:		
wrd	:	stores a word
len	:	to store the length of the word
Methods/Member functions:		
Check( )	:	default constructor
void acceptword( )	:	to accept the word
boolean palindrome ( )	:	checks and returns 'true' if the word is a palindrome otherwise returns 'false'
void display( )	:	displays the word along with an appropriate message

Specify the class **Check** giving details of the **constructor**, **void acceptword( ), boolean palindrome( )** and **void display( ).** Define the **main( )** function to create an object and call the functions accordingly to enable the task.

OR

(ii) Design a class **Toggle** which toggles a word by converting all upper case alphabets to lower case and vice versa.

Example: The word "mOTivATe" becomes "MotIVatE"

The details of the members of the class are given below:

Class name	:	**Toggle**

Data members/instance variables:

str	:	stores a word
newstr	:	stores the toggled word
len	:	to store the length of the word

Methods/Member functions:

Toggle( )	:	default constructor
void readword( )	:	to accept the word
void toggle ( )	:	converts the upper case alphabets to lower case and all lower case alphabets to upper case and stores it in *newstr*
void display( )	:	displays the original word along with the toggled word

Specify the class **Toggle** giving details of the **constructor**, **void readword( ), void toggle( )** and **void display( ).** Define the **main( )** function to create an object and call the functions accordingly to enable the task.

Question 6 **[6]**

(i) A class **Fibo** has been defined to generate the Fibonacci series 0, 1, 1, 2, 3, 5, 8, 13,……. (Fibonacci series are those in which the sum of the previous two terms is equal to the next term).

Some of the members of the class are given below:

Class name	:	**Fibo**
Data member/instance variable:		
start	:	integer to store the start value
end	:	integer to store the end value
Member functions/methods:		
Fibo( )	:	default constructor
void read( )	:	to accept the numbers
int fibo(int n)	:	return the n^{th} term of a Fibonacci series using **recursive technique**
void display( )	:	displays the Fibonacci series from *start* to *end* by invoking the function *fibo()*

Specify the class **Fibo**, giving details of the **Constructor**, **void read( ), int fibo(int),** and **void display( )**. Define the **main()** function to create an object and call the functions accordingly to enable the task.

OR

(ii) A class **Gcd** has been defined to find the Greatest Common Divisor of two integer numbers. Some of the members of the class are given below:

Class name	:	**Gcd**
Data member/instance variable:		
num1	:	integer to store the first number
num2	:	integer to store the second number
Member functions/methods:		
Gcd( )	:	default constructor
void accept( )	:	to accept the numbers
int gcd(int x,int y)	:	return the GCD of the two number x and y using **recursive technique**
void display( )	:	displays the result with an appropriate message

Specify the class **Gcd**, giving details of the **Constructor**, **void accept( ), int gcd(int,int), and void display( )**. Define the **main()** function to create an object and call the functions accordingly to enable the task.

Question 7 **[4]**

A super class **Godown** has been defined to store the details of the stock of a retail store. Define a subclass **Update** to store the details of the items purchased with the new rate and update the stock. Some of the members of both the classes are given below:

Class name	:	**Godown**
Data members/instance variables:		
item	:	to store the name of the item
qty	:	to store the quantity of an item in stock
rate	:	to store the unit price of an item
amt	:	to store the net value of the item in stock
Member functions/methods:		
Godown(…)	:	parameterized constructor to assign value to the data members
void display( )	:	to display the stock details
Class name	:	**Update**
Data members/instance variables:		
pur_qty	:	to store the purchase quantity
pur_rate	:	to store the unit price of the purchased item
Member functions / methods		
Update(…)	:	parameterized constructor to assign values to the data members of both the classes
void update( )	:	to update the stock by adding the previous quantity by the purchased quantity and replace the rate of the item if there is a difference in the purchase rate. Also update the current stock value as: (quantity * unit price)
void display( )	:	to display the stock details before and after updating

*Assume that the super class **Godown** has been defined*. Using the **concept of inheritance**, specify the class **Update** giving details of the **constructor**, **void update ( )** and **void display( )**.

The super class, main function and algorithm need NOT be written.

Question 8 **[4]**

A Queue is a linear data structure in which the operations are performed based on FIFO (First In First Out).

Define a class Q**ueue** with the following details:

Class name	:	**Queue**

Data member/instance variable:

dat[ ]	:	array to hold the integer elements
cap	:	stores the maximum capacity of the queue
front	:	to point the index of the front
rear	:	to point the index of the rear.

Member functions/methods:

Queue(int max)	:	constructor to initialize the data member cap = max, front = rear = 0 and create the integer array
void add_dat(int v)	:	to add integers from the rear index if possible else display the message("Queue full")
int pop_dat( )	:	to remove and return elements from front, if any, else returns -999
void display()	:	to display elements of the queue

Specify the class Q**ueue** giving the details of **void add_dat(int)** and **int pop_dat( )**. Assume that the other functions have been defined.

The main( ) function and algorithm need NOT be written.