ISC

COMPUTER SCIENCE

PAPER 1 (THEORY)

- **SOLUTION OF 2012**
- **COMMENTS OF COUNCIL EXAMINERS**
- **SUGGESTIONS FOR TEACHERS**

Dedicated to all my lovely students. May God help you always.

This small booklet contains solution of 2012 ISC Computer Science Paper 1 (Theory). The comments from the council examiners under solution of every question makes this a very handy guide for students to understand what the council expects as answer from the students.

I hope that the students will find this to be useful.

Md. Zeeshan Akhtar

15th January, 2015

BLANK PAGE

COMPUTER SCIENCE PAPER 1 (THEORY)

PART I

*Answer **all** questions.*

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

Question 1

(a) Using a truth table, verify the following expression: **[2]**

$X + (Y + Z) = (X + Y) + Z$

Also state the law.

(b) Given, $F (X, Y, Z) = (X' + Y') . (Y + Z')$ **[2]**

write the function in canonical product-of-sum form.

(c) Draw the truth table and logic circuit for a 2-input XNOR gate. **[2]**

(d) Find the complement of the following expression: **[2]**

$X' + XY'$

(e) If $(X \Rightarrow Y)$ then write its : **[2]**

(i) Converse

(ii) Contra positive

Comments of Examiners

(a) Most of the candidates answered correctly. Some were confused in stating / identifying the law.

(b) Most of the candidates answered correctly. Several candidates wrote the function in SOP form. Some reduced the Boolean expression and written it in POS form.

(c) Candidates answered this part correctly. Some drew the XOR gate instead of XNOR gate.

(d) Most of the candidates answered this part correctly and scored full credit. Some candidates did not show the working and some wrote the dual.

(e) Candidates either scored full credit or did not score at all. Some were confused with 'inverse', 'converse' and 'contra positive' and interchanged the answers.

Suggestions for teachers

- Explain Laws and Theorems and prove with the help of truth table. Also give practice in proving equations using Truth Table or algebraic method.
- Give practice of canonical and cardinal form of expression along with inter conversion.
- Give more practice with all logic gates along with truth table and symbol. Difference between XOR and XNOR should be clearly explained.

- Teach candidates to implement De Morgan's law to find the compliment of any given Boolean expression.
- Teach proposition logic using all the terms that are required. Example may be given to differentiate the terms.

MARKING SCHEME

Question 1.

(a) Truth table for X + (Y+Z) = (X+Y) + Z

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

The law is **Associative Law**.

(b) F (X, Y, Z) = (X' + Y') . (Y + Z')

= (X' + Y' + 0) . (Y + Z' + 0)

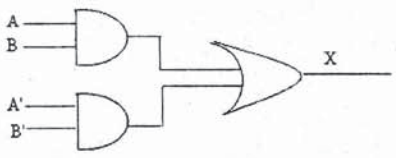
= (X' + Y' + (Z . Z')) . (Y + Z' + (X . X'))

= (X' + Y' + Z) . (X' + Y' + Z') . (X + Y + Z') . (X' + Y + Z')

(c)

A	B	X
0	0	1
0	1	0
1	0	0
1	1	1

X = A B + A' B'



(d) F = X' + XY'

F' = [X' + XY']'

= X''. (XY')'

= X . (X'+Y'')

= X.X' + XY

= XY

(e) If (X => Y)

Converse = (Y => X)

Contra positive = (~ Y => ~ X)

Question 2

(a) Differentiate between the keywords *extends* and *implements*. **[2]**

(b) State how a binary tree is a recursive data structure. **[2]**

(c) A matrix B[10][7] is stored in the memory with each element requiring 2 bytes of storage. If the base address at B [**x**][1] is 1012 and the address at B [7][3] is 1060, determine the value **'x'** where the matrix is stored in **Column Major** wise. **[2]**

(d) Convert the following infix notation to its postfix form: **[2]**

A + ((B+C) + (D+E) * F) / G

(e) What is a constructor? State *one* difference between a constructor and any other member function of a class. **[2]**

Comments of Examiners

(a) Some candidates were confused with the term 'implements'. The term 'extends' was answered correctly by most of the candidates. Use of interface or multiple inheritance were given credit.

(b) Most of the candidates answered using a tree diagram to represent recursive structure. Some failed to explain the concept of recursive data structure and relate binary tree with recursion.

(c) Candidates having the knowledge of the formula to find the address of a cell in an array could solve and get the correct answer. Others answered vaguely. Some calculated using memory cell diagram.

(d) Most candidates were able to solve this problem correctly. Some candidates wrote the correct answer without showing the working. Some applied the postfix correctly, but could not derive the final answer.

(e) Most of the candidates answered this part correctly. Some have explained using examples.

Suggestions for teachers

- Give more practice in interface and its use. Single and multiple inheritance must be explained to differentiate between class and interface.
- Give knowledge of recursion and recursive data structure with examples. Memory blocks may be used to demonstrate repetitions.
- Give practice in understanding the formula using row major and column major and to make the subject of formula that is required in the question.
- Ask students to practice examples with conversion of Infix to Postfix notation, the order of precedence and also the Polish string method.
- Explain definition and differences with examples.

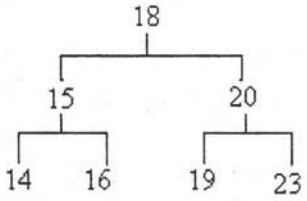
MARKING SCHEME

Question 2.

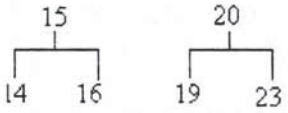
(a) *extends* : - It is a keyword used to inherit the properties of class by another class or by instance by another instance. E.g. A extends B

implements : - It is a keyword used to implement the properties of an interface by a class. E.g. class A implements B

(b) A binary tree consists of other binary trees as its children, i.e. a binary tree is made up of other binary trees. Example: The following binary tree



has two binary tree as its children which are



This property of binary tree is applicable to all levels of binary tree. So binary trees are recursive data structures.

(c) Column major address formula : $B[i][j] = BA + W\,[(j - l_c) * row + (i - l_r)]$

$BA = 1012,\ l_r = x\ ,\ l_c = 1\ ,\ W = 2\ ,\ rows = 10\ ,\ column = 7\ ,\ i = 7\ ,\ j = 3\ ,\ B[7][3] = 1060$

$1060 = 1012 + 2\,[\,(\,3 - 1\,) \times 10 + (\,7 - x\,)\,]$

$1060 = 1012 + 40 + 14 - 2x$

$2x = 6$

X=3

(d) A + ((B+C) + (D+E)*F)/G

= A+((BC+) + (DE+)*F)/G

= A+((BC+) + (DE+F*))/G

=A+(BC+DE+F*+)/G

= A+(BC+DE+F*+G/)

= ABC+DE+F*+G/+

(e) A constructor is a member function which has the same name as its class and it gets invoked every time a new object is created. It is used to construct memory and initialize object values.

Constructor	Member function of class
• Constructors do not return a value not even void. • Invoked automatically. • Same name as that of the class	• Member function of a class can return values. • Has to be called. • Has a different name.

Question 3

(a) The following function is a part of some class which computes and sorts an array arr[ ] in ascending order using the **bubble sort technique**. There are some places in the code marked by **?1?, ?2?, ?3?, ?4?, ?5?** which must be replaced by a statement / expression so that the function works properly:

```
void   bubblesort(int arr[ ])
{
      int  i, j, k, tmp;
      for(i= 0; ?1?; i++)
        {
          for(j = 0; ?2?; j++)
            {
              if(arr[j] > ?3?)
                {
                  tmp      =  arr[j];
                  ?4?      =  arr[j+1];
                  arr[j+1] =  ?5?;
                }
            }
        }
}
```

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

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

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

(b) The following function **witty( )** is a part of some class. What will be the output of the function witty( ) when the value of n is "SCIENCE" and the value of p = 5. Show the dry run/ working: **[5]**

```
void witty(String n, int p)
{
        if (p < 0 )
                System.out.println ("");
        else
        {       System.out.println (n.charAt(p) + " . " ) ;
                witty( n , p-1 ) ;
                System.out.print (n.charAt(p) ) ;
        }
}
```

Comments of Examiners

(a) Most of the candidates answered correctly. Some candidates were not clear about the size of the array and gave multiple answers.

(b) Many candidates answered correctly. Some were not clear about the concept of recursive technique. A few candidates did not show the working / dry run. Some were confused as the counter itself was decrementing.

Suggestions for teachers

- Give practice on program using arrays and other output related programs.
- Explain the meaning of "Array Index Out of Bound" exception.
- Give practice in various techniques relating to problem solving using recursion. Output programs must be explained using diagrams for each function call.

MARKING SCHEME

Question 3.

(a) (i) i < arr.length -1 **or** i < size-1 **or** i < k-1 (where k = arr.length)

(ii) j < arr.length – 1 **or** j < arr.length-1 – i [same size and k as above]

(iii) arr[j + 1]

(iv) arr[j]

(v) tmp

(b) witty (" SCIENCE" , 5)

C. witty (" SCIENCE" , 4)

N. witty (" SCIENCE" , 3)

E. witty (" SCIENCE" , 2)

I. witty (" SCIENCE" , 1)

C. witty (" SCIENCE" , 0)

S. witty (" SCIENCE" , -1)

OUTPUT : C.

N.

E.

I.

C.

S.

SCIENC

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, **$F(A,B,C,D) = \Sigma$ (4, 6, 7, 10, 11, 12, 14, 15)**

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

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

(b) Given, **$F(P,Q,R,S) = \pi$ (0, 5, 7, 8, 10, 12, 13, 14, 15)**

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

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

Comments of Examiners

(a) Most candidates answered this question correctly. Some candidates made errors in place value and putting variables in K-Map. In some cases the groups were reduced by laws. Some candidates drew the K-Map incorrectly. Many candidates included the redundant group in the final expression.

(b) Most candidates fared well in this part. Some candidates were not able to draw the K-Map for the POS expression correctly. For a number of candidates, the "Map rolling" concept was not very clear. Some converted the canonical form to cardinal form and then reduced it.

Suggestions for teachers

- Emphasize on arranging the variables in proper order and the importance of cell values corresponding with the variables. Explain clearly how the groups are framed and reduced. Redundant groups are not to be included in the final reduced expression.
- Make students reduce POS and SOP expressions using K-Map simultaneously. Students should be told not to include the redundant group in the final expression.

MARKING SCHEME

Question 4.

(a) $F(A,B,C,D) = \Sigma\ (4, 6, 7, 10, 11, 12, 14, 15)$

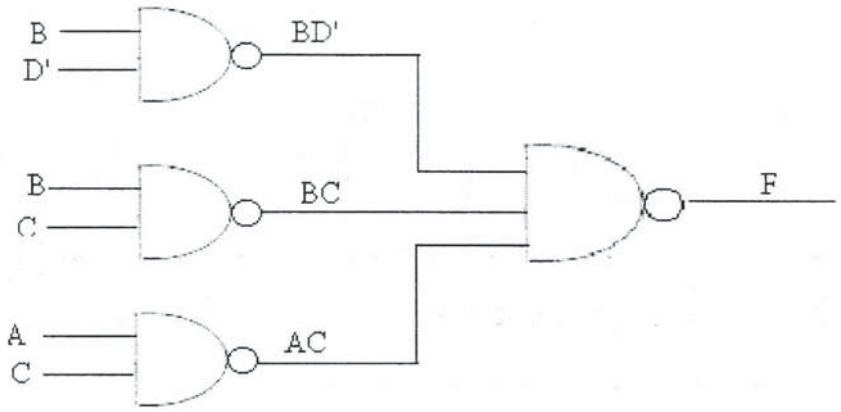
	C'D'	C'D	CD	CD'
A'B'	0 0	1 0	3 0	2 0
A'B	4 1	5 0	7 1	6 1
AB	12 1	13 0	15 1	14 1
AB'	8 0	9 0	11 1	10 1

There are three quads :

Quad1 ($m_4 + m_6 + m_{12} + m_{14}$) = BD' Quad2 ($m_6 + m_7 + m_{14} + m_{15}$) = BC

Quad3 ($m_{10} + m_{11} + m_{14} + m_{15}$) = AC

Hence F (A, B, C, D) = BD' + BC + AC



(b) $F(P,Q,R,S) = \pi (0, 5, 7, 8, 10, 12, 13, 14, 15)$

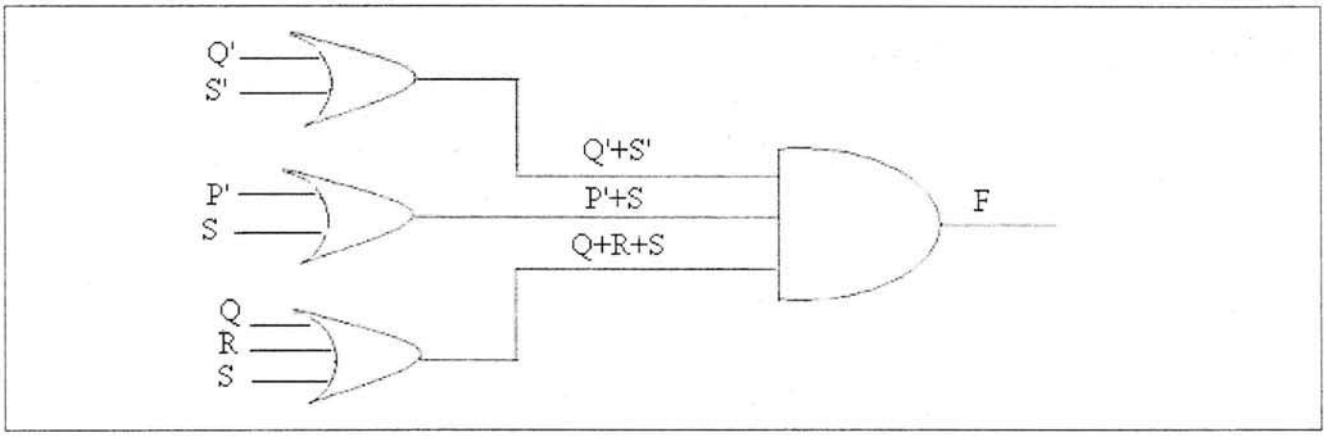
	R+S	R+S'	R'+S'	R'+S
P+Q	0 0	1 1	3 1	2 1
P+Q'	4 1	5 0	7 0	6 1
P'+Q'	12 0	13 0	15 0	14 0
P'+Q	8 0	9 1	11 1	10 0

There are two quads and a pair :

Quad 1 : ($M_5 M_7 M_{13} M_{15}$) = Q' + S' Quad 2 : ($M_8 M_{10} M_{12} M_{14}$) = P' + S

Pair : ($M_0 M_8$) = Q + R + S

Hence F(P,Q,R,S) = (Q' + S') . (P' + S) . (Q + R + S)



Question 5

The Principal of a school intends to select students for admission to class XI on the following criteria:

- Student is of the same school and has passed the class X Board Examination with more than 60% marks.

OR

- Student is of the same school, has passed the class X Board Examination with less than 60% marks but has taken active part in co-curricular activities.

OR

- Student is not from the same school but has either passed the class X Board Examination with more than 60% marks or has participated in sports at the National level.

The inputs are :

INPUTS	
S	Student of same school.
P	Has passed the class X Board Examination with more than 60% marks.
C	Has taken active part in co-curricular activities.
T	Participated in sports at the National level.

Output: **X** - Denotes admission status [1 indicates granted and 0 indicates refused in all the cases]

(a) Draw the truth table for the inputs and outputs given above and write the **SOP** expression along with its logic gate diagram. **[5]**

(b) Reduce **X (S, P, C, T)** using Karnaugh's Map. **[5]**

Draw the logic gate diagram for the reduced **SOP** expression for X (S, P, C, T) using AND and OR gate. You may use gates with two or more inputs. Assume that the variable and their complements are available as inputs.

Comments of Examiners

(a) Most of the candidates answered this part well and scored full credit. Some did not mention the final expression.

(b) Some candidates made the same mistake for the K-map i.e. place value, Map rolling, cell reference etc. However other candidates answered well.

Suggestions for teachers

- Ask students to read the question carefully and to answer accordingly so that no part is left unanswered.
- Emphasize on arranging the variables in proper order and the importance of cell values corresponding with the variables. Explain clearly how the groups are framed and reduced. Redundant groups are not to be included in the final reduced expression.

MARKING SCHEME

Question 5.

(a)

S	P	C	T	X
Student of same school	Has passed the class X Board Examination with more than 60% marks.	Has taken active part in co-curricular activities.	Has participated in sports at the National level.	**OUTPUT**
0	0	0	0	0
0	0	0	1	**1**
0	0	1	0	0
0	0	1	1	**1**
0	1	0	0	**1**
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	**1**
1	0	1	1	**1**
1	1	0	0	**1**
1	1	0	1	**1**
1	1	1	0	**1**
1	1	1	1	**1**

X (A,B,C,D) = Σ **(1 , 3 , 4 , 5 , 6 , 7 , 10 , 11 , 12 , 13 , 14 , 15)**

(b)

	C'T'	C'T	CT	CT'
S'P'	0 0	1 1	3 1	2 0
S'P	4 1	5 1	7 1	6 1
SP	12 1	13 1	15 1	14 1
SP'	8 0	9 0	11 1	10 1

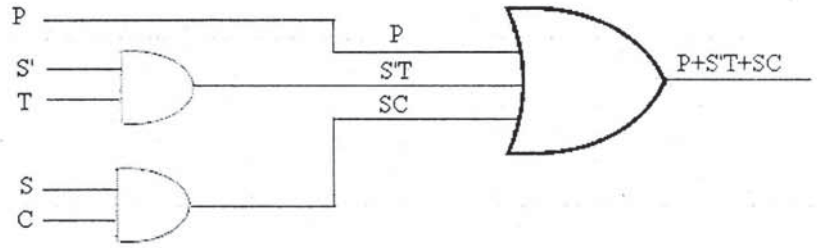
There is one octal and two quads :

Octal (m4+ m5+ m6+ m7 +m12+ m13+ m14+ m15) = P

Quad 1 (m1+ m3+ m5+ m7) = S'T

Quad 2 (m10+ m11+ m14+ m15) = SC

Hence X (S, P, C, T) = P + S'T + SC



Question 6

(a) Verify algebraically if, **[2]**

X′Y′Z′ + X′Y′Z + X′YZ + X′YZ′ + XY′Z′ + XY′Z = X′+Y′

(b) Represent the Boolean expression X+YZ′ with the help of NOR gates only. **[2]**

(c) Define the terms CONTINGENCY, CONTRADICTION and TAUTOLOGY. **[3]**

(d) Consider the following truth table where A and B are two inputs and X is the output:

A	B	X
0	0	0
0	1	1
1	0	1
1	1	0

(i) Name and draw the logic gate for the given truth table. **[2]**

(ii) Write the POS of X(A,B). **[1]**

Comments of Examiners

(a) Most of the candidates answered this part correctly. Some used the truth table to verify.

(b) This part was well answered by most candidates. Some converted the expression to POS and then drew the circuit using NOR gates.

(c) Many candidates answered well. Some explained the term with the help of Truth Table.

(d) (i) This part was well answered by most candidates. Some did not name the gate.

(ii) Some candidates wrote the SOP expression instead of POS. Some wrote the expression in cardinal form.

Suggestions for teachers

- Give practice in Boolean laws and its implication to verify or minimize a Boolean expression.
- Give more practice in drawing logic circuits using Universal gates (NOR and NAND).
- Give ample practice to students in deriving both SOP and POS expression from truth table.

MARKING SCHEME

Question 6.

(a) X'Y'Z'+X'Y'Z+X'YZ+X'YZ'+XY'Z'+XY'Z = X'+Y'

Taking LHS of the above equation

X'Y'(Z'+Z) + X'Y(Z+Z')+XY'(Z'+Z)

= X'Y'.1 + X'Y.1 + XY'.1 [Z+Z' = 1 by complimentary law]

= X'Y'+ X'Y+ XY' [properties of 0 and 1]

= X'(Y'+Y) + XY'

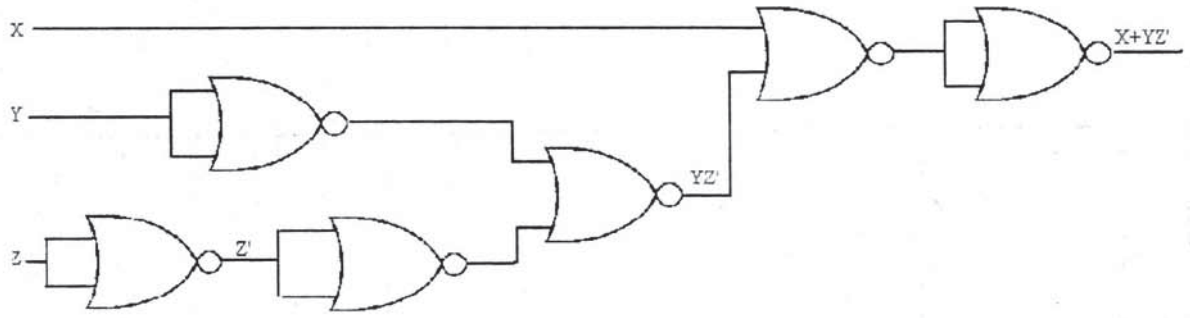
= X'.1 + XY' [Y+Y' = 1 by complimentary law]

= X' + XY' [X'.1 = X' by properties of 0 and 1]

= (X'+X)(X'+Y') [Distributive law]

= X'+Y' [X+X' = 1 by complimentary law]

(b)



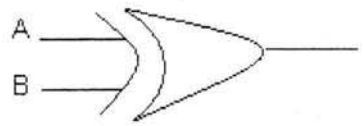
(c) CONTINGENCIES: The propositions that have some combination of 1's and 0's in their truth table output column.

CONTRADICTIONS: The propositions that have all 0's in their truth table output column.

TAUTOLOGY: The propositions that have all 1's in their truth table column.

(d) (i) The output X represents a **XOR** gate.

Logic gate for XOR is



(ii) POS for X(A,B) = (A+B) (A'+B')

Question 7

(a) Define Multiplexer and state *one* of its uses. Draw the logic diagram for a 4:1 Multiplexer. **[4]**

(b) State how a *Half Adder* is different from a *Full Adder*. Also give their respective uses. **[3]**

(c) Minimize the following expression using Boolean laws: **[3]**

$$Q \,.\, (Q' + P) \,.\, R \,.\, (Q + R)$$

Also draw the logic gate for the reduced expression.

Comments of Examiners

(a) The terms 'combinational circuit' was not used in the definition by many candidates. Some were able to give suitable examples of their uses / applications. The logic diagram was well answered by most candidates. Some drew the OR gate instead of AND gate. Some did not use the OR gate to combine parallel lines to a single serial line. A few gave only the block diagram.

(b) Most of the candidates answered this part correctly. The word 'adding' was missing in some cases.

(c) Most of the candidates were able to attempt this part correctly.

Suggestions for teachers

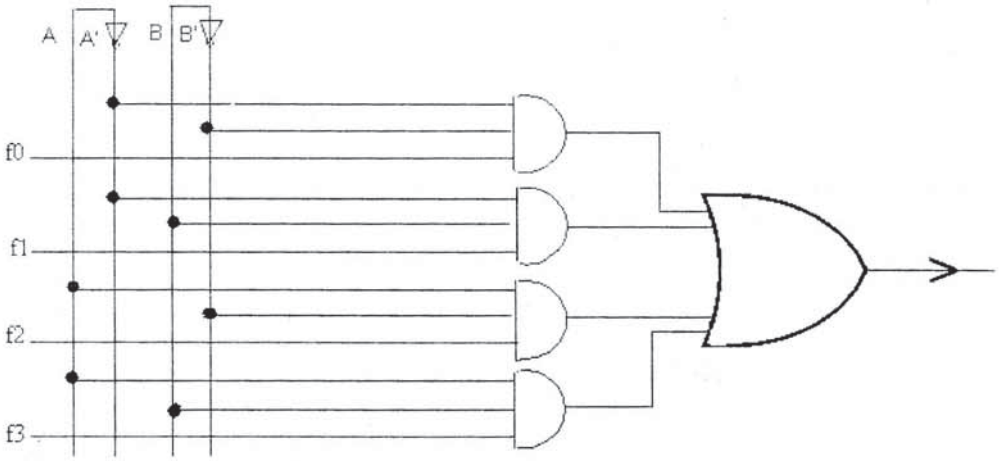
- Encourage students to write the correct definition and also the logic diagram for the same along with their use. Use of proper gates must be given more attention while drawing the logic diagram.
- Give practice in drawing the full adder and half adder along with its expression, truth table and logic circuit / diagram.
- Give practice in reducing / minimizing expressions using Boolean laws.

MARKING SCHEME

Question 7.

(a) **Multiplexer** is a combinational circuit which inputs parallel data and outputs one serial data.

It is used in signaling, telephone exchange, data transmission etc.



(b) **Half adder** are combinational circuits which adds 2 inputs binary bits and outputs 2 binary bits (partial sum and carry)

Use : To add two binary bits

Full adder are combinational circuits which adds 3 inputs binary bits and outputs 2 binary bits (sum and carry zero)

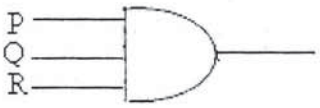
Use : To add three binary bits

(c) $Q . (Q' + P) . R . (Q + R)$

$= (Q.Q' + QP) . (RQ + RR)$

$= QP . (RQ + R)$

$= PQR$



AND gate

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 Combine contains an array of integers which combines two arrays into a single array including the duplicate elements, if any and sorts the combined array. Some of the members of the class are given below: **[10]**

Class name	:	**Combine**
Data members / instance variables:		
com[ ]	:	integer array
size	:	size of the array
Member functions/methods:		
Combine (int nn)	:	parameterized constructor to assign size = nn
void inputarray( )	:	to accept the array elements
void sort( )	:	sorts the elements of combined array in ascending order using the **selection sort** technique
void mix(Combine A, Combine B)	:	combines the parameterized object arrays and stores the result in the current object array along with duplicate elements, if any
void display( )	:	displays the array elements

Specify the class Combine giving details of the **constructor(int), void inputarray( ), void sort( ), void mix(Combine, Combine)** and **void display( )** . Also define the **main( )** function to create an object and call the methods accordingly to enable the task.

Comments of Examiners

Most of the candidates answered this part well.
Some candidates used different sorting techniques instead of Selection sort method. Passing of objects was not clear to some of the candidates. The other functions were well answered including the constructor. The main function was not attempted by some candidates.

Suggestions for teachers

- Explain clearly passing objects to functions. Emphasize that only the mentioned technique should be written.
- More practice should be given in writing the main function with each and every program.
- Explain the importance of loop to display all the elements of array.

MARKING SCHEME

Question 8.

```
import java.io.*;
public class Combine
{  int com[]=new int[100];
   int size;
   static BufferedReader br=new BufferedReader(new InputStreamReader(System.in));
  Combine(int nn)
   {  size=nn;   }
 void inputarray()throws IOException
 {  System.out.print("\n Enter" + size + " elements");
    for(int i=0;i<size;i++)
       com[i]=Integer.parseInt(br.readLine());
 }
 void sort()
 {  int m,t;
    for(int i=0;i<size-1;i++)
    {  m=i;
       for(int j=i+1;j<size;j++)
       {  if(com[m]>com[j])
             m=j;
       }
         if(m!=i)
         {  t=com[m];
            com[m]=com[i];
            com[i]=t;
         } } }
 void mix(Combine A,Combine B)
 {   int x=0,y=0,z=0;
     while(x<A.size)
       com[z++]=A.com[x++];
     while(y<B.size)
       com[z++]=B.com[y++];
 }
```

```
    void display( )
    { for(int i=0;i<size;i++)
      System.out.println(com[i]);
    }
    static void  main()throws IOException
    {
       System.out.print("\n enter size of first array");
       int a=Integer.parseInt(br.readLine());
       System.out.print("\n enter size of second array");
       int b=Integer.parseInt(br.readLine());
       Combine P=new Combine(a);
       Combine Q=new Combine(b);
       Combine R=new Combine(a+b);
       P.inputarray();
       Q.inputarray();
       R.mix(P,Q);
       R.sort();
       R.display();
     }
}
```

Question 9

Design a class VowelWord to accept a sentence and calculate the frequency of words that begin with a vowel. The words in the input string are separated by a single blank space and is terminated by a full stop. The description of the class is given below: **[10]**

Class name	:	**VowelWord**
Data members / instance variables:		
str	:	to store a sentence
freq	:	store the frequency of the words beginning with a vowel
Member functions:		
VowelWord( )	:	default constructor to initialize data members to legal initial value
void readstr( )	:	to accept a sentence
void freq_vowel( )	:	counts the frequency of the words that begin with a vowel

void display( )	:	to display the original string and the frequency of the words that begin with a vowel

Specify the class VowelWord giving details of the **constructor( ), void readstr( ), void freq_vowel( ) and void display( )**. Also define the **main( )** function to create an object and call methods accordingly to enable the task.

Comments of Examiners

A number of candidates answered correctly. In some cases, breaking the sentence into words was not properly done. Some candidates attempted using string tokenizer. Several candidates found the frequency of vowels as an occurrence in the sentence and not at the beginning of a word. The other functions were well answered including the constructor. Input stream was not written in some cases. The main function was not attempted by some of the candidates.

Suggestions for teachers

- Give practice in use of string tokenizer as it simplifies the string program to a large extent.
- Teach string manipulative methods to assist in string related problems.

MARKING SCHEME

Question 9.

```
import java.util.Scanner;
public class VowelWord
{
  String str;
  int freq;
  public VowelWord()
  {
    str = "";
    freq = 0;
  }
  public void readstr()
  {
    Scanner s = new Scanner(System.in);
    System.out.println("enter a string");
    str = s.nextLine();
   }
  public void freq_vowel( )
  {
    char c,c1=' ';
    for(int i=0;i<str.length();i++)
     {  c=str.charAt(i);
       if(i==0)
         {
           if(c=='a'||c=='e'||c=='i'||c=='o'||c=='u'||c=='A'||c=='E'||c=='I'||c=='O'||c=='U')
             freq++;
         }
```

```
        else if(c==' ')
        {
          c1=str.charAt(i+1);

          if(c1=='a'||c1=='e'||c1=='i'||c1=='o'||c1=='u'||c1=='A'||c1=='E'||c1=='I'||c1=='O'||c1=='U')
            freq++;
        } }
}
 void display()
 {
    System.out.print("\n ORIGINAL SENTENCE:  " + str);
    System.out.print("\n FREQUENCY OF WORDS BEGINNING WITH A VOWEL   " +freq);
    }
 public static void main()
 {
   VowelWord c = new VowelWord();
   c.readstr();
   c.freq_vowel();
   c.display();
  }
}
```

Question 10

A happy number is a number in which the eventual sum of the square of the digits of the number is equal to 1. **[10]**

Example: $28 = (2)^2 + (8)^2 = 4 + 64 = 68$

$68 = (6)^2 + (8)^2 = 36 + 64 = 100$

$100 = (1)^2 + (0)^2 + (0)^2 = 1 + 0 + 0 = 1$

Hence, 28 is a happy number.

Example: $12 = (1)^2 + (2)^2 = 1 + 4 = 5$

Hence, 12 is not a happy number.

Design a class Happy to check if a given number is a happy number. Some of the members of the class are given below:

Class name	:	**Happy**
Data members/instance variables:		
n	:	stores the number
Member functions:		
Happy( )	:	constructor to assign 0 to n
void getnum(int nn)	:	to assign the parameter value to the number n = nn

int sum_sq_digits(int x)	:	returns the sum of the square of the digits of the number x, using the **recursive technique**
void ishappy( )	:	checks if the given number is a happy number by calling the function sum_sq_digits (int) and displays an appropriate message

Specify the class Happy giving details of the **constructor( )**, **void getnum(int)**, **int sum_sq_digits(int)** and **void ishappy( )**. Also define a **main( )** function to create an object and call the methods to check for happy number.

Comments of Examiners

The concept of recursion was not clear in some cases. Checking the base case in a recursive function was not properly done. Some candidates solved the program directly using loops without using the recursive technique. Some did not invoke the sum_of_digits() function in ishappy() function and answered directly. A number of candidates answered the entire program using sum_of_digits() function only by finding the sum of digits and checking for Happy number also. Main ( ) function not attempted by some candidates.

Suggestions for teachers

- Give more practice in solving programs using recursive techniques. Much attention should be paid by the teachers towards recursion and its techniques with examples.
- Give knowledge of base case and recursive case to the students for every program using recursive technique.
- Function calling and nesting of functions must be practiced more.

MARKING SCHEME

Question 10.

```
import java.io.*;
class Happy
{
   int n;

   Happy ( )
   {
       n=0;
   }
   void getnum( int nn)
   {
       n=nn;
   }
    int sum_sq_digits( int x )
```

```
    {
        return (x<=0)? 0 : (x%10 * x%10) + sum_sq_digits(x/10);
    }
    void ishappy ( )
    {
         int x=n;
         while(x>9)
            x=sum_sq_digits(x);
         if (x==1)
            System.out.print("\n" +   n + " IS A HAPPY NUMBER");
         else
           System.out.print("\n" +   n + " IS NOT A HAPPY NUMBER");
    }
    static void main( ) throws IOException
    {
        BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
        System.out.print("\n Enter a number " );
        int a =Integer.parseInt(br.readLine());
        Happy  X=new Happy( );
        X.getnum(a);
        X.ishappy( );

    }
}
```

SECTION – C

*Answer any **two** questions.*

Each Program / Algorithm should be written in such a way that it clearly depicts the logic of the problem step wise. This can also be achieved by using pseudo codes.

(Flowcharts are **not** required).

The programs must be written in Java.

The Algorithm must be written in general/standard form, wherever required.

Question 11

Link is an entity which can hold a maximum of 100 integers. Link enables the user to add elements from the rear end and remove integers from the front end of the entity. Define a class Link with the following details:

Class name	:	**Link**
Data members/instant variables :		
lnk[ ]	:	entity to hold the integer elements
max	:	stores the maximum capacity of the entity
begin	:	to point to the index of the front end
end	:	to point to the index of the rear end
Member functions:		
Link(int mm)	:	constructor to initialize max = mm, begin = 0, end = 0
void addlink(int v)	:	to add an element from the rear index if possible otherwise display the message "OUT OF SIZE..."
int dellink( )	:	to remove and return an element from the front index , if possible otherwise display the message "EMPTY..." and return –99
void display( )	:	displays the elements of the entity

(a) Specify the class Link giving details of the **constructor(int)**, **void addlink(int)**, **int dellink( )** and **void display( )**. **[9]**

THE MAIN FUNCTION AND ALGORITHM NEED NOT BE WRITTEN.

(b) What type of data structure is the above entity? **[1]**

Comments of Examiners

(a) Several candidates had problems in declaring the array. Some declared the array twice. A few candidates initialized both the begin and end as -1 instead of 0. Some candidates used front and rear instead of begin and end. The constructors were not properly declared and the queue array was not created inside the constructor in some cases. The functions addlink() and dellink( ) were not answered properly. Some of the candidates were confused by the class name 'Link' and wrote the link list program. Some overlooked the return -99 condition. The rest of the functions were well answered.

(b) Most of the candidates were able to attempt this part well.

Suggestions for teachers

- Give more practice in data structure programs like the stacks, queues, dequeues, etc. Working must be shown as to how the stack or a queue performs (examples can be supportive).
- Initialization of front and rear must be explained with both 0 and -1.
- The concept of LIFO and FIFO must be explained to the students with lively examples related to real world.

MARKING SCHEME

Question 11.

(a)

```
class Link
{
  int lnk[ ] = new int[100];
  int max,begin,end;
  Link (int mm )
  { max=mm;
    begin=0;
    end=0;
    lnk =new int [max];
  }
  void addlink( int v)
  { if(end<max-1)
        lnk[++end]=v;
    else
      System.out.println("OUT OF SIZE…");
  }
  int dellink( )
  { int v;
    if(begin!=end)
      { v=lnk[++begin];
        return v; }

    else
      { System.out.println("EMPTY…");
        return -99;
```

```
        }  }
    void display ( )
     {  for(int i=begin + 1;i<=end;i++)
          System.out.println(lnk[i]);
     }}
```

(b) Queue / Linear Data Structure / Circular Queue / FIFO.

Question 12

A super class Detail has been defined to store the details of a customer. Define a sub class Bill to compute the monthly telephone charge of the customer as per the chart given below: **[10]**

NUMBER OF CALLS	RATE
1 – 100	Only rental charge
101 – 200	60 paisa per call + rental charge
201 – 300	80 paisa per call + rental charge
Above 300	1 rupee per call + rental charge

The details of both the classes are given below:

Class name : **Detail**

Data members / instance variables:

name	:	to store the name of the customer
address	:	to store the address of the customer
telno	:	to store the phone number of the customer
rent	:	to store the monthly rental charge

Member functions:

Detail(...)	:	parameterized constructor to assign values to data members
void show( )	:	to display the details of the customer

Class name : **Bill**

Data members /instance variables:

n	:	to store the number of calls
amt	:	to store the amount to be paid by the customer

Member functions :

Bill(...)	:	parameterized constructor to assign values to data members of both classes and to initialize amt = 0.0
void cal( )	:	calculates the monthly telephone charge as per the chart given above
void show( )	:	displays the details of the customer and amount to be paid

Specify the class Detail giving details of the **constructor( )** and **void show( )**. Using the **concept of inheritance,** specify the class Bill giving details of the **constructor( ), void cal( )** and **void show( )**.
THE MAIN( ) FUNCTION AND ALGORITHM NEED NOT BE WRITTEN.

Comments of Examiners

The concept of Inheritance (extends) was not clear to some candidates. Constructor with inheritance was not answered correctly. Accessing the members of the super class by the derived class was not clear to some of the candidates. Many candidates declared the base class data members as private. String data members were not declared properly by some candidates. Different methods were used to calculate the telephone bill. Several candidates did not use the slab given to compute the telephone bill. The rest of the functions were answered well.

Suggestions for teachers

- Give practice on inheritance and the keyword '*extends*' must be explained to students. Use of constructor using the base class member should be made clear. Explain to students the different visibility modes and their accessing capability. Knowledge of calling the member function from the super class to the derived class must be given.
- The use of '*super*' keyword and its application must be practiced.

MARKING SCHEME

Question 12.

```
class Detail
{
   String name,address;
   int telno;
   double rent;
   Detail (String n, String a, int t, double r)
   {  name=n;
      address = a;
      telno=t;
      rent=r;
   }
   void show( )
```

```
    {
      System.out.println("Name : "+ name);
      System.out.println("Address : "+ address);
      System.out.println("Telephone Number : "+ telno);
      System.out.println("Monthly Rent : "+ rent);
    }
}
class Bill extends Detail
  {
     int n ;
     double amt ;
    Bill (String b, String a, int t, double r, int y )
    {  super ( b,a,t,r);
       n=y;
      amt = 0.0;
    }
void cal ( )
    {
        if(n>=1 && n<=100)
          amt=rent;
        else if ( n >= 101 && n < = 200 )
          amt = (n-100) *. 6 + rent;
        else if ( n >= 201 && n < = 300 )
          amt = (100 * .6 ) + (n-200) *. 8 + rent;
        else if ( n > 300)
           amt = (100 * .6 ) + (100 * .8 ) + (n-300) * 1.0 + rent ;
   }
void show ( )
    {  super.show ( ) ;
       System.out.println ( "Number of calls = " + n);
       System.out.println ( "Amount to pay = " + amt);
    }
}
```

Question 13

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

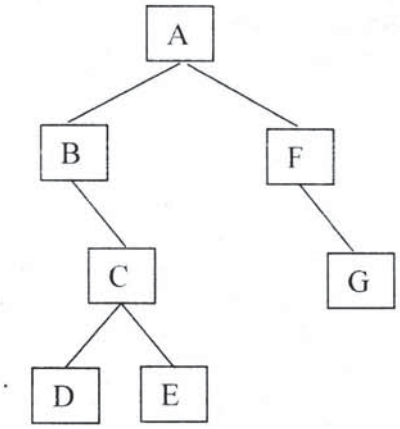
```
class node
   {
      int p;
      String n;
      node  next;
   }
```

Write an **ALGORITHM / METHOD** to search for a name and display the contents of that node. The method declaration is given below:

void search(node start, String b)

(b) What is the role of constants in complexity? Explain briefly with an example. **[2]**

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



(i) External nodes of the tree **[1]**

(ii) Parent of node D. **[1]**

(iii) Inorder traversal of the tree. **[1]**

(iv) Right subtree of Node B. **[1]**

Comments of Examiners

(a) A number of candidates were confused with the position of the node to be searched. Some were not clear in shifting the pointer of the link list to the next node and also checking for null. Several candidates wrote the algorithm while some wrote in program/code form. A few candidates did not declare the temporary variable to point to the beginning of the list or to the first node.

(b) Some candidates gave correct answers while the answers given by others were rather vague. Some candidates simply gave the definition of complexities with the term constants mentioned in it.

(c) (i) A number of candidates were confused with root also as an external node.
(ii) This part was well answered by most of the candidates.
(iii)Almost all candidates answered this part correctly barring a few who gave incomplete answers.
(iv)While most of the candidates were able to attempt this part correctly, some wrote the subtree of the entire tree including the node B.

Suggestions for teachers

- Ensure that more programs / algorithms are practiced with link list and binary tree data structure. Use of diagrams to illustrate the link list and the Binary Trees must be practiced.
- Definition of complexities / big 'O' and the three cases of complexities must be explained in details along with examples. The role of constants in various situations in complexities must be explained.
- Explain binary tree with the different parts like root, nodes (internal and external), height, depth, level, size, tree traversal (preorder, inorder and postorder) etc.

MARKING SCHEME

Question 13.

(a) **Algorithm to search for a node in a linked list.**

Steps :

1 - Start

2 - Set temporary pointer to first node

3 - Repeat steps 4 and 5 until the pointer reaches null

4 - if the name matches with the parameter value then Print details, STOP ELSE

5 - move temporary pointer to the next node

6 - End

Method to search for a node in a linked list

```
void search ( node start, String b )
    {       node a = new node(start)
          int f=-1;
             while (a!=null && f==-1)
            {   if(a.n. equals(b))
                 f=1;
                else
                        a=a.next;
                }
             if(f==1)
                 System.out.println(a.p);
             else
              System.out.println(" Node not found");
    }
```

(b) When Big 'O' notation is used, constants and low-order terms are dropped.

However, if two algorithms have the same Big 'O' time complexity then also one may be faster than the other. In such cases Constants do matter in terms of which algorithm is actually faster.

Example :

Suppose algorithm 1 requires N^2 time and algorithm 2 requires $10 * N^2 + N$ time.

For both algorithms, the time is $O(N^2)$, but algorithm 1 will be faster than algorithm 2.

(c) 1. D, E, G

2. C

3. B D C E A F G

4. C, D, E

GENERAL COMMENTS

(a) Topics found Difficult by candidates in the Question Paper:

- Canonical and Cardinal form of expressions.
- Propositional logic (Converse, Contra positive)
- Keyword 'implement' as used in interfaces.
- Binary tree as a recursive data structure.
- K-MAPS (Grouping, map-rolling , place value)
- Passing objects to functions.
- Recursive technique and invoking function within another function (Nesting of functions).
- Role of constants in complexities.

(b) Concepts between which candidates got confused:

- Converse, inverse and contra positive.
- XOR and XNOR gates diagram
- Address calculation.
- Size of array in the output program.
- Multiplexer logic diagram.
- Passing of objects to functions.
- Invoking function within function.
- Slab to calculate telephone bill in inheritance.
- Class name 'Link' was misunderstood as link list program.
- Initialization of begin and end in Queue.
- Role of constants in complexities.
- Algorithms for link list

(c) Suggestions for students:

- Prepare summary for each chapter or use highlighters to recognize the important terms and definitions.
- Run each program and debug where necessary. Test for logical errors with different values.
- Relate to real life application of the concept learned.
- Give answers and definitions that are short and precise and according to marks intended.
- Read the question properly and answer accordingly. Important words and terms should be underlined or highlighted.
- Working should be shown at the side of each question wherever required.
- Laws must be mentioned while reducing a Boolean Expression.
- Practice one form of K-Map with proper place value for both SOP and POS.
- In programming documentation is compulsory and should be mentioned with each program.
- Declare the class with data members and member functions. Expand or define each function according to the instructions given by the side of each function.
- Do not memorize the program, try to understand the logic. Practice constructors with every program. Treat each function of a class as separate program.