ISC YEAR 2026

COMPUTER SCIENCE - PAPER 2

PRACTICAL EXAMINATION

Instructions and Guidelines for Visiting Examiners

I. PREPARATION OF QUESTION PAPER/S AND CONDUCT OF PRACTICAL EXAMINATION

The Practical Examination must be conducted between 1st October 2025 and 31st January 2026.

Steps to be followed for conducting the Computer Science Practical Examination.

Deciding the number of Batches and Dates for the Practical Examination/s

1. The Visiting Examiner must collect the soft copies/hard copies of all documents related to the conduct of the Practical Examination from the Head of the School.
2. The Head of the School, in consultation with the Visiting Examiner and the Chief Supervising Examiner should decide the number of batches in which the candidates of the school would take the practical examination. This should be done keeping in mind the size of the laboratory and total number of candidates appearing for Computer Science practical examination.
3. Each batch should be assigned a batch number – e.g.,

   Batch 1 (Index no. 226001/001 to 226001/010);

   Batch 2 (Index no. 226001/011 to 226001/020).

   A record of this information should be maintained by the Head of the School. A copy of this record should be submitted to the Convener.
4. Dates of Practical Examination for each batch must be decided according to the convenience of both the Visiting Examiner and the School.
5. The candidates must be informed of the dates of the Practical Examination (for their batch) ***at least one week in advance***.

Finalising the Question Papers/s

1. The Visiting Examiner will be required to set the question paper/s for the candidate(s) concerned based on the syllabus for the ISC Year 2026 Practical Examination and other guidelines provided by CISCE. The question paper/s prepared could be similar to the ones in the Sample Question Papers provided by CISCE in this document.
2. The Visiting Examiner should select the questions, as per the Blueprint and other guidelines provided by CISCE, include relevant details and run the programs where required, to ensure that the desired results are obtained before finalising the Question Paper.
3. The Visiting Examiner should use a range of questions (as provided in the sample questions) for each batch, while ensuring that all batches get Question Papers of similar difficulty level.
4. After finalising the Question Paper/s, the Visiting Examiner must inform the Head of the School, regarding the arrangements to be made in the Computer Science laboratory for the conduct of the practical examination.

Preparation and Storage of Question Paper/s prepared

5. The Head of the School, in presence of the Visiting Examiner and the Chief Supervising Examiner, should ensure that the question papers are photocopied in adequate numbers (as per the number of candidates concerned), well in advance.

6. The photocopied question paper/s should be packed and sealed in envelope(s) by the Visiting Examiner which should be labelled as follows:

Subject: <u>Computer Science-Paper 2 Practical Examination</u>

Date of Examination: ______________________

Batch Number: ___________________________

Index No.(s):____________________

Number of Question Papers packed:____________

Both the Chief Supervising Examiner, and the Visiting Examiner should put their signatures across the seal of the envelope(s), with the date.

7. The Visiting Examiner should handover the sealed envelope(s) containing copies of the finalised question paper/s to the Chief Supervising Examiner for safe custody.

8. The Chief Supervising Examiner must ensure that the sealed envelope(s) of the question papers are kept securely, under lock and key, in a cupboard, in the office of the Head of the School.

9. All concerned must maintain strict confidentiality of the finalised question papers.

Uploading of Marks awarded by the Visiting Examiner.

10. The marks awarded by the Visiting Examiner, as per the Guidelines for Evaluation of Computer Science Paper 2 (Practical) (given on page 6), must be uploaded onto the CAREERS Portal after the evaluation of answer script(s) of candidate(s)concerned.

11. The Visiting Examiner must also evaluate and upload the marks awarded for the Project work and Practical files of the candidates.

12. After uploading the marks, the data must be saved on the Portal by clicking the 'SAVE' button. The final submission of marks to CISCE should then be made after the marks of all the candidates of the school, for the given subject, have been uploaded onto the Portal.

13. It is necessary to ensure that the marks of all candidates are entered correctly, before submitting the same to CISCE.

14. After the final submission of marks to CISCE, the Visiting Examiner must take a print-out of the marks submitted, and put his name, signature and date on each sheet. This print-out of marks must be handed over in a sealed envelope to the Head of the School.

15. The marks awarded to the candidates must be treated as strictly confidential and must not be disclosed to anyone by the Visiting Examiner.

16. The evaluated answer scripts must be packed securely in durable envelope(s). The Chief Supervising Examiner must sign the packed answer script envelopes to ensure that the answer scripts were packed in her/his presence and that she/he has verified the number of scripts enclosed in the envelope.

    All answer script envelopes must be addressed to:

    The Chief Executive & Secretary
    Council for the Indian School Certificate Examinations
    Plot No. 35-36, Sector 6,
    Pushp Vihar, Saket
    New Delhi – 110017

II. BLUEPRINT OF THE PRACTICAL QUESTION PAPER

The Computer Science Practical Question Paper comprises THREE questions and the candidate must select any ONE question. Each question must consist of four components. The details /marks distribution is given below:

1.	Algorithm	**3 marks**
2.	Program in Java with documentation	**7 marks**
3.	Hard copy / printout	**2 marks**
4.	Output / execution	**3 marks**
	Total	**15 Marks**

The three programs must be based on the topics as given below:

Question 1.	Number Logic / Date concept / Time concept/ Number encryption
Question2.	Single dimensional array / Double dimensional array
Question 3.	String manipulation / String encryption

In addition to the above, the Visiting Examiner will also be required to evaluate the following for each candidate:

• Programming assignments done throughout the year (assessed by the Subject teacher)	**10 marks**
• Programming assignments done throughout the year (assessed by the Visiting Examiner)	**5 marks**
Total	**15 Marks**

III. QUESTION SPECIFIC GUIDELINES & INSTRUCTIONS

The Visiting Examiner, in consultation with the subject teacher, must decide on the programs to set the question paper. The Sample Question Papers provided by CISCE may be referred to for this purpose.

The questions must be set as follows:

Question 1: Number Logic / Date concept / Time Concept

The program should check the validity of a given input / range and output the desired result in a formatted manner. Logic for checking the validity, extraction of digits from a number, changing a Numeric literal to String literal and vice-versa may be tested.

- Number logic- to extract digits from a number, checking and generating a particular type of number (i.e. Smith Number, Kaprekar Number etc.).
- Date concept- to check validity of dates, to find the number of days between two given dates in the same calendar year, future date etc.
- Time concept- to find the time elapsed between any two-given time in a particular day, to find the future time after some duration (in hours and minutes)
- Number encryption – to convert decimal numbers to different bases (Binary, Octal and Hexadecimal) and vice versa. Coding of numbers using their ASCII values

Question 2: Single dimensional array/Double dimensional array

The program should check the validity of the size of the matrix/array and output the desired result in a formatted manner. Formula to check if the elements lie on the boundary, non-boundary, diagonal etc. in a given matrix/array.

- Single dimensional array- merging of two arrays, to arrange the array elements in a particular manner etc.
- Double dimensional array / matrix- to find the sum/sort the elements of each row/column, boundary elements, non-boundary elements, saddle point etc.
- Square matrix to find sum/sort elements of the diagonal, maximum and minimum value with location in the matrix, mirror/inversion of the matrix etc.

Question 3: String manipulation / String encryption

The program to check the validity of a sentence/paragraph with the terminating character given. Logic to extract the words from a sentence, characters from words, vowels, and consonants etc.

- Paragraph input with a maximum of two sentences and finding the number of words in each sentence.
- To separate the words beginning with a vowel in a sentence and find its frequency.
- To change the Upper-case characters to lower case and vice versa in each sentence along with the count of palindrome words present in the sentence, if any. Encoding and decoding of characters, sorting of characters in a string etc.

IV.GUIDELINES FOR EVALUATION OF COMPUTER SCIENCE PAPER–2 (PRACTICAL)

(for the use of the Visiting Examiners)

Marks (Out of a total of 30) for this paper are to be distributed as follows:		
A.	Assessment of Practical Examination:	**15 Marks**
B.	Assessment of programming assignments done throughout the year:	**15 Marks**
<u>Part A: Assessment of Practical Examination (15 Marks)</u>		
Candidates are expected to plan their programs and test run them on the computer. The various stages are to be assessed as follows:		
1.	**Algorithm:**	**[3]**
	❖ Choice of algorithm and implementation strategy.	[1½]
	❖ Complete and clearly expressed algorithm using any standard schemes (i.e. pseudo code or stepwise) and according to the program.	[1½]
2. (a)	**Java Program: (Selected by candidate)**	**[5]**
	❖ Knowledge of Input requirement and the choice of data type.	[1]
	❖ Clear description of classes, functional behavior, functional argument and return type and any formulae or special logic/method used.	[1]
	❖ The program follows the algorithm correctly and is logically correct.	[2]
	❖ Use of correct and formatted output statement.	[1]
(b)	**Documentation** to be provided in the handwritten or printed coded program by comments/mnemonic names.	**[2]**
3.	**The final listing (hard copy)** that the candidate submits at the end, follows the algorithm and is logically correct. Large differences between planned program and the printout will result in loss of marks.	**[2]**
4.	**Execution and testing**:	**[3]**
	❖ The program runs successfully on the sample inputs to produce the correct sample outputs (correct output on known inputs).	[2]
	❖ The examiner should ask the candidate to run program on other inputs to ensure that it is correct. These would typically be the extreme boundary conditions (correct output on unknown inputs).	[1]
	❖ Candidates are required to execute programs using JDK environment/BlueJ/Eclipse/NetBeans. *(BlueJ allows the user to write a class, create objects and test their functionality without having to write public static void main (). Hence, the program executed using BlueJ environment should also be accepted.)*	

Part B: Assessment of Programming assignments done throughout the year (15 Marks)	
❖	The Subject Teacher will assess the year’s work of candidates (assignments done as practical work throughout the year) and award marks out of 10.
❖	These marks, along with the year’s work, should be made available to the Visiting Examiner.
❖	After taking the internally awarded marks into consideration, the Visiting Examiner should award marks out of 5, for the year’s work of candidates.
❖	The total marks for continuous evaluation, therefore, shall be out of 15 marks (out of 10 marks awarded through internal evaluation and out of 5 marks awarded by the Visiting Examiner).

ISC YEAR 2026

COMPUTER SCIENCE PAPER 2

PRACTICAL EXAMINATION

SAMPLE PAPERS

PREPARING THE QUESTION PAPER

- The Sample Question Papers provided by CISCE include different questions for each type of Practical Work given in the syllabus. These questions may be used by the Visiting Examiner for setting the question Paper/s.
- The Visiting Examiners are advised to go through the ***Instructions and Guidelines for Visiting Examiners***, before setting the Question Paper/s.
- The questions must be selected as per the Blueprint and other details provided by CISCE, include relevant details and run the program himself/herself, where required, to ensure that the desired results are obtained before finalizing the Question Paper.
- The text for the questions may be similar to the text used for the Sample Questions.
- The provided Top Sheet, detailing the time provided for reading the Question Paper, the duration of the Examination and other necessary information, must be attached as the first sheet of the Question Paper.

COMPUTER SCIENCE

PAPER – 2

PRACTICALS

(Maximum Marks: 30)

Time allowed: Three Hours

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

The total time to be spent on the Planning Session and the Examination Session is Three hours.
After completing the Planning Session, the candidate may begin the Examination Session.
A maximum of 90 minutes is permitted for the Planning Session.
However, if candidates finish earlier, they are permitted to begin the Examination Session.

This paper consists of ***three*** *problems from which candidates are required to attempt* ***any one*** *problem.*

Candidates are expected to do the following:

A. Planning Session:

1. Write an algorithm for the selected problem. **[3marks]**

   (Algorithm should be expressed clearly using any standard scheme such as pseudo code or in steps which are simple enough to be obviously computable.)

2. Write a program in **JAVA** language. The program should follow the algorithm and should be logically and syntactically correct. Document the program using mnemonic names / comments, identifying and clearly describing the choice of data types and meaning of variables. **[7marks]**

B. Examination Session:

1. Code / Type the program on the computer and get a printout (hard copy). Typically, this should be a program that compiles and runs correctly. **[2marks]**

2. Test run the program on the computer using the given sample data and get a printout of the output in the format specified in the problem. **[3marks]**

Note: *The candidates must not carry any stationery, items such as pen / pencil / eraser to the Computer Laboratory for the Examination Session.*

Top Sheet: To be attached as the first page of the Question Paper.

SAMPLE PAPER 1

Solve **any one** of the following Problems.

Question 1

A **Digit Permutation Cipher** is a simple form of number encryption where the digits of a number are rearranged based on a given key where (1<= key <=size of the number). The key is a sequence of integers that defines the new positions of the digits.

Example: If number = 2613 and key = 4213, then the encrypted number will be 1632 by positioning the first digit to the 4th place, second digit to the second place, third digit to the first place and the fourth digit to the third place as per the key given.

Write a program to enter a number and a permutation key (a sequence of digits which is greater than 0 and less than or equal to the size of the number). The program should encrypt the number by permuting its digits according to the key. The number of digits in the key must match the number of digits in the number to be encrypted.

Test your program with the following data and some random data:

Example 1
INPUT: Number: 12345
Key: 31524
OUTPUT: The encrypted number is 24153

Example 2
INPUT: Number: 9876
Key: 4132
OUTPUT: The encrypted number is 8679

Example 3
INPUT: Number: 5239
Key: 4765
OUTPUT: INVALID KEY DIGITS

Example 4
INPUT: Number: 123
Key: 2134
OUTPUT: INVALID KEY SIZE

Question 2

Write a program to declare a single-dimensional array A [ ] of size L, where L is an integer greater than or equal to 3 and less than or equal to 50. Allow the user to input integers into this array. Display an appropriate error message for an invalid input.

Perform the following tasks on the array:

(a) Display the array.

(b) Find and print all equilibrium indices in increasing order. An index i (0-based) is called an equilibrium index if the sum of elements on its left equals the sum of elements on its right.

(c) If no equilibrium index exists, display an appropriate message.

Test your program with the following data and some random data:

Example 1

```
INPUT:     L = 7
           Array Elements:   2, 3, -1, 8, 4, -2, 2
OUTPUT:    Array:      2, 3, -1, 8, 4, -2, 2
           Equilibrium Indices: 3
```

Example 2

```
INPUT:     L = 5
           Array Elements:   0, -7, 3, -2, 6
OUTPUT:    Array:      0, -7, 3, -2, 6
           Equilibrium Indices: 0
```

Example 3

```
INPUT:     L = 6
           Array Elements:   1, 2, 3, 4, 5, 6
OUTPUT:    Array:      1, 2, 3, 4, 5, 6
           Equilibrium Indices: NIL
```

Example 4

```
INPUT:     L = 7
           Array Elements:   -7, 1, 5, 2, -4, 3, 0
OUTPUT:    Array:      -7, 1, 5, 2, -4, 3, 0
           Equilibrium Indices: 3, 6
```

Example 5

```
INPUT:     L = 2

OUTPUT:    INVALID INPUT
```

Question 3

Write a program to accept a sentence which may be terminated by either '.', '?'or '!' only. The words may be separated by a single blank space and should be case-insensitive.

Perform the following tasks:

(a) If the sentence is a pangram (a sentence or a phrase that uses every letter of the alphabet at least once), then print PANGRAM.

(b) If it misses exactly one letter of the alphabet, print PANGRAMMATIC LIPOGRAM and the missing letter.

(c) Else print NEITHER and list all missing letters in alphabetical order.

Test your program for the following data and some random data:

Example 1

INPUT: The quick brown fox jumps over the lazy dog!

OUTPUT:
```
IT IS A PANGRAM
```

Example 2

INPUT: A quick movement of the enemy will jeopardize.

OUTPUT:
```
PANGRAMMATIC LIPOGRAM
MISSING:  x
```

Example 3

INPUT: Hello world.

OUTPUT:
```
NEITHER

MISSING:  a b c f g i j k m n p q s t u v x y z
```

Example 4

INPUT: Alas! it failed #

OUTPUT:
```
INVALID INPUT
```

SAMPLE PAPER 2

Solve **any one** of the following Problems.

Question 1

A distinct-prime-digit integer is a positive integer (without leading zeros) in which all digits are prime numbers and no digit is repeated. The prime digits are **2, 3, 5, 7**. For example, 2, 37, 253 are distinct-prime-digit integers, whereas 33, 252, 29 are not.

Given two positive integers **m** and **n**, where **m < n**, write a program to determine how many distinct-prime-digit integers are there in the range between **m** and **n** (both inclusive) and output them.

Test your program for the following data and some random data:

Example 1

INPUT:
```
m = 20
n = 60
```

OUTPUT:
```
THE DISTINCT-PRIME-DIGIT INTEGERS ARE:
23, 25, 27, 32, 35, 37, 52, 53, 57
FREQUENCY OF DISTINCT-PRIME-DIGIT INTEGERS IS: 9
```

Example 2

INPUT:
```
m = 70
n = 120
```

OUTPUT:
```
THE DISTINCT-PRIME-DIGIT INTEGERS ARE:
72, 73, 75, 77
FREQUENCY OF DISTINCT-PRIME-DIGIT INTEGERS IS: 4
```

Example 3

INPUT:
```
m = 100
n = 180
```

OUTPUT:
```
THE DISTINCT-PRIME-DIGIT INTEGERS ARE:  NIL
FREQUENCY OF DISTINCT-PRIME-DIGIT INTEGERS IS: 0
```

Example 4

INPUT:
```
m = 200
n = 150
```

OUTPUT:
```
INVALID INPUT
```

Question 2

Write a program to declare a single-dimensional array arr[ ] of size N, where N is an integer greater than 2 and less than 10. Allow the user to input positive integers into this array. Display an appropriate error message for an invalid input.

Rearrange the array in zig-zag order in the form arr0 ≤ arr1 ≥ arr2 ≤ arr3 ≥ arr4 …. using **in-place** swaps only.

Example: Input: [4, 3, 7, 8, 6, 2, 1]
Output: [3, 7, 4, 8, 2, 6, 1] after rearranging in zig-zag order.

Perform the following tasks on the array:

(a) Display original array.
(b) Transform to zig-zag order.
(c) Display transformed array.

Test your program with the following data and some random data:

Example 1

INPUT: N=5
Array Elements: 20, 5, 25, 30, 18

OUTPUT: Original Array: 20, 5, 25, 30, 18
Zig-zag Array: 5, 25, 20, 30, 18

Example 2

INPUT: N=8
Array Elements: 100, 50, 175, 25, 12, 150, 17, 200

OUTPUT: Original Array: 100, 50, 175, 25, 12, 150, 17, 200
Zig-zag Array: 50, 175, 100, 150, 12, 25, 17, 200

Example 3

INPUT: N=6
Array Elements: 21, 4, 34, –2, 9 10

OUTPUT: **INVALID INPUT**

Example 4

INPUT: N=4

OUTPUT: **INVALID INPUT**

Question 3

Write a program to accept a sentence which may be terminated by either **'.'**, **'?'**or **'!'** only. The words may be separated by a single blank space and should be case-insensitive.

Perform the following tasks:

(a) Check if the sentence is a Palindrome Sentence.
[A sentence is a Palindrome Sentence if, after removing spaces and punctuation, the letters read the same forward and backward.]
Example: " Never odd or even "

(b) Display the first-occurring most frequent word in the sentence (in lower-case). If there is a tie, choose the word that appears first in the sentence and if no words are repeated then print NONE.

Test your program for the following data and some random data:

Example 1

INPUT: No lemon no melon.

OUTPUT: IT IS A PALINDROME SENTENCE
MOST FREQUENT WORD: no

Example 2

INPUT: Was it a car or a cat I saw?

OUTPUT: IT IS A PALINDROME SENTENCE
MOST FREQUENT WORD: a

Example 3

INPUT: It is a rainy day!

OUTPUT: IT IS NOT A PALINDROME SENTENCE
MOST FREQUENT WORD: NONE

Example 4

INPUT: Always be careful#

OUTPUT: INVALID INPUT

SAMPLE PAPER 3

Solve **any one** of the following Problems.

Question 1

The **12-hour clock format** (A.M./P.M.) is a time convention in which the 24 hours of the day are divided into two periods. The **24-hour clock format** is the international standard.

Example: If time = 9:35 PM, then output will be 21:35

Write a program to accept a time in 12-hour clock format (HH:MM AM/PM) and convert it to 24-hour clock format (HH:MM).

Test your program for the following data and some random data:

Example 1

INPUT: 11:30 PM

OUTPUT: 23:30

Example 2

INPUT: 12:00 AM

OUTPUT: 00:00

Example 3

INPUT: 1:45 AM

OUTPUT: 01:45

Example 4

INPUT: 12:00 PM

OUTPUT: 12:00

Example 5

INPUT: 13:30 AM

OUTPUT: INVALID INPUT

Question 2

Write a program to declare a matrix A[ ][ ] of order (M × N) where ‘M’ is the number of rows and ‘N’ is the number of columns such that both M and N must be greater than 2 and less than 10. Allow the user to input positive integers into this matrix. Display appropriate error message for an invalid input.

Perform the following tasks on the matrix:

(a) Display the input matrix
(b) Compute the **sum of prime elements** for each row and column
(c) Identify the **row index** with maximum prime-sum and the **column index** with maximum prime-sum.
(d) Print both indices and sums; if no primes found in all, print a message.

Example: Input: M=3 & N=4 and array is:

```
4    5   6    7
8   11  10   13
1    2   3    4
```

then, Output: Row with max prime-sum: 1 (sum=24)
Column with max prime-sum: 3 (sum=20)

Test your program for the following data and some random data:

Example 1

INPUT: M = 4
N = 4
Array elements: 11,3,4,7,13,9,5,17,2,3,19,23,12,6,13,4

OUTPUT: Original array:

```
11  3   4   7
13  9   5  17
 2  3  19  23
12  6  13   4
```

Row with max prime-sum: 2 (sum=47)
Column with max prime-sum: 3 (sum=47)

Example 2

INPUT: M = 3
N = 3
Array elements: 2, 6, 4, 11, 5, 9, 12, 7, 13

OUTPUT: Original array:

```
 2   6   4
11   5   9
12   7  13
```

Row with max prime-sum: 2 (sum=20)
Column with max prime-sum: 0 and 3 (sum=13)

Example 3

INPUT: M = 3
N = 3
Array elements: 12, 11, –5, 9, 1, 7, 13, 2, 3

OUTPUT: INVALID IN PUT

Example 4

INPUT: M = 2
N = 3

OUTPUT: INVALID INPUT

Question 3

Accept a sentence terminated by **'.', '?' or '!' only**. Sentence may contain any of ()[]{} plus letters/spaces. It must contain no digits and no other symbols besides the final terminator. For any violation, display: INVALID INPUT.

Perform the following tasks:

(a) Check if the brackets are **balanced and properly nested**.
(b) Print the maximum nesting depth of brackets.
(c) If unbalanced, print the index (1-based indexing) of the first error.
- A mismatched/extra closing bracket is an error at that bracket's index.
- If brackets are incomplete at the terminator, the first error is the index of the first bracket that cannot be matched (typically where the mismatch first occurs).

Test your program for the following data and some random data:

Example 1

INPUT: This (is [very {deep}]) indeed!

OUTPUT: BALANCED: YES
MAX DEPTH: 3

Example 2

INPUT: Simple () test.

OUTPUT: BALANCED: YES
MAX DEPTH: 1

Example 3

INPUT: Hello world.

OUTPUT: BALANCED: YES
MAX DEPTH: 0

Example 4

INPUT: We love (balanced [brackets)!

OUTPUT: BALANCED: NO
ERROR AT INDEX: 22

Example 5

INPUT: Hi #there!

OUTPUT: INVALID INPUT