ISC

COMPUTER SCIENCE

PAPER 1 (THEORY)

- **SOLUTION OF 2013**
- **COMMENTS OF COUNCIL EXAMINERS**
- **SUGGESTIONS FOR TEACHERS**

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

This small booklet contains solution of 2013 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) State the Principle of Duality. Write the dual of: **[2]**

$(P+Q').R.1 = P.R + Q'.R$

(b) Minimize the expression using Boolean laws: **[2]**

$F = (A + B').(B + CD)'$

(c) Convert the following cardinal form of expression into its canonical form: **[2]**

$F(P,Q,R) = \pi\ (1,3)$

(d) Using a truth table verify: **[2]**

$(\sim p \Rightarrow q) \wedge p = (p \wedge \sim q) \vee (p \wedge q)$

(e) If A = 1 and B = 0, then find: **[2]**

(i) $(A' + 1).B$

(ii) $(A + B')'$

Comments of Examiners

(a) This part was well answered by most of the candidates. Some candidates did not give the definition, but wrote the dual correctly. Some did not bracket the terms for the dual equation. Interchanging of signs '+' to '-'and 1's to 0's was not mentioned in some cases. A few candidates did not mention that the complements do not change.

(b) Some candidates confused it with De Morgan's Law and reduced the expression incorrectly. Some did not change the signs when complements were taken.

(c) The terms 'Cardinal' and 'Canonical' were confusing to a number of candidates. Several candidates wrote the SOP expression instead of POS as they were confused with the symbols $\sum$ and Π.

Suggestions for teachers

- Practice should be given to candidates to find the dual of equations and also the application of Principle of duality in Boolean equations. Importance of brackets must be explained. Differences between complementation and duality must be clarified.
- More practice should be given to candidates to minimize a Boolean expression using Boolean laws. Each and every law must be practiced with examples.

(d) Most of the candidates answered this part well. Some were not clear with the terms and symbols (=>, Λ, V, ~) used in a Proposition.

(e) (i) Most of the candidates were able to answer this part correctly. Only a few did not show the working and wrote the answer directly. Some represented the values but did not find the final result.

(ii) A few candidates did not show the working and wrote the answer directly.

- Inter conversion of canonical and cardinal expression must be given more practice. The terms along with definition must be explained with examples. The representation of the symbols ∑ and Π should be explained.
- Proposition logic should be taught using all the terms that are required. All the symbols related to propositional logic must be explained in detail.
- More practice should be given in solving such type of Boolean expressions. Importance of the word 'find' in a question must be emphasized.
- More practice should be given in solving such type of Boolean expressions.

MARKING SCHEME

Question 1.

(a) To every Boolean equation there exists another equation which is dual to the previous equation. This is done by changing AND's to OR's and vice-versa, 0's to 1's and vice-versa, complements remain unchanged.

Dual : (P . Q') + R + 0 = (P + R) . (Q' + R)

(b) F = (A + B') . (B + CD)'

F = (A + B') . (B' . (CD)')

F = AB' + B'B' . (C' + D')

F = B' . (C' + D')

(c) F (P,Q,R) = π (1 , 3)

= 001 , 011

= (P + Q + R') . (P + Q' + R')

(d) (~p => q) Λ p = (p Λ ~q) V (p Λ q)

p	q	~p	~q	~p => q	L.H.S	p Λ ~q	p Λ q	R.H.S
0	0	1	1	0	0	0	0	0
0	1	1	0	1	0	0	0	0
1	0	0	1	1	1	1	0	1
1	1	0	0	1	1	0	1	1

(e) (i) (A' + 1) . B

= (0 + 1) . 0 = 0

(ii) (A + B') '

= (1 + 1)' = (1)' = 0

Question 2

(a) Differentiate between *throw* and *throws* with respect to exception handling. **[2]**

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

E* (F / (G – H) * I) + J

(c) Write the algorithm for push operation (to add elements) in an array based stack. **[2]**

(d) Name the File Stream classes to: **[2]**

(i) Write data to a file in binary form.

(ii) Read data from a file in text form.

(e) A square matrix M [ ][ ] of size 10 is stored in the memory with each element requiring 4 bytes of storage. If the base address at M [0][0] is 1840, determine the address at M [4][8] when the matrix is stored in **Row Major Wise**. **[2]**

Comments of Examiners

(a) Most of the candidates were not aware of the term 'throw'. They wrote about 'throws' with respect to exceptional handling. Some used examples while others explained in their own words.

(b) Most candidates were able to solve this problem correctly. Some candidates wrote the correct answer without showing the working. Brackets were ignored as operator precedence was not clear in some cases. Some applied the postfix correctly, but could not derive the final answer.

(c) Some candidates wrote the function with syntax while others wrote in their own language. Stack overflow condition was not mentioned in some cases. Important steps were missing which includes the increment of the top pointer variable.

(d) (i) Several candidates found it difficult to recollect the exact file stream class and gave various answers related to file handling.
(ii) Same as above

(e) Candidates who had 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.

Suggestions for teachers

- Exceptional handling should be covered widely with all types of exceptions and emphasis should be laid on its use in programming using the terms 'throw' and 'throws' with examples.
- Examples need to be practiced with conversion of Infix to Postfix notation, the order of precedence; also, the Polish stack method should be taught.
- Candidates should be given more practice to write algorithms in any form i.e. pseudo codes, standard form etc. Checking for overflow for push operation and underflow for pop operation (LIFO / FIFO) must be explained.
- File handling should be taught using a tabular / chart form including the various types of files and their streams required

– Practice should be given in understanding the formula using row major and column major and to make the subject of formula that is required in the question. Diagrammatical explanation of row and column major with respect to two dimensional array must be practiced.

MARKING SCHEME

Question 2.

(a) Throw : - used to explicitly raise a exception within the program, the statement would throw new exception.

Throws : - this clause is used to indicate the exception that are not handled by the method.

(b) E * (F / (G – H) * I) + J

E * (F / G H – * I) + J

E * F G H – / I * + J

E F G H – / I * * J +

(c) Step 1 : Start

Step 2 : if top >= capacity then OVERFLOW, Exit

Step 3 : top = top+1

Step 4 : Stack[top] = value

Step 5 : Stop

(d) (i) FileOutputStream / DataOutputStream/ FileWriter/ OutputStream

(ii) FileReader / DataInputStream/ InputStream/ FileInputStream

(e) Row Major address formula : **M[i][j] = BA + W [(i – lr) * column + (j – lc)]**

BA = 1840, lr = 0 , lc = 0 , W = 4 , rows = 10 , column = 10 , i = 4 , j = 8

M[4][8] = 1840 + 4 [(4 - 0) x 10 + (8 – 0)]

= 1840 + 192

= **2032**

Question 3

(a) The following function **Recur ( )** is a part of some class. What will be the output of the function **Recur** ( ) when the value of n is equal to 10. Show the dry run / working. **[5]**

```
void  Recur (int n)
{    if (n > 1 )
        {     System.out.print ( n + "  " );
              if (n%2 !=0)
              {
                   n = 3 * n+1;
                   System.out.print(n + "  ");
               }
              Recur (n/2);
         }
}
```

(b) The following function is a part of some class. Assume 'n' is a positive integer. Answer the given questions along with dry run / working.

```
int unknown (int n)
{
     int i, k;
     if  (n%2==0)
     {     i = n/2;
           k=1;
      }
      else
      {
           k=n;
           n– –;
           i=n/2;
       }
       while (i > 0)
       {
           k=k*i*n;
           i– –;
           n– –;
       }
       return k;
}
```

(i) What will be returned by unknown(5)? **[2]**

(ii) What will be returned by unknown(6)? **[2]**

(iii) What is being computed by unknown (int n)? **[1]**

Comments of Examiners

(a) Most of the candidates answered this part correctly. Some candidates were not clear with the concept of recursive technique. A few did not show the working / dry run. Some were confused with the odd and even numbers and solved only half the output i.e. incomplete answer. A few candidates were not able to solve the back tracking part (LIFO). In several cases, the output was shown vertical instead of horizontal. A few candidate gave the answer in reverse order.

(b) (i) Most of the candidates answered well and scored full credit. Some did not show the working and wrote the answer directly. Some could not calculate properly and lost track after the first step.

(ii) Same as above

(iii) This part was answered correctly by most of the candidates.

Suggestions for teachers

- Various techniques relating to problem solving using recursion should be given more practice. Output programs must be explained using diagrams for each function call. Memory blocks can be used to represent each function call.
- Practice should be given on program using conditions / looping and other output related programs. Teachers should show the dry run/ working of program and emphasize that working is necessary to get full credit. Working should be done in a tabular form, calculating the values of each variable after each iteration.

MARKING SCHEME

Question 3.

(a)
```
Recur ( 10 )
10  Recur ( 5 )
        5
        16  Recur ( 8 )
              8  Recur ( 4 )
                    4  Recur ( 2 )
                          2  Recur ( 1 )
```

OUTPUT : 10 5 16 8 4 2

(b) (i) 120

(ii) 720

(iii) calculate factorial/ product

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: **$F(A,B,C,D) = \Sigma (0, 2, 4, 5, 8, 9, 10, 12, 13)$**

(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 the Boolean function: **$F(P,Q,R,S) = \pi (0, 1, 3, 5, 7, 8, 9, 10, 11, 14, 15)$**

(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) (i) Many 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. Several candidates drew the K-Map incorrectly. Some marked the group as pairs instead of quads. A number of candidates included the redundant group in the final expression
(ii) Several candidates drew the logic circuit using NAND gates while some others drew vague diagrams. In some cases, the gates were not in proper shape and the logic diagram was not labeled.

(b) (i) Several 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.
(ii) Many candidates drew the logic circuit using NOR gates while some others drew vague diagrams.

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 with the highest reducing group marked first. Redundant groups are not to be included in the final reduced expression.
- More and more practice should be given in drawing logic circuits using basic gates and also with universal gates,
- Make students reduce POS and SOP expressions using K-Map simultaneously.

MARKING SCHEME

Question 4.

(a) $F(A,B,C,D) = \sum (0, 2, 4, 5, 8, 9, 10, 12, 13)$

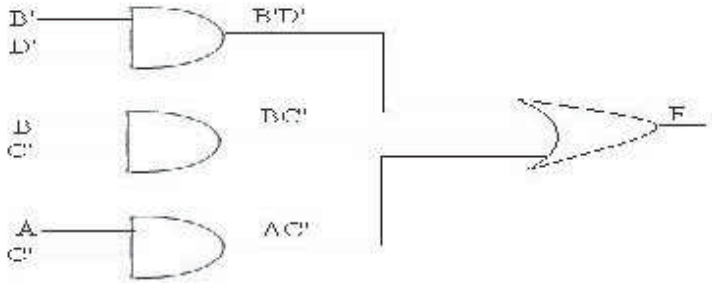
	C'D'	**C'D**	**CD**	**CD'**
A'B'	0 **1**	1 0	3 0	2 **1**
A'B	4 **1**	5 **1**	7 0	6 0
AB	12 **1**	13 **1**	15 0	14 0
AB'	8 **1**	9 **1**	11 0	10 **1**

There are three quads:

Quad1 $(m_0 + m_2 + m_8 + m_{10})$ = B'D' Quad2 $(m_4 + m_5 + m_{12} + m_{13})$ = BC'

Quad3 $(m_8 + m_9 + m_{12} + m_{13})$ = AC'

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



(b) F(P,Q,R,S) = **π (0, 1, 3, 5, 7, 8, 9, 10, 11, 14, 15)**

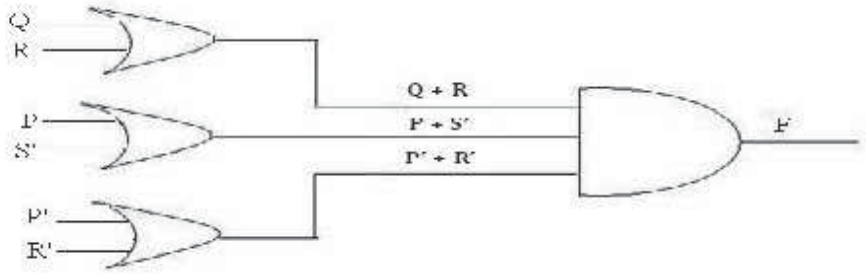
	R+S	R+S'	R'+S'	R'+S
P+Q	0 **0**	1 **0**	3 **0**	2 1
P+Q'	4 1	5 **0**	7 **0**	6 1
P'+Q'	12 1	13 1	15 **0**	14 **0**
P'+Q	8 **0**	9 **0**	11 **0**	10 **0**

There are three quads:

Quad 1 ($M_0 M_1 M_8 M_9$) = Q + R Quad 2 ($M_1 M_3 M_5 M_7$) = P + S'

Quad 3 ($M_{10} M_{11} M_{14} M_{15}$) = P' + R'

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



Question 5

A Football Association coach analyzes the criteria for a win/draw of his team depending on the following conditions.

- If the Centre and Forward players perform well but Defenders do not perform well.

OR

- If Goal keeper and Defenders perform well but the Centre players do not perform well.

OR

- If all the players perform well.

The inputs are :

INPUTS	
C	Centre players perform well.
D	Defenders perform well.
F	Forward players perform well.
G	Goalkeeper performs well.

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

Output: **X** - Denotes the win/draw criteria [1 indicates win/draw and 0 indicates defeat in all cases.]

(a) Draw the truth table for the inputs and outputs given above and write the **POS** expression for X(C, D, F, G). **[5]**

(b) Reduce **X (C, D, F, G)** using Karnaugh's Map. **[5]**
Draw the logic gate diagram for the reduced **POS** expression for X (C, D, F, G) 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) While a number of candidates answered well, some did not mention the final expression. Several candidates were confused with the POS expression and took the output with 1's instead of 0's. Some took 0's as outputs but wrote the minterms instead of maxterms.

(b) Some candidates were not able to draw the K-Map for the POS expression correctly. Some drew the K-Map of SOP, but grouping and reducing was done in POS. For a number of candidates the "Map rolling" concept was not very clear. Several candidates converted the canonical form to cardinal form and then reduced it. Some candidates did not draw the logic circuit.

Suggestions for teachers

- Student should be told to read the question carefully and answer accordingly so that no part is left unanswered. More practice should be given to derive SOP and POS expression from any given truth table (i.e. Minterms and Maxterms).
- Make students reduce POS and SOP expressions using K-Map simultaneously. Students should be asked to read the question carefully (i.e. SOP or POS) and not to include the redundant group in the final expression.

MARKING SCHEME

Question 5.

(a)

C	D	F	G	X
Centre players perform well	Defenders perform well	Forward players perform well	Goalkeeper perform well	**OUTPUT**
0	0	0	0	**0**
0	0	0	1	**0**
0	0	1	0	**0**
0	0	1	1	**0**
0	1	0	0	**0**
0	1	0	1	1
0	1	1	0	**0**
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	**0**
1	1	0	1	**0**
1	1	1	0	**0**
1	1	1	1	1

X (A,B,C,D) $= \pi$ (0, 1, 2, 3, 4, 6, 8, 9, 12, 13, 14)

(b)

	F + G	**F + G'**	**F' + G'**	**F'+G**
C + D	0 **0**	1 **0**	3 **0**	2 **0**
C + D'	4 **0**	5 1	7 1	6 **0**
C'+ D'	12 **0**	13 **0**	15 1	14 **0**
C'+ D	8 **0**	9 **0**	11 1	10 1

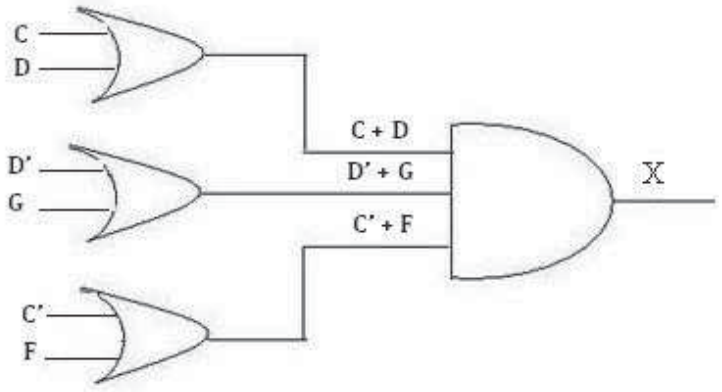
There are three quads:

Quad 1 ($M_0\ M_1\ M_2\ M_3$) = C + D

Quad 2 ($M_4\ M_6\ M_{12}\ M_{14}$) = D' + G

Quad 3 ($M_8\ M_9\ M_{12}\ M_{13}$) = C' + F

Hence X (C, D, F, G) = (C + D) . (D' + G) . (C' + F)



Question 6

(a) In the following truth table x and y are inputs and B and D are outputs: **[3]**

INPUT		OUTPUT	
x	y	B	D
0	0	0	0
0	1	1	1
1	0	0	1
1	1	0	0

Answer the following questions:

(i) Write the SOP expression for D.

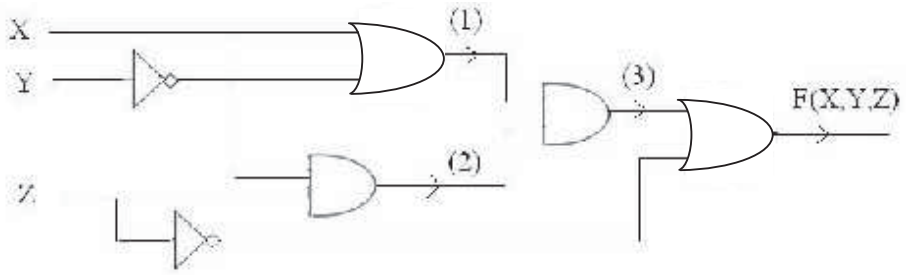
(ii) Write the POS expression for B.

(iii) Draw a logic diagram for the SOP expression derived for D, using only NAND gates.

(b) Using a truth table, verify if the following proposition is valid or invalid: **[3]**

$(a => b) \wedge (b => c) = (a => c)$

(c) From the logic circuit diagram given below, name the outputs (1), (2) and (3). Finally derive the Boolean expression and simplify it to show that it represents a logic gate. Name and draw the logic gate. **[4]**



Comments of Examiners

(a) (i) Some candidates wrote the POS expression for D. Some interchanged 0's with 1's and wrote the minterms for all outputs.
(ii) This part was answered correctly by most of the candidates. A few candidates wrote the SOP expression for B. Some gave the incomplete expression.
(iii) Most of the candidates answered this part correctly barring a few who drew vague diagram. Some candidates drew the logic diagram with multiple use of NAND gates.

(b) Most of the candidates answered this part well except for a few who were not aware of proposition statements and the terms and symbols used in proposition. Some candidates tried to make it equal and wasted time.

(c) The first three sub parts were answered correctly by almost all the candidates, but the final expression and its reducing was confusing to some. Some candidates were not able to arrive at the final expression and reduce it. The final gate symbol for the reduced expression (x + z') was not clear / ambiguous to some of the candidates.

Suggestions for teachers

- Deriving both SOP and POS expressions from truth table should be given more practice. Students should be told that 0(zero) output columns are for Maxterms and 1(one) output columns are for Minterms..
- More practice should be given to draw logic circuits using Universal gates (NOR and NAND). Teach students that in SOP expression the logic diagram drawn with basic gates (AND, OR, NOT) can be replaced by NAND gates and in POS expression the basic gates can be replaced by NOR gates directly.
- Propositional logic should be taught using all types of symbols, terms etc. that are used to solve a proposition and to state its validity
- Deriving expression from a logic diagram and reducing it must be given more practice. The basic gates i.e. AND, OR and NOT gate must be explained in detail.

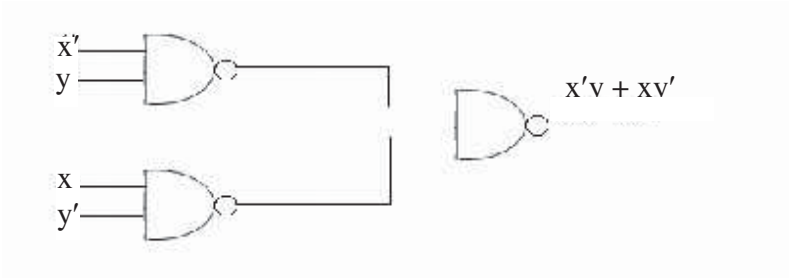
MARKING SCHEME

Question 6.

(a) (i) $D(x,y) = x'y + xy'$ OR $D(x,y) = \sum(1,2)$ OR $X \oplus Y$

(ii) $B(x,y) = (x+y)\cdot(x'+y)\,(x'+y')$ OR $B(x,y) = \pi\,(0,2,3)$

(iii)



(b) **(a => b) Λ (b => c) = (a => c)**

a	b	c	a => b	b => c	L.H.S.	a => c
0	0	0	1	1	1	1
0	0	1	1	1	1	1
0	1	0	1	0	0	1
0	1	1	1	1	1	1
1	0	0	0	1	0	0
1	0	1	0	1	0	1
1	1	0	1	0	0	0
1	1	1	1	1	1	1

The proposition is **invalid**

(c) (1) X + Y'

(2) X.Z

(3) (X + Y') . XZ

Final Expression F(X,Y,Z) = (X + Y') . XZ + Z'

= XZ + XY'Z + Z'

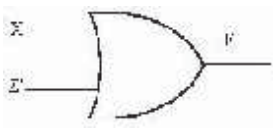
= XZ (1 + Y') + Z'

= XZ + Z'

= (X+Z')·(Z+Z')

= X + Z'

The gate is OR gate.



Question 7

(a) What are Decoders? How are they different from Encoders? **[2]**

(b) Draw the truth table and a logic gate diagram for a 2 to 4 Decoder and briefly explain its working. **[4]**

(c) A combinational logic circuit with three inputs P, Q, R produces output 1 if and only if an odd number of 0's are inputs. **[4]**

(i) Draw its truth table.

(ii) Derive a canonical SOP expression for the above truth table.

(iii) Find the complement of the above derived expression using De Morgan's theorem and verify if it is equivalent to its POS expression.

Comments of Examiners

(a) The term 'combinational circuit' was not used in the definition by many candidates. Some were able to give suitable examples of their uses / applications / differences. In some cases, the differences were given but no definition of decoder was given. Several candidates explained the multiplexer instead of decoders.

(b) The logic diagram was well answered. Some candidates drew the OR gate instead of AND gate. Some used the OR gate to combine parallel lines to a single serial line which was not required. There was no labeling in some cases. The working / explanation was not shown in some of the cases.

(c) (i) This part was well answered by most of the candidates. A few candidates drew the wrong truth table with improper combinations.
(ii) Several candidates gave incomplete expression. Some wrote the minterms for all outputs.
(iii) Some candidates were confused as the complement was not equal / equivalent to its POS expression and wasted time in proving it unnecessarily.

Suggestions for teachers

- Encourage students to write the correct definition along with their use.
- The differences between encoders, decoders and multiplexers must be explained clearly.
- Practice should be given in drawing the decoders along with the expression, truth table and logic circuit / diagram.
- Candidates should be asked to read the question carefully and answer accordingly.
- Practice should be given in drawing 3-variable and 4-variable truth table combinations according to the gray coding. Odd and even parity must be explained in detail.
- More practice should be given in deriving the SOP and POS expression from the truth table.
- Complement of an expression should be taught in detail. Differences between duality and complementation must be explained with examples.

MARKING SCHEME

Question 7.

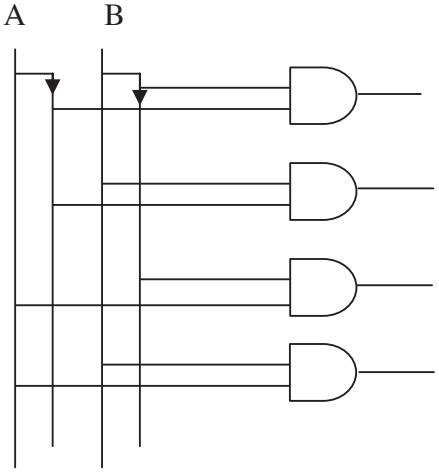
(a) Decoders are combinational circuit which inputs 'n' lines and outputs 2^n or fewer lines.

Encoders convert HLL to LLL i.e. Octal, Decimal and Hexadecimal to binary where as Decoders convert LLL to HLL i.e. Binary to Octal, Decimal and Hexadecimal.

(b)

Inputs		Outputs				X
A	B	d_0	d_1	d_2	d_3	
0	0	1	0	0	0	0
0	1	0	1	0	0	1
1	0	0	0	1	0	2
1	1	0	0	0	1	3

Truth Table for 2 to 4 decoder



Cicuit diagram for 2 to 4 decoder

Working : If any number is required as output then the inputs should be the binary equivalent. For example, if the input is 01 (A'.B) then the output is 1 and so on.

(c) (i)

P	Q	R	X (OUTPUT)
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

(ii) X (P,Q,R) = P'Q'R' + P'QR + PQ'R + PQR'

(iii) Complement of X (P,Q,R) = (P+Q+R) . (P+Q'+R') . (P'+Q+R') . (P'+Q'+R) which is not equal to POS expression for the above Truth Table.

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

An emirp number is a number which is prime backwards and forwards. Example: 13 and 31 are both prime numbers. Thus, 13 is an emirp number. **[10]**

Design a class **Emirp** to check if a given number is Emirp number or not. Some of the members of the class are given below:

Class name	:	**Emirp**
Data members/instance variables:		
n	:	stores the number
rev	:	stores the reverse of the number
f	:	stores the divisor
Member functions:		
Emirp(int nn)	:	to assign n = nn, rev = 0 and f = 2
int isprime(int x)	:	check if the number is prime using the **recursive technique** and return 1 if prime otherwise return 0
void isEmirp( )	:	reverse the given number and check if both the original number and the reverse number are prime, by invoking the function isprime(int) and display the result with an appropriate message

Specify the class **Emirp** giving details of the **constructor(int), int isprime(int)** and **void isEmirp( ).** Define the **main( )** function to create an object and call the methods to check for Emirp number.

Comments of Examiners

Common errors made by candidates were: the concept of recursion was not clear to some candidates; some took the parameter 'x' as the number while some assumed it as a factor. The data member 'f' was declared as double and extra data members were included. Some wrote the function isprime() without using the recursive technique. Several candidates did not invoked the function isprime() to check for prime in the method isEmirp(). The object creation in the main() function was incorrect in some cases. The other function including the constructor as well answered. A few candidates did not write the main() function.

Suggestions for teachers

– More practice should be given in solving programs using recursive techniques. Much attention should be paid by the teachers towards recursion and its techniques with examples. Knowledge of base case and recursive case should be given to the students for every program using recursive technique. Invoking function within another function should be given more practice. It should be emphasized that constructors are invoked automatically and cannot return any value.

MARKING SCHEME

Question 8.

```
import java.util.Scanner;
public class Emirp
{
  int n,rev,f;
  Emirp(int nn)
  {   n=nn;
      rev=0;
      f=2;
   }
  int isprime(int x)
  {
      if(n==x)
        {
           return 1;
         }
      else if(n%x==0 ||n==1)
       {
          return 0;
       }
      else
        return isprime(x+1);
  }
  void isEmirp( )
  {
```

```
      int x=n;
      while(x!=0)
      {
         rev=(rev*10) + x%10;
         x=x/10;
       }
      int ans1=isprime(f);
      n=rev;
      f=2;
      int ans2=isprime(f);
      if(ans1==1 && ans2==1)
        System.out.println(n+ "  is an Emirp number");
      else
        System.out.println(n+ "  is not an Emirp number");
    }
   public static void main()
   {  Scanner sc=new Scanner(System.in);
      System.out.println("\n Enter a number " );
      int x=sc.nextInt();
     Emirp obj = new Emirp(x);
     obj.isEmirp();
   }
}
```

Question 9

Design a class **Exchange** to accept a sentence and interchange the first alphabet with the last alphabet for each word in the sentence, with single letter word remaining unchanged. The words in the input sentence are separated by a single blank space and terminated by a full stop. **[10]**

Example: Input: It is a warm day.

Output: tI si a marw yad

Some of the data members and member functions are given below:

Class name	:	**Exchange**
Data members/instance variables:		
sent	:	stores the sentence
rev	:	to store the new sentence
size	:	stores the length of the sentence
Member functions:		
Exchange( )	:	default constructor
void readsentence( )	:	to accept the sentence

void exfirstlast( )	:	extract each word and interchange the first and last alphabet of the word and form a new sentence rev using the changed words
void display( )	:	display the original sentence along with the new changed sentence.

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

Comments of Examiners

Most of the candidates answered correctly. Different methods / logic were used to extract the words from the sentence. Some used String tokenizer while others used the split function and charArray to separate words from the sentence. In some cases, the words were exchanged / reversed instead of exchanging the first and last alphabet. Reframing the sentence after exchanging the first and last character was not done properly in several cases. A number of candidates did not define the default constructor. Some used the replace() function which was incorrect. A few candidates did the entire program without answering the exfirstlast() function. The main() function and constructor were not answered by a number of candidates.

Suggestions for teachers

– Practice should be given in extracting characters from words, words from sentences and sentences from paragraphs. Different methods /logic should be adopted so that wider exposure to string manipulation related programs is given to the candidates. Knowledge of constructors to initialize a string and other data members should be given.

MARKING SCHEME

Question 9.

```
import java.util.*;
public class Exchange
{
  String sent,rev;
  int size;
  Exchange()
  { sent=null;
    rev="";
  }
  void readsentence()
  {  Scanner sc=new Scanner(System.in);
     System.out.print("\n Enter a sentence  ");
     sent=sc.nextLine();
     size=sent.length();
  }
  void exfirstlast()
  {
```

```
   int p=0; char ch; String b;
   for(int i=0;i<size;i++)
    { ch=sent.charAt(i);
      if(ch ==' '||ch =='.')
       {  b=sent.substring(p,i);
          if(b.length() != 1)
           { rev += b.charAt(b.length()-1);
             rev = rev + b.substring(1,b.length()-1);
             rev += b.charAt(0);

           }
          else
              rev = rev + b;
         rev = rev + " ";
         p=i+1;
       }
    }
  }
  void display()
  { System.out.print("\n  Input:  " + sent);
    System.out.print("\n Output:  " + rev );
  }
  public static void main()
  { Exchange obj = new Exchange();
    obj.readsentence();
    obj.exfirstlast();
    obj.display();
  }
}
```

Question 10

A class **Matrix** contains a two dimensional integer array of order [m × n]. The maximum value possible for both 'm' and 'n' is 25. Design a class **Matrix** to find the difference of the two matrices. The details of the members of the class are given below: **[10]**

Class name	:	**Matrix**
Data members/instance variables:		
arr[ ][ ]	:	stores the matrix element
m	:	integer to store the number of rows
n	:	integer to store the number of columns

Member functions:

Matrix (int mm, int nn)	:	to initialize the size of the matrix m = mm and n = nn
void fillarray( )	:	to enter the elements of the matrix
Matrix SubMat(Matrix A)	:	subtract the current object from the matrix of parameterized object and return the resulting object
void display( )	:	display the matrix elements

Specify the class **Matrix** giving details of the **constructor(int,int), void fillarray( ), Matrix SubMat(Matrix)** and **void display**. Define the **main( )** function to create objects and call the methods accordingly to enable the task.

Comments of Examiners

Common errors made by candidates while attempting this question were: (i) passing objects to the function and accessing its members by the dot operator (ii) creating a new local object and returning the object. Several candidates re-declared and created the array in the constructor. In some cases, 3 different arrays were created instead of creating objects. Some candidates used the same object to store the answer without creating a local object. Printing of the array in matrix form was not done properly in a number of cases. The main() function was incomplete in some cases as the objects were not created nor the function was called properly.

Suggestions for teachers

- Passing objects to functions must be explained clearly.
- It must be emphasized that only the mentioned technique should be written.
- More practice should be given in writing the main function with each and every program. Practice should be given in both single and double dimension programs.
- Instruct the students to read the question properly and answer accordingly.

MARKING SCHEME

Question 10.

```
import java.util.Scanner;
public class Matrix
{    static Scanner sc=new Scanner(System.in);
     int arr[][]=new int[25][25];
     int m,n;
     Matrix(int mm,int nn)
     { m=mm;
       n=nn;
     }
     void fillarray()
     {   System.out.print("\n Enter elements of array  ");
```

```
        for(int i=0;i<m;i++)
        { for(int j=0;j<n;j++)
              arr[i][j]=sc.nextInt();
         }
    }

    Matrix SubMat(Matrix A)
    {
      Matrix B=new Matrix(m,n);
      for(int i=0;i<m;i++)
       { for(int j=0;j<n;j++)
          B.arr[i][j]= arr[i][j] - A.arr[i][j];
         }
        return B;
       }
    void display()
    { for(int i=0;i<m;i++)
      { System.out.println();
        { for(int j=0;j<n;j++)
            System.out.print(arr[i][j] + " \t ");
        }       }     }
    public static void main()
    {   System.out.print("\n Size of array " );
       int x=sc.nextInt();
       int y=sc.nextInt();
       Matrix A=new Matrix(x,y);
       Matrix B=new Matrix(x,y);
       Matrix C=new Matrix(x,y);
       A.fillarray();
       B.fillarray();
       C=A.SubMat(B);
       C.display();
    }
}
```

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

A super class **Perimeter** has been defined to calculate the perimeter of a parallelogram. Define a subclass **Area** to compute the area of the parallelogram by using the required data members of the super class. The details are given below: **[10]**

Class name	:	**Perimeter**
Data members/instance variables:		
a	:	to store the length in decimal
b	:	to store the breadth in decimal
Member functions:		
Perimeter(…)	:	parameterized constructor to assign values to data members
double Calculate( )	:	calculate and return the perimeter of a parallelogram as 2 * (length + breadth)
void show( )	:	to display the data members along with the perimeter of the parallelogram
Class name	:	**Area**
Data members/instance variables:		
h	:	to store the height in decimal
area	:	to store the area of the parallelogram
Member functions:		
Area(…)	:	parameterized constructor to assign values to data members of both the classes
void doarea( )	:	compute the area as (breadth * height)
void show( )	:	display the data members of both classes along with the area and perimeter of the parallelogram.

Specify the class **Perimeter** giving details of the **constructor(**…**), double Calculate( )** and **void show( ).** Using the **concept of inheritance**, specify the class **Area** giving details of the **constructor(**… **), void doarea(**) and **void show( ).**

The main function and algorithm need not be written.

Comments of Examiners

The concept of Inheritance was not clear to several candidates. The keywords 'extends' and 'super' was missing in many answers. 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. Several candidates declared the base class data members as private. Double data members were not declared properly by some candidates. Invoking the show() function in the derived class was not answered properly. In some cases, the algorithm was written instead of a program. The rest of the functions were well answered.

Suggestions for teachers

- Practice should be given on inheritance to students. Use of constructor using the base class member should be made clear. The different visibility modes and their accessing capability should be made clear. Knowledge of calling the member function from the super class to the derived class must be clarified. Function overriding concept must be clearly explained and given more practice.

MARKING SCHEME

Question 11.

```
import java.util.*;
class Perimeter
{
   protected double a,b;
   Perimeter(double aa,double bb)
   {
       a=aa;
       b=bb;
    }
   double Calculate()
   {
        return (2*(a+b));
    }
   void show()
   {
         System.out.print("\n Length = " + a);
         System.out.print("\n Breadth = " + b);
         System.out.print("\n Perimeter =" + Calculate());
   }
}

import java.util.*;
class Area extends Perimeter
{
     double h;
     double area;
  Area(double aa, double bb, double cc)
  {
```

```
      super(aa,bb);
       h=cc;
   }
  void doarea()
  {
     area=super.b*h;
  }
  void show()
  {
       super.show();
       System.out.print("\n Height =  " + h);
       System.out.print("\n Area =  " + area);
  }

}
```

Question 12

A doubly queue is a linear data structure which enables the user to add and remove integers from either ends, i.e. from front or rear. Define a class **Dequeue** with the following details: **[10]**

Class name	:	**Dequeue**
Data members/instance variables:		
arr[ ]	:	array to hold up to 100 integer elements
lim	:	stores the limit of the dequeue
front	:	to point to the index of front end
rear	:	to point to the index of the rear end
Member functions:		
Dequeue(int l)	:	constructor to initialize the data members lim=1; front=rear=0
void addfront(int val)	:	to add integer from the front if possible else display the message (“Overflow from front”)
void addrear(int val)	:	to add integer from the rear if possible else display the message (“Overflow from rear”)
int popfront( )	:	returns element from front, if possible otherwise returns - 9999
int poprear( )	:	returns element from rear, if possible otherwise returns - 9999

Specify the class **Dequeue** giving details of the **constructor(int), void addfront(int), void addrear(int), int popfront( )** and **int poprear( ).**

The main function and algorithm need not be written.

Comments of Examiners

The concept of dequeue was not clear to most of the candidates. Common errors made by candidates were: (i) the condition / logic for underflow and overflow was not answered correctly (ii) increment / decrement of front and rear index was not done properly. Some candidates did not return values from popfront() and poprear() functions. Some wrote the queue program instead of dequeue. The methods addfront() and poprear() were found to be difficult by several candidates. Different approaches / methods / logic was used to solve the dequeue concept. The class declaration and constructors was well answered.

Suggestions for teachers

- More practice should be given 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).
- The concept of LIFO and FIFO must be explained to the students with lively examples related to real world. Implementation of stacks, queues and dequeues using arrays should be emphasized. Only the concept has to be explained taking the base as an array. It should be made clear to students that it is not an array related program which can be manipulated by shifting / inserting or initializing by any value since these data structures require pointers and pointers are not supported in java. So, the array is used to show the working of a stack, queue or a dequeue.

MARKING SCHEME

Question 12.

```
public class Dequeue
{
   int arr[] = new int[100];
   int lim,front,rear;
   Dequeue(int l)
   {
      lim=l;
      front=0;
      rear=0;
      arr=new int[lim];
   }
   void addfront(int val)
   {
       if(front>0)
           arr[front--]=val;
       else
           System.out.print("\n Overflow from front ");
```

```
    }
    void addrear(int val)
    {
        if(rear<lim-1)
            arr[++rear]=val;
        else
            System.out.print("\n Overflow from rear ");
    }
    int popfront()
    {
         if(front!=rear)
              return arr[++front];
         else
              return -9999;
    }
    int poprear()
    {
         if(front!=rear)
             return arr[rear--];
         else
             return -9999;
    }
}
```

Question 13

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

```
class Node
  {
     int item;
     Node next;
   }
```

Write an *Algorithm* **OR** a *Method* to count the number of nodes in the linked list. The method declaration is given below:

int count(Node ptr_start)

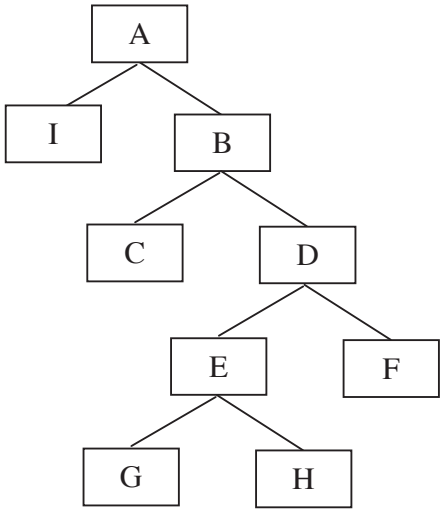
(b) What is the Worst Case complexity of the following code segment: **[2]**

(i)
```
for (int p = 0; p < N; p++)
    {
     for (int q=0; q < M; q++)
          {
             Sequence of statements;
          }
```

```
    }
for (int r = 0; r < X; r++)
    {
        Sequence of statements;
    }
```

(ii) How would the complexity change if all the loops went upto the same limit N?

(c) Answer the following from the diagram of a Binary Tree given below: **[4]**



(i) Preorder Transversal of tree.

(ii) Children of node E.

(iii) Left subtree of node D.

(iv) Height of the tree when the root of the tree is at level 0.

Comments of Examiners

(a) This part was well answered by most of the candidates. A few candidates had problem in moving the pointer to the next node and checking for null. Some wrote the algorithm in simple English language covering all the main steps.

(b) (i) A number of candidates were able to answer this part well. Some candidates did not mention the symbol 'O' in the final answer. A few candidates wrote only the definition of complexity while others mentioned (m*n) as dominant term instead.

Suggestions for teachers

- More programs / algorithms should be practiced with link list and binary tree data structure.
- Definition of complexities / big 'O' and the three cases of complexities must be explained in detail along with examples. The role of the dominant term in complexities must be explained.

(ii) Some candidates did not reduce the complexity by taking the dominant term. Some calculated correctly, but represented incorrectly. Candidates were not clear with the dominant term

(c) (i) Some candidates gave incomplete answers. Some were confused with Preorder and Inorder.

(ii) This part was well answered by most of the candidates.

(iii) Sub tree not clear as compared to nodes and only one tree child was mentioned in some cases.

(iv) Different answers were given by the candidates i.e. height of the tree can be 4 or 5.

- Explain binary tree with the different parts like root, nodes(internal and external), height, depth, level, size, tree traversal (preorder, inorder and postorder) etc.
- Candidates should be taught that height and level of a tree is same when the root is at level 0

MARKING SCHEME

Question 13.

(a) **Algorithm to count the number of nodes in a linked list.**

Steps :

1 - Start

2 - Set temporary pointer to first node and counter to 0.

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

4 - Increment the counter

5 - move temporary pointer to the next node

6 - Return the counter value

7 - End

Method to count for the number of nodes in a linked list

```
int count ( Node ptr_start )
{
      Node a = new Node(ptr_start);
      int c=0;
      while (a!=null )
       {   c++;
           a=a.next;
         }
       return c;

}
```

(b) (i) O(N x M) + O(X) **OR** O(NM + X)

(ii) $O(N^2)$ **OR** $O(N^2 + N) = O(N^2)$

(by taking the dominant term)

(c)	(i)	A, I, B, C, D, E, G, H, F
	(ii)	G and H
	(iii)	E G H
	(iv)	4

GENERAL COMMENTS:

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

- Canonical and Cardinal form of expressions and inter conversion.
- Keywords 'throw' and 'throws' with respect to Exceptional Handling.
- File Stream classes (both reading and writing data to files).
- K-MAPS (Grouping , map-rolling , place value)
- Redundant groups in K-Maps
- Passing objects to functions.
- Functions exfirstlast( ) and SubMat(Matrix A)
- Recursive technique and invoking function within another function (Nesting of functions).
- Dequeue operations for adding in front and removing from rear.

(b) Concepts between which candidates got confused:

- Canonical and Cardinal form of expression
- Difference between 'throw' and 'throws'.
- File handling stream classes
- Symbols of propositional logic
- Duality and complementation
- Deriving POS expression from Truth table
- Representing a single gate after reducing a logic circuit diagram.
- Passing of objects to functions.
- Invoking function within function.
- Written algorithms for inheritance program.
- Dominant term in complexity.

(c) Suggestions for Candidates:

- Prepare summary for each chapter or use high lighters to recognize the important terms and definitions.
- Answers and definitions should be short and precise and according to marks intended.
- Read the question properly and answer accordingly.
- 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.