ISC

YEAR 2020 EXAMINATION

Analysis of Pupil Performance

COMPUTER SCIENCE

Research Development and Consultancy Division
Council for the Indian School Certificate Examinations
New Delhi

Year 2020

***Published by*:**

Research Development and Consultancy Division (RDCD)

Council for the Indian School Certificate Examinations

Pragati House, 3rd Floor

47-48, Nehru Place

New Delhi-110019

Tel: (011) 26413820/26411706
E-mail: council@cisce.org

FOREWORD

This document of the Analysis of Pupils' Performance at the ISC Year 12 and ICSE Year 10 Examination is one of its kind. It has grown and evolved over the years to provide feedback to schools in terms of the strengths and weaknesses of the candidates in handling the examinations.

We commend the work of Mrs. Shilpi Gupta (Deputy Head) and the Research Development and Consultancy Division (RDCD) of the Council who have painstakingly prepared this analysis. We are grateful to the examiners who have contributed through their comments on the performance of the candidates under examination as well as for their suggestions to teachers and students for the effective transaction of the syllabus.

We hope the schools will find this document useful. We invite comments from schools on its utility and quality.

Gerry Arathoon
Chief Executive & Secretary

November 2020

PREFACE

The CISCE has been involved in the preparation of the ICSE and ISC Analysis of Pupil Performance documents since the year 1994. Over these years, these documents have facilitated the teaching-learning process by providing subject/ paper wise feedback to teachers regarding performance of students at the ICSE and ISC Examinations. With the aim of ensuring wider accessibility to all stakeholders, from the year 2014, the ICSE and the ISC documents have been made available on the CISCE website www.cisce.org.

The documents for the ICSE and ISC Examination Year 2020 include a detailed qualitative analysis of the performance of students in different subjects. The purpose of this analysis is to provide insights into how candidates have performed in individual questions set in the question paper. This section is based on inputs provided by examiners from examination centers across the country. It comprises of question wise feedback on the performance of candidates in the form of *Comments of Examiners* on the common errors made by candidates along with *Suggestions for Teachers* to rectify/ reduce these errors. The *Marking Scheme* for each question has also been provided to help teachers understand the criteria used for marking. Topics in the question paper that were generally found to be difficult or confusing by candidates, have also been listed down, along with general suggestions for candidates on how to prepare for the examination/ perform better in the examination.

The Analysis of Pupil Performance document for ICSE for the Examination Year 2020 covers the following subjects/papers: English (English Language, Literature in English), History and Civics, Mathematics, Physics, Chemistry, Commercial Studies and Environmental Science.

Subjects covered in the ISC Analysis of Pupil Performance document for the Year 2020 include English (English Language and Literature in English), Hindi, Physics, Chemistry, Mathematics, Computer Science, History, Political Science, Economics, Commerce, Accounts, and Environmental Science.

I would like to acknowledge the contribution of all the ICSE and the ISC examiners who have been an integral part of this exercise, whose valuable inputs have helped put this document together.

I would also like to thank the RDCD team of Dr. M.K. Gandhi, Dr. Manika Sharma, Mrs. Roshni George and Ms. Mansi Guleria, who have done a commendable job in preparing this document.

We hope that this document will enable teachers to guide their students more effectively and comprehensively so that students prepare for the ICSE/ ISC Examinations, with a better understanding of what is required from them.

Shilpi Gupta
Deputy Head - RDCD

November 2020

CONTENTS

PART I (20 Marks)

Answer **all** *questions*

Question 1

(a) State the properties of zero in Boolean algebra. **[1]**

(b) Find the complement of the following Boolean expression using De Morgan's law: **[1]**

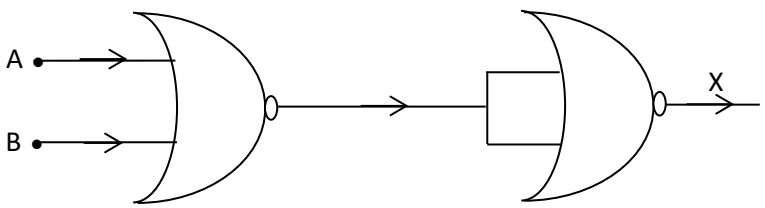
F(P,Q,R) = P + (Q′ • R)

(c) Find the dual of: (A′ + 0) • (B′ + 1) = A′ **[1]**

(d) State whether the following proposition is a tautology, contradiction or a contingency: **[1]**

F = (P => Q) V (Q => ~P)

(e) Study the diagram given below and answer the questions that follow: **[1]**



(i) Name the basic gate which is represented by the diagram.

(ii) What will be the value of X when A=1 and B=0 ?

Comments of Examiners	Suggestions for Teachers
(a) Most of the candidates answered well. Some could state only one property. In some cases, the properties of zero was explained in statements. The law written by a few of candidates had 0 however, it was not the expected law. (b) This part was well answered by most candidates. However, some wrote the direct answer without showing the working. A few modified the expression and then applied De Morgan's law. Some candidates forgot to change the sign or complements in the expression while applying the De Morgan's Law. Candidates seemed to be confused between Principle of Duality and De Morgan's law. In the final expression the bracket was missing in a few cases.	▪ *Practice all laws / properties of Boolean algebra and prove them with the help of the truth table. Advise students on writing all the properties and laws with their respective names.* ▪ *Teach students how to find the complement of an expression using De Morgan's law. This law is very useful in reducing / minimizing / simplifying a Boolean expression. Instruct students to not simplify the complemented expression.*

(c) Majority of the candidates answered this question correctly; nonetheless a few candidates made the following errors:

- The compliments were changed and the 0's and 1's were not interchanged.
- Only the definition was given.
- Found the dual of LHS and RHS separately instead of one single equation.
- Wrote the compliment instead of Dual or wrote the principle of duality.

(d) This part was well answered by most of the candidates. However, a few got confused by the symbols ~, Λ, V and =>. In some cases, candidates used the truth table for 3 variables instead of 2 variables. Also a few wrote one column (=>) incorrectly, and hence got the wrong answer. Some candidates proved using law instead of the Truth Table. A few forgot to mention the proposition being a tautology.

(e) This part was answered correctly by most candidates. Some got confused with the name of the basic gate. They wrote NOR gate as given in the diagram and not its representation. Some used the truth table to put the values and state the output.

- *De Morgan's Law and its application should be explained in detail.*
- *Teach the principal of duality to show that there exists another equation in every Boolean equation. Clarify the difference between complement and duality with examples.*
- *Advise students to practice all the laws of Boolean algebra and Propositional logic.*
- *Emphasise the importance of proving of all the laws.*
- *Explain the use of the symbols Λ, V, ~, => and <=> in a proposition.*
- *Give more practice in the concept of a tautology, contingency or contradiction. Teach Truth tables for conditional (=>) and bi-conditional (<=>) thoroughly. Consistent practice on Implication and Equivalence Law is a must.*
- *Give sufficient practice on how to simplify logic diagrams, especially using universal gates. Illustrate the working of an AND, OR and a NOT gate using the universal gates (NAND and NOR).*
- *Teach Truth tables for all the gates thoroughly in class.*

MARKING SCHEME

Question 1

(a)	Properties of zero: A + 0 = A and A • 0 = 0
(b)	F(P,Q,R) = P + (Q' • R) = (P + (Q' • R))' = P' • (Q' • R)' = P' • (Q + R')
(c)	Find the dual of : (A' + 0) • (B' + 1) = A' Dual = A'•1 + B'• 0 = A'

(d) F = (P=>Q) V (Q=> ~P)

P	Q	~P	P=>Q	Q=> ~P	(P=>Q) V (Q=> ~P)
0	0	1	1	1	1
0	1	1	1	1	1
1	0	0	0	1	1
1	1	0	1	0	1

(using laws) : (P' + Q) + (Q' + P') = 1 (as Q + Q' = 1)
Hence, it is a **TAUTOLOGY**

(e) (i) OR gate

(ii) 1

Question 2

(a) State the difference between a *Binary Tree structure* and a *single Linked List*. **[2]**

(b) A matrix B[10][20] is stored in the memory with each element requiring 2 bytes of storage. If the base address at B[2][1] is 2140, find the address of B[5][4] when the matrix is stored in **Column Major Wise**. **[2]**

(c) Convert the following infix notation to prefix form: **[2]**

(X + Y) / (Z * W / V)

(d) State the best case and the worst case complexity for bubble sort algorithm. **[2]**

(e) What is the significance of the keyword '**new**' in Java? Mention the areas where it is used. **[2]**

Comments of Examiners

(a) This part was well answered by most candidates. However, a few errors were made such as, vague differences between a *binary tree structure* and a *single linked list* were given, gave only the examples/diagrams without the definition or vice-versa.

(b) Several candidates gave correct answers. Some wrote the answer directly without showing the working/formula. A few made calculation mistakes, while some solved using Row major method instead of Column major method. In some cases, the values were not substituted

Suggestions for Teachers

- *Elaborate on the basic concepts of Java along with the programs in class. Emphasise that the definitions / differences should be precise and also supported by examples wherever possible. The concept of data structures such as, stacks, queues, link lists, and binary trees should be made clear.*
- *Give more practice in calculating addresses using Row major wise and Column major wise. Clarify the*

properly in the formula. The base address was not clear to some of the candidates, they assumed it to be b[0][0] by default.

(c) Most candidates were able to solve this problem correctly. However, some candidates wrote the correct answer without showing the working. Some applied the prefix concept correctly but, could not derive the final answer due to wrong operator precedence. BODMAS was followed in a few cases, but the concept of left-to-right was not followed. A few candidates converted the expression to postfix notation instead of prefix notation.

(d) This part was answered well by many candidates. However, some only gave examples to illustrate their answers. Some drew graphs to illustrate the best case and worst case complexities of bubble sort. A few candidates wrote the definitions of both best case and worst case complexity, however failed to represent both cases in big-O notation.

(e) Most of the candidates wrote correct definitions / examples to this question. Some gave vague answers not related to the topic. A few wrote the syntax of the keyword 'new' as significance. They were familiar with the use of the new operator in declaring array or creating an object but were not able to write the significance of the keyword '**new**' in Java.

different terms used in address calculations.

- *Explain diagrammatically the difference in accessing the elements in Row Major and Column major arrangement. Give students different types of problems instead of direct application of the formula.*
- *Different questions with examples need to be practiced with regard to conversion of Infix to prefix and postfix notation, the order of precedence and also the Polish Stack method. Teach Precedence rule/ order of operators to students. Clarify the difference between Postfix and Prefix to students.*
- *Explain with examples all the three cases of complexities with their factor that influence it. Explain the various programs where complexity differs, example: loops, nested loops, conditional, recursion, etc. Teach students the best case and worst case for the entire standard algorithm and how to calculate the complexity of any given code. Give students code snippets to calculate complexity.*
- *Revise class XI syllabus in class XII. Basic concepts of Java must be elaborated along with their programs. Teach the exact use of the keyword 'new'. Explain to students that definitions should be precise and supported with examples,if possible.*

MARKING SCHEME **Question 2**	
(a)	**Binary Tree**: A data structure in which each node has at most two sub nodes/children (left & right) **Linked list**: A linear collection of data elements whose order is not given by their physical placement in memory. Instead, each element points to the next. It is a data structure consisting of a collection of nodes which together represent a sequence.
(b)	**Column Major Wise**: B[i][j] = BA + W [(j – l_c)* row + (i – l_r] Putting the values: = 2140 + 2[(4-1) * 10 + (5-2)] = 2140 + 2[33] = 2140 + 66 = **2206**
(c)	Infix to Prefix : (X + Y) / (Z * W / V) = +XY / (* Z W / V) = +XY / /*ZWV = /+XY/*ZWV = **/+XY/*ZWV**
(d)	O(n): Best case for bubble sort algorithm. O(n^2): Worst case for bubble sort algorithm.
(e)	'**new**' is a dynamic memory allocation operator, which is used to allot memory to non-primitive data types (example: class, array etc.). Areas of use: To declare an object, to declare an array.

Question 3

[5]

The following function **check( )** is a part of some class. What will the function check( ) return when the value of (i) **n=25** and (ii) **n=10.** Show the dry run/ working.

```
int check(int n)
 {  if(n<=1)
        return 1;
    if( n%2==0)
        return 1 + check(n/2);
   else
       return 1 + check(n/2 + 1);
 }
```

Comments of Examiners	Suggestions for Teachers
This question was answered correctly by almost all the candidates. However, in a few cases, the concept of recursion was not clear. Some were unable to accumulate the result along with the recursive calling. In some cases, the intermediate result was returned with each recursive call. Most of the students could do the dry run well but made mistakes in returning the right answer as the final step. A few candidates showed the working but were unable to calculate the final sum. Some wrote the output directly without showing the working / steps and ended up losing marks. In some cases, vague workings / outputs were given.	▪ *Give sufficient practice on programs using recursion / conditions / looping and other output related programs. Advise students to show the dry run/ working of program and emphasize that working is necessary.* ▪ *Give more emphasis on Standard programs especially recursive programs. Practice should be given on writing base case, recursive case, and showing the working at each step of the program code. Use Memory blocks / tables to show the concept of a stack working for recursive programs.*

MARKING SCHEME

Question 3

(i) check(25)
1+ check(12+1)
1+ check(6+1)
1+ check(3+1)
1+ check(2)
1+ check(1)
1

Final output: (i) 1+1+1+1+1+1 = **6**

(ii) check(10)
1+ check(5)
1+ check(2+1)
1+ check(1+1)
1+ check(1)
1

(ii) 1+1+1+1+1 = **5**

PART – II (50 Marks)

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

SECTION – A

*Answer **any two** questions.*

Question 4

(a) Given the Boolean function: F(A, B, C, D) = Σ (0, 1, 2, 3, 4, 5, 8, 9, 10, 11, 12, 13,14).

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

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

(b) Given the Boolean function: F(A, B, C, D) = π (3, 4, 6, 9, 11, 12, 13, 14, 15).

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

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

Comments of Examiners	Suggestions for Teachers
(a) (i) Most candidates managed to fare well in this part. Some candidates were not able to draw the K-Map for the SOP expression correctly. Group identification was improper in some cases. "Map rolling" concept was not clear to a few candidates. In some cases, redundant groups were also included in the final expression which was not required. Some drew the POS K-Map instead of SOP, while some filled the K-map with 0's instead of 1's. A few candidates made mistakes in labelling the Karnaugh's map, due to which they got the final answer wrong. (ii) Most of the candidates gave correct answers. A few, however, drew the logic circuit using universal gates (NAND/NOR), while some drew vague diagrams with different shapes instead of basic gates. Labelling and flow lines were not shown in some cases.	▪ *Show students how to reduce SOP and POS expressions using K-Map simultaneously.* ▪ *Instruct students to not include the redundant group in the final expression.* ▪ *Practice should be given in drawing the K-Map, filling the K-Map with 0's and 1's, marking the groups and reducing the groups.* ▪ *Practice of Gray's coding in Binary number system along with the labelling of rows and columns must be given in class.* ▪ *Teach students how to draw logic circuits using basic gates (AND, OR, NOT) / universal gates (NAND, NOR).*

(b) (i) Most candidates answered this question correctly. Some candidates made errors in place value and putting of variables in K-Map or drew it incorrectly. In some cases, groups were reduced by laws or group identification was done incorrectly. Some candidates included the redundant group in the final expression. Some candidates drew the SOP K-Map instead of POS.

(ii) This part was answered well by most candidates. However, a few errors were committed which were the same as given in subpart (a) (ii).

- *While teaching, highlight the arranging of variables in proper order and the importance of cell values corresponding with the variables.*
- *A clear explanation how a group is framed and reduced should be given to students.*
- *Clarify the difference between POS and SOP using examples.*

MARKING SCHEME

Question 4

(a)(i)

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

There are two octets and one quads:

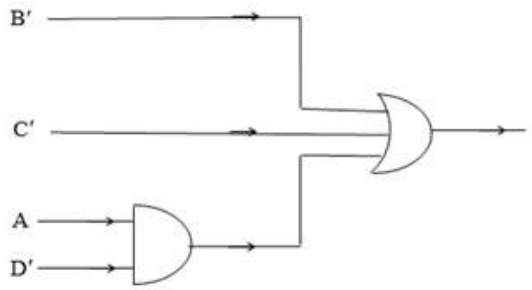
Octet1 ($m_0+m_1+m_2+m_3+m_8+m_9+m_{10}+m_{11}$) =**B′**

Octet2 ($m_0+m_1+m_4+m_5+m_8+m_9+m_{12}+m_{13}$) = **C′**

Quad ($m_8+m_{10}+m_{12}+m_{14}$) = **AD’**

Hence F (A, B, C, D) = **B’ + C’ + AD’**

(ii)



(b) (i)

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

There are two quads and one pair:

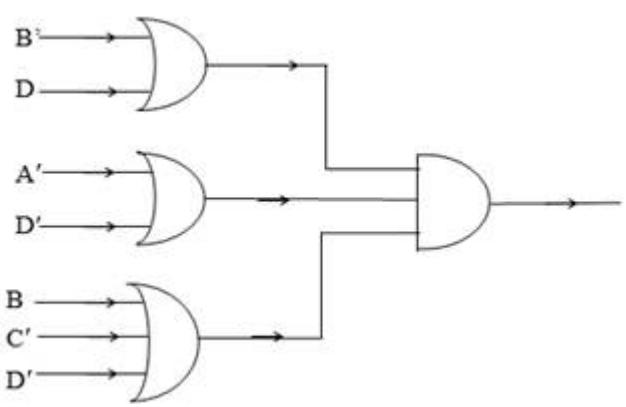
Quad 1 : ($M_4M_6M_{12}M_{14}$) = **B’ + D**

Quad 2 : ($M_9M_{11}M_{13}M_{15}$) = **A’ + D’**

Pair : (M_3M_{11}) = **B + C’ + D’**

Hence F(A, B, C, D) = **(B’ + D) • (A’ + D’) • (B + C’ + D’)**

(ii)



Question 5

(a) Draw the logic circuit diagram for an octal to binary encoder and explain its working when a particular digit is pressed. Also, state the difference between *encoders* and *decoders*. **[5]**

(b) Draw the circuit of a two input XOR gate with the help of NOR gates. **[3]**

(c) Convert the following expression to its cardinal SOP form: **[2]**

$F(P,Q,R) = P'Q'R + P'QR + PQ'R' + PQR'$

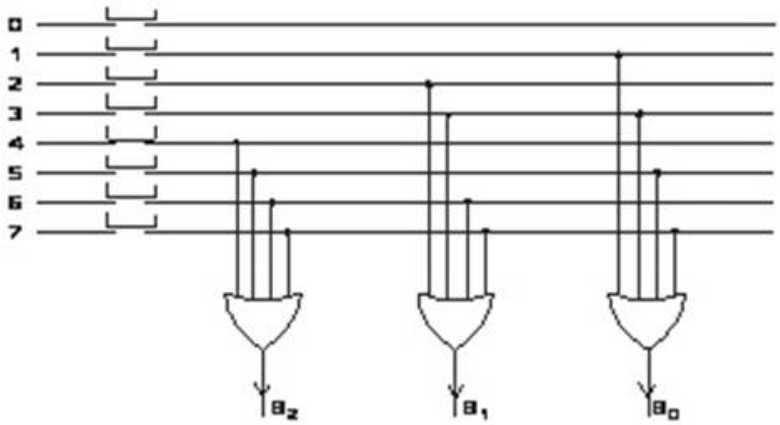
Comments of Examiners	Suggestions for Teachers
(a) Most of the candidates managed to fare well in this question. The logic diagram was well answered. However, a few candidates committed errors, such as: drew the decimal encoder instead of octal encoder, wrote the difference of encoder in place of decoder and vise-versa, used the AND gate instead of OR gate in the circuit diagram. In some cases, the working / explanation was not shown. Some candidates wrote the definitions of encoders and decoders separately to illustrate their difference while a few stated their respective uses. Some candidates drew four OR gates instead of three. In a few answers, the connection of lines to the OR gate were incorrect according to the values of input. (b) This part was answered well by most candidates. In a few cases, it was evident that candidates got confused using NOR gates in drawing circuits, they drew the expanded circuit instead of the simplified circuit. In some cases, an extra NOR gate was drawn which resulted in the wrong output. After deriving the basic expression of XOR gate, some candidates could not get the right equivalent diagram using NOR gate. (c) Most candidates gave correct answers. Some however, used the truth table to decode the cardinal values of the minterms, while others used the binary values directly. From a few answers, it was evident that candidates were not aware of the cardinal SOP form. Some gave vague answers despite the correct working.	▪ *Give sufficient Practice in drawing the encoders along with its expression, truth table and logic circuit / diagram.* ▪ *Ensure that students know the circuit diagram, truth table, expression, definition, and their uses for all applications of Boolean algebra, i.e. Half adder, Full adder, Encoders, Decoders, etc. The purpose and working of the gates for the said applications and the application of encoder and decoder must be discussed in class. All workings should be explained for each and every combinational circuit along with examples.* ▪ *Highlight the universality of gates (NAND as basic gates and NOR as basic gates) in class. The corresponding expressions using the universal gates (i.e. SOP expression with NAND and POS expression with NOR) must be taught to students. Teach students how to achieve basic gates from the universal gates.* ▪ *Give adequate Practice to convert canonical form of expression to its cardinal form and vice-versa.* ▪ *Explain Minterms and maxterms in detail.*

- *Teach students to fraw the truth table and to write the SOP and POS expression from truth table. Train students to convert Cardinal expression to Canonical expression and vice-versa.*

MARKING SCHEME

Question 5

(a) Circuit diagram of a octal to binary encoder:

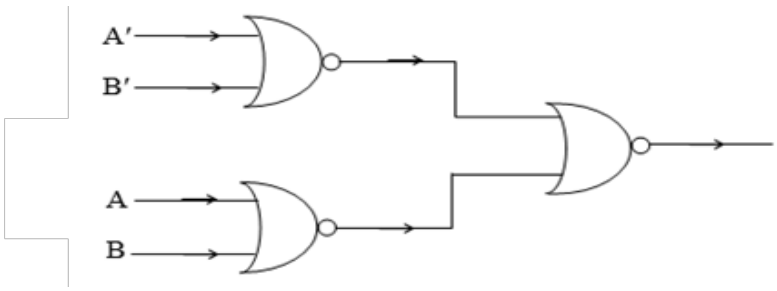


Working: It converts High Level to Low Level. Example: If input is $(5)_8$ the output will be $(101)_2$, i.e. binary value of 5 and so on. To convert the output signal to 1, a trigger is applied to the required gate otherwise it remains 0 as default.

An **encoder** is a combinational circuit which inputs 2^n or fewer lines and outputs 'n' lines. It converts HLL to LLL e.g. octal, decimal and hexadecimal to binary

A **decoder** is a combinational circuit which inputs 'n' lines and outputs 2^n or fewer lines. It converts LLL to HLL e.g. binary to octal, decimal and hexadecimal.

(b) XOR gate using only NOR gates: Expression for XOR in POS : F = (A'+B') • (A+B)



(c) F(P,Q,R) = P'Q'R + P'QR + PQ'R' + PQR'

= 001 011 100 110

F(P,Q,R) = ∑(1,3,4,6)

Question 6

(a) A company intends to develop a device to show the high-status power load for a household invertor depending on the criteria given below: **[5]**

- If Air conditioner and Geyser are on

OR

- If Air conditioner is off, but Geyser and Refrigerator are on

OR

- If Geyser is off, but Air conditioner and Water purifier are on

OR

- When all are on

The inputs are:

INPUTS	
A	Air conditioner is on
G	Geyser is on
R	Refrigerator is on
W	Water purifier is on

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

Output: **X** [1 indicates high power, 0 indicates low power for all cases]

Draw the truth table for the inputs and outputs given above and write the **SOP** expression for **X(A,G,R,W)**.

(b) Draw the truth table and derive an SOP expression for *sum* and *carry* for a full adder. Also, draw the logic circuit for the *carry* of a full adder. **[3]**

(c) Simplify the following expression using Boolean laws: **[2]**

$F = [(X' + Y) \bullet (Y' + Z)]' + (X' + Z)$

Comments of Examiners

(a) A number of candidates answered this part well. However, some failed to mention the final expression. A few got confused with the SOP/POS expression and interchanged 0's and 1's and vice-versa. Some candidates reduced the expression using K-Map which was not required.

(b) Most of the candidates answered this question well. A few errors were seen such as, the truth table and expression was drawn for a 'half adder' in place of a 'full adder', the logic circuits for both *sum* and *carry* were drawn, instead of the required *carry* expression. the block diagram was drawn

Suggestions for Teachers

- *Give regular practice of Truth table with 4 input variables i.e. 16 combinations. Explain Propositional logic to find the criteria for the output. Advise students to write the final expression in either Canonical or Cardinal form for both SOP and POS expressions. Teach students to derive POS or SOP expression from the truth table.*

instead of logic circuit. Some candidates were not able to derive the expression from the truth table and ended up writing the final simplified expression. *Sum* and *Carry* were converted to an SOP expression and the circuits drawn accordingly.

(c) Majority of the candidates answered this part correctly. In some cases, only the final answer was given without showing the working/steps or mentioning the laws. A few candidates wasted time using very lengthy steps for simplifying. In a few cases, the laws applied was incorrect while some were not able to reduce the expression completely as they did not have a clear idea of postulates and laws of Boolean algebra.

- *Clarify the difference between Half Adder and Full Adder and make them understand how to write the final expression for the circuit and draw the logic gate. Also, instruct students that a Full adder shows the sum of three bits and a Half adder shows the sum of only two bits.*
- *Reducing / simplifying expressions must be given more practice while ensuring the maximum laws are covered. Instruct students to mention laws along with their workings. Practice must be given to students on all the laws of Boolean algebra and Propositional logic.*

MARKING SCHEME

Question 6

(a)

A	G	R	W	X (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	0
0	1	1	0	1
0	1	1	1	1
1	0	0	0	0
1	0	0	1	1
1	0	1	0	0
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

SOP Expression: X (A,G, R, W) = ∑ (6, 7, 9, 11, 12, 13, 14, 15) OR
X = A'GRW' + A'GRW + AG'R'W + AG'RW + AGR'W' + AGR'W + AGRW' + AGRW

(b) Truth Table for a Full adder:

A	B	C	S	C_0
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

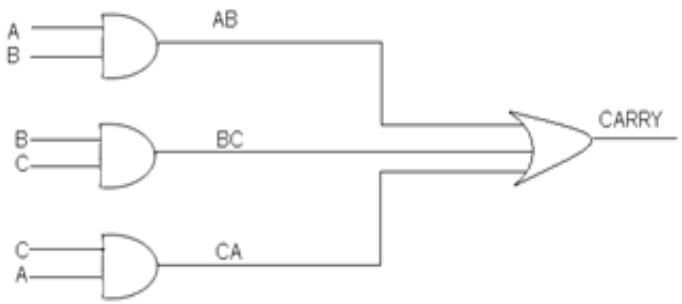
SOP Expression: Sum = A'B'C + A'BC' + AB'C' + ABC

$\sum (1,2,4,7)$

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

$\sum(3,5,6,7)$

Circuit for carry: C_0 = AB + BC + AC



(c) Simplify :[(X'+Y)•(Y'+Z)]' + (X'+Z)

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

= XY' + YZ' + X' + Z

= X' + XY' + Z + YZ'

= (X'+X)•(X'+Y')+(Y+Z)•(Z+Z')

= X'+Y'+Z +Y

= 1 as (Y + Y' =1)

SECTION – B

*Answer **any two** questions.*

Question 7

Design a class **Convert** to find the date and the month from a given day number for a particular year. **[10]**

Example: If day number is 64 and the year is 2020, then the corresponding date would be:

March 4, 2020 i.e. (31 + 29 + 4 = 64)

Some of the members of the class are given below:

Classname	:	**Convert**
Data members/instance variables:		
n	:	integer to store the day number
d	:	integer to store the day of the month (date)
m		integer to store the month
y	:	integer to store the year
Methods/Member functions:		
Convert ( )	:	constructor to initialize the data members with legal initial values
void accept( )	:	to accept the day number and the year
void day_to_date( )	:	converts the day number to its corresponding date for a particular year and stores the date in 'd' and the month in 'm'
void display( )	:	displays the month name, date and year

Specify the class **Convert** giving details of the **constructor( )**,void accept( ), **void day_to_date( )** and **voiddisplay( )**. Define a **main( )** function to create an object and call the functions accordingly to enable the task.

Comments of Examiners	Suggestions for Teachers
A number of candidates answered this question well. A few candidates used additional instance variables, which is not in accordance with the question. Some of the candidates accepted all the instance variables (month name, date, day, and year) as inputs which was not required. Various methods/logic were used to convert the day number to date in the method day_to_date(). Most of	▪ *Give more practice in class on number logic and working with numbers. Teach the concept of storing data in an array and accessing them using a loop. Give questions based on date logic, time logic, etc. to the students for*

the candidates used a longer process to calculate the date and have used 'if-else if' instead of array. Some did not include a leap year check in their program. A few candidates used a separate local string variable to store the month name. The format for display given in the question was not used by some, instead they gave the output directly in the display() method. The other methods including the object creation and method calling in the main( ) was answered corectly. Documentation/comments were missing in some cases.

regular practice. Teach them the order in which the function needs to be called. Advise students to read the question carefully and answer as directed. Explain the usage of arrays in storing certain information, which makes the implementation of logic very simple and small.

MARKING SCHEME

Question 7

```
import java.util.*;
class Convert
{    int n,d,m,y;
     Convert( )
     {
         n=0;
         y=0;
      }
      void accept()
      {
          Scanner x=new Scanner(System.in) ;
          System.out.println(("Enter day number and year") ;
          n=x.nextInt() ;
          y=x.nextInt() ;
       }
       void day_to_date()
       {
           int a[]={0,31,28,31,30,31,30,31,31,30,31,30,31};
           if(y%4==0)
               a[2]=29;
           int s=0, c=0;
           while(s<n)
               s=s+a[c++];
           s=s-a[--c];
           d=n-s;
           m=c;
       }
       void display()
       {    String x[]={"","JANUARY","FEBRUARY","MARCH","APRIL","MAY",
                        "JUNE","JULY","AUGUST","SEPTEMBER","OCTOBER",
```

```
                        "NOVEMBER","DECEMBER"};
        System.out.print("\n Day Number: " + n);
        System.out.print("\n Date: " );
        System.out.print(x[m]+" " +d + "," + y);
    }
    static void main()
    {
          Convert obj =new Convert();
          obj.accept( ) ;
          obj.day_to_date();
          obj.display();
     }
}
```

Question 8

Design a class **BinSearch** to search for a particular value in an array. **[10]**

Some of the members of the class are given below:

Classname	:	**BinSearch**
Data members/instance variables:		
arr[ ]	:	to store integer elements
n	:	integer to store the size of the array
Member functions/methods:		
BinSearch(int nn)	:	parameterized constructor to initialize n=nn
void fillarray( )	:	to enter elements in the array
void sort( )	:	sorts the array elements in ascending order using any standard sorting technique
int bin_search(int l,int u,int v)	:	searches for the value 'v' using **binary search** and **recursive technique** and returns its location if found otherwise returns -1

Define the class **BinSearch** giving details of the **constructor( ), void fillarray( ), void sort( )** and **int bin_search(int,int,int)**. Define the **main( )** function to create an object and call the functions accordingly to enable the task.

Comments of Examiners

Most of the candidates managed to answer this question well. In some cases, additional instance variables were used which is not in accordance with the question. A few candidates did not declare memory to the array using the 'new' operator. The concept of recursion was not clear to some candidates. Also, the parameters in the method bin_search() was not properly understood by a few. Some candidates were unable to apply the sorting technique correctly. Recursive case was improper, and the return statement was missing in some answers. Some candidates attempted the recursive method without using the recursive technique and used iteration (loops) instead. A few had problems in sort() method and used vague techniques for sorting. The other methods including the constructor were well answered by many. In some cases, object creation and method calling was not done properly in the main() function. A few did not write the main() method. Documentation/comments were missing in some cases.

Suggestions for Teachers

- *Give adequate practice to solve programs using recursive techniques. 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.*
- *Explain the difference between iteration and recursion.*
- *Give knowledge of instance variables and their accessibility in the class.*
- *Advise students to read the question and answer accordingly what is required and not to apply their own logic which will change the question requirement. Calling the methods is another area of concern, ensure that students not only define methods but also learn how to call these methods.*

MARKING SCHEME

Question 8

```
import java.util.*;
class BinSearch
{
    int arr[];
    int n;
    static Scanner x=new Scanner(System.in);
    BinSearch(int nn)
    {
        n=nn;
    }
    void fillarray()
    {   arr=new int[n];
        System.out.println("Enter "+n + " elements");
        for(int i =0;i<n;i++)
        arr[i]=x.nextInt();
    }
```

```
    void sort()
    {
        int t;
        for(int i=0;i<n-1;i++)
            for(int j =0;j<n-1-i;j++)
                { if (arr[j]>arr[j+1])
                    {  t=arr[j];
                       arr[j]=arr[j+1];
                       arr[j+1]=t;
                     }
                 }
      }
      int bin_search(int l,int u, int v )
     {
           int m=(l+u)/2;
           if(arr[m]==v)
                return m;
           else if(l>u)
                return -1;
           else if (arr[m]>v)
                return bin_search(l,m-1,v);
           else
                return bin_search(m+1,u,v);
      }
      static void main()
     {
            BinSearch obj = new BinSearch(5);
            obj.fillarray();
            obj.sort();
            System.out.println(" location: " + obj.bin_search(0,4,20) );
       }
}
```

Question 9

A class **Mix** has been defined to mix two words, character by character, in the following manner: **[10]**

The first character of the first word is followed by the first character of the second word and so on. If the words are of different length, the remaining characters of the longer word are put at the end.

Example: If the First word is “JUMP” and the second word is “STROLL”, then the required word will be “JSUTMRPOLL”

Some of the members of the class are given below:

Classname	:	**Mix**
Data member/instance variable:		
wrd	:	to store a word
len	:	to store the length of the word
Member functions/methods:		
Mix( )	:	default constructor to initialize the data members with legal initial values
void feedword( )	:	to accept the word in UPPER case
void mix_word(Mix P, Mix Q)	:	mixes the words of objects P and Q as stated above and stores the resultant word in the current object
void display( )	:	displays the word

Specify the class **Mix** giving the details of the **constructor( )**, **void feedword( )**, **void mix_word(Mix, Mix)** and **void display( )**. Define the **main( )** function to create objects and call the functions accordingly to enable the task.

Comments of Examiners

A number of candidates answered this question correctly. The constructor was properly defined in most of the cases. Some candidates accepted the word and converted it into upper case. Passing of objects was not done properly in a few cases. Different methods/logic were used to mix the words in the two objects. Some candidates did not find the length of a word which was required for mixing letters and form a new word. In some cases, object creation was incorrect in the main() method. Knowledge of multiple object creations was not clear in some cases. A few candidates created only one object instead of three in the main( ). In some answers, calling the method mix_word() was incomplete.

Suggestions for Teachers

- *Give more practice to students in passing of objects to a function through parameters. Explain pass by value and pass by reference in detail with examples.*
- *Teach students how to extract characters from words, words from sentences and sentences from paragraphs.*
- *Teach different methods /logic to students so that wider exposure to string manipulation related programs is gained.*
- *Give sufficient practice to students on:*
  - *How to create multiple objects and call the functions with multiple objects.*
  - *Knowledge of constructors to initialize a string and other data members.*
  - *Conversion of a string into characters and concatenating of strings.*

MARKING SCHEME

Question 9

```
import java.util.*;
class Mix
{
        String wrd;
        int len;
        static Scanner x=new Scanner(System.in);
        Mix()
        {
             wrd="";
             len=0;
         }
         void feedword()
         {
               System.out.println( "Enter a word in UPPER CASE");
               wrd=x.next();
               len=wrd.length();
          }
          void mix_word(Mix P,Mix Q)
          {
                int s=(P.len <Q.len)? P.len:Q.len;
                for(int i=0;i<s;i++)
                      wrd += P.wrd.charAt(i)+""+Q.wrd.charAt(i);
                 if (P.len > Q.len)
                        wrd +=P.wrd.substring(Q.len);
                 else
                        wrd +=Q.wrd.substring(P.len);
          }
          void display()
          {
                System.out.println("WORD = " + wrd);
          }
          static void main()
          {
                Mix obj=new Mix();
                Mix obj1=new Mix();
                Mix obj2= new Mix();
                obj.feedword();
                obj1.feedword();
                obj2.mix_word (obj,obj1);
                obj2.display();
          }
    }
```

SECTION – C

*Answer **any two** questions.*

Question 10

A Circular queue is a linear data structure which works on the principle of FIFO, enables the user to enter data from the rear end and remove data from the front end with the rear end connected to the front end to form a circular pattern. Define a class **CirQueue** with the following details:

Class name	:	**CirQueue**
Data member/instance variable:		
cq[ ]	:	array to store the integers
cap	:	stores the maximum capacity of the array
front	:	to point the index of the front end
rear	:	to point the index of the rear end
Member functions/methods:		
CirQueue (int max)	:	constructor to initialize the data member cap=max, front=0 and rear=0
void push(int n)	:	to add integer in the queue from the rear end if possible, otherwise display the message "QUEUE IS FULL"
int pop( )	:	removes and returns the integer from the front end of the queue if any, else returns -9999
void show( )	:	displays the queue elements

(a) Specify the class **CirQueue** giving details of the functions **void push(int)** and **int pop( ).** **[4]**
Assume that the other functions have been defined.

The main function and algorithm need NOT be written.

(b) How is a *linear queue structure* different from a *circular queue structure*? **[1]**

Comments of Examiners	Suggestions for Teachers
(a) The concept of circular queue was not clear to most candidates. Some wrote the code for queue instead of circular queue as asked in the question. The common errors made by the 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	▪ *Give practice in data structure programs like the stacks, queues, circular queues, de-queues, etc. regularly.* ▪ *Emphasise the importance of implementation of stacks,*

properly. Some candidates found the methods push() and pop()to be difficult. A few left the program incomplete. In some cases, all the methods were attempted which was not as asked for.

(b) This part was answered well by most of the candidates. The difference between linear and circular queue was not clear to some candidates and hence gave vague answers. Some candidates illustrated their answers in diagrams. In some cases, the principles of LIFO and FIFO were mentioned as differences.

queues, circular queues, and de queues using arrays.

- *Instruct students to show the workings how the stack or queue performs. (examples can be supportive).*
- *Explain the concept of LIFO and FIFO to the students with examples related to real world.*
- *Students should be made to understand 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. Hence, the array is used to show the working of a stack, queue, circular queue or a de-queue.*
- *Teach students about front and rear pointer index in a queue and top index in a stack with different values. They must understand the difference when the pointer for front = rear = 0 and front = rear = -1 for queues or top with 0 and top with -1 for stacks.*
- *Explain Stacks and Queues (linear, circular and de queue) in detail giving all its applications / uses / differences. Give real world examples to explain the working of these entities. Explain the purpose of % operator in a circular queue to students.*

MARKING SCHEME

Question 10

(a)

```
class CirQueue
 {
       void push(int v)
       {  if( (rear+1)% cap !=front)
            {  rear = (rear + 1)% cap ;
               cq[rear] =v;
             }
          else
               System.out.println(" QUEUE IS FULL");
        }
        int pop()
       {   if (front != rear)
              {  front = ( front + 1)% cap ;
                 return cq[front];
               }
           else
               return -9999;
         }
  }
```

(b) In linear queue structure the front index is always less or equal to rear index, whereas in Circular queue the front index may be greater than rear index.

Question 11

An interface **Data** is defined with a data member and a method volume( ) which returns the volume of the implementing shape. A super class **Base** has been defined to contain the radius of a geometrical shape. Define a sub class **CalVol** which uses the properties of the interface **Data** and the class **Base** and calculates the volume of a cylinder. **[5]**

The details of the members of the interface and both the classes are given below:

Interface name : **Data**

Data member:

double pi	:	initialize pi = 3.142
Member functions/methods:		
double volume( )	:	
Class name:		**Base**
Data member/instance variable:		
rad	:	to store the radius in decimal
Member functions/methods:		
Base(…)	:	parameterized constructor to initialize the data member
void show( )	:	displays the radius with an appropriate message
Class name:		**CalVol**
Data member/instance variable:		
ht	:	to store the height in decimal
Member functions/methods:		
CalVol(…)	:	parameterized constructor to initialize the data members of both the classes
double volume( )	:	calculates the volume of a sphere by using the formula (pi x radius^2 x height)
void show( )	:	displays the data members of both the classes and the volume of the sphere with appropriate message

Assume that the interface ***Data*** *and the super class* ***Base*** *has been defined*. Using the **concept of inheritance**, specify the class **CalVol** giving the details of the **constructor(…)**, **double volume( )** and **void show( )**.

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

Comments of Examiners

Identical mistakes were made by the candidates as in previous years. Concept of Inheritance was not clear to some candidates. The keywords 'extends', 'implements' and 'super' were missing in a few cases. Constructor with inheritance was not answered correctly. Accessing the members of the interface and the super class by the derived class was not clear to some candidates. A few candidates declared the interface and base class also, which was not required. In some cases, an algorithm was written instead of a program.

Suggestions for Teachers

- *Give practice on inheritance to students.*
- *Clarify the importance of keywords 'extends', 'implements' and 'super' in inheritance.*
- *Point out the concept of overriding in inheritance in class.*
- *Explain clearly the use of constructor of the base class.*

- *Explain to students the different visibility modes and their accessing capabilities.*
- *Calling members from the interface and super class to the derived class must be explained clearly. Concept of overriding in inheritance must be explained with examples.*

MARKING SCHEME

Question 11

```
class CalVol extends Base implements Data
{    double ht;
     CalVol(double r, double h)
     {   super(r);
         ht=h;
     }
     public double volume()
    {   double x=pi * rad *rad * ht;
        return x;
     }
     void show()
    {   super.show();
       System.out.println("Height= " + ht);
       System.out.println("Volume= " + volume());
     }
 }
```

Question 12

(a) A linked list is formed from the objects of the class **Node**. The class structure of the Node is given below: **[2]**

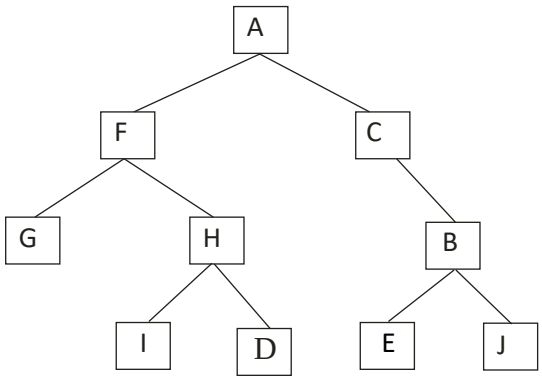
```
class   Node
{
int n;
Node next;
}
```

Write an *Algorithm* **OR** a *Method* to find the product of the integer numbers from an existing linked list.

The method declaration is as follows:

void Product_Node(Node str)

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



(i) Write the *post-order* traversal of the left subtree of the above structure. **[1]**

(ii) State the degree of the Nodes E and H. **[1]**

(iii) Mention the external nodes of the right subtree. **[1]**

Comments of Examiners

(a) This part was well attempted as most candidates scored full marks. However, a few candidates had difficulty in moving the pointer to the next node and checking for null. Some candidates failed to create a temporary pointer. In some cases, the algorithm was written in simple English covering all the main steps. Initializing the accumulator was not correct in some of the cases. In a few answers, the loop was missing.

(b) (i) This subpart was answered correctly by a number of candidates. Some wrote the post-order of the entire tree instead of the left sub tree. In some cases, one or two nodes were incorrectly placed.

(ii) Only a few candidates managed to answer this part correctly. Most of the candidates did not understand the term 'degree' and hence gave vague answers. A few candidates gave only one answer correct.

(iii) This subpart was answered well by most candidates. A few candidates failed to read the question carefully and ended writing the

Suggestions for Teachers

- *Have students practice methods / algorithms with link list and binary tree data structure. Make students use diagrams to illustrate the link list and the Binary Trees.*
- *Give practice on temporary pointer, checking for null condition and moving pointer to the next node.*
- *Teach the algorithms to access elements, edit elements and delete elements of link list with sufficient examples. Explain the difference of arrays and linked list to the students.*
- *With the help of a binary tree diagram explain root, height, depth, size, degree, siblings, nodes (internal and external), levels, path, tree traversals, sub tree, etc.*
- *Instruct students to read the question carefully and answer accordingly.*

external nodes of the entire tree instead of the right sub tree.

MARKING SCHEME

Question 12

(a)

ALGORITHM:

Step 1. Start
Step 2. Set temporary pointer to the first node
Step 3. Repeat steps 4 and 5 until the pointer reaches null. Display product, exit
Step 4. Accumulate the product by accessing each node
Step 5. Move pointer to the next node
Step 6. End algorithm

METHOD:

```
void Product_Node (Node str)
{
    int s=1;
    while (str!=null)
      {  s=s*str.n;
         str=str.next;
       }
    System.out.println("Product="+s);
 }
```

(b)	(i)	G I D H F
	(ii)	Degree of E = 0 and Degree of H = 2
	(iii)	E and J

NOTE: For questions having more than one correct answer/solution, alternate correct answers / solutions, apart from those given in the marking scheme, have also been accepted.

GENERAL COMMENTS

Topics found difficult/confusing by candidates

- Passing objects to functions
- Circular queue operations for push() and pop() operations
- Interfaces and its implementation using inheritance
- Degree of a node in a binary tree
- Properties of zero
- Dual and Compliment of a Boolean expression
- Infix to prefix notation
- Output using recursion
- Circular queue
- Concept of inheritance with interface and super class
- Degree of a node in a Binary tree
- Properties of zero in Boolean algebra
- Conversion from infix notation to prefix notation
- Best case and worst case for bubble sort
- Output using recursive function
- K-MAPS (grouping, redundant groups, map-rolling, place value)
- Drawing XOR gate using NOR gate
- Recursive technique

Suggestions for Students

- Apart from Textbook learning, surfing the net (Wikipedia, School net, etc.) and other reliable resources would be helpful to gather knowledge on the latest concepts and enhancements in programming. Proper definitions, output programs, algorithms, etc. can be downloaded directly from the internet.
- Prepare a summary for each chapter or use high lighters to highlight the important terms and definitions.
- Revise of the entire syllabus thoroughly.
- Relate everything taught in class with Real Life Environment examples which helps in remembering / recall of the concepts learnt.
- Practical work on the computer is necessary on regular basis to understand the syntax and to correct errors.
- Answers and definitions should be short and precise and according to the marks allocated.
- Solve ISC previous years' question papers.
- Underline/ Highlight Important words and terms.
- Show working at the side of each question, wherever required.
- Mention laws 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 memorise the program, try to understand the logic.
- Practice constructors with every program. Treat each function of a class as a separate program.
- Do your preparations section-wise following the scope of the syllabus.