COMPUTER SCIENCE

Paper – 2

(PRACTICAL)

(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.

Planning session: 90 minutes

Examination session: 90 minutes

Note: Candidates are to be permitted to proceed to the Examination Session only after 90 minutes of the Planning session are over.

---

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

Candidates are expected to do the following:

1. Write an algorithm for the selected problem. **[3]**
   (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. **[5]**
3. Document the program using mnemonic names / comments, identifying and clearly describing the choice of data types and meaning of variables. **[2]**
4. Code / Type the program on the computer and get a printout (hard copy). Typically, this should be a program that compiles and runs correctly. **[2]**
5. 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. **[5]**
6. Viva-Voce on the **Selected Problem**. **[3]**

In addition to the above, the practical file of the candidate containing the practical work related to programming assignments done during the year is to be evaluated as follows:

- Programming assignments done throughout the year (by the teacher) **[5]**
- Programming assignments done throughout the year (by the Visiting Examiner) **[5]**

This Paper consists of 5 printed pages and 1 blank page.

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

Question 1

Design a program to accept a day number (between 1 and 366), year (in 4 digits) from the user to generate and display the corresponding date. Also, accept 'N' (1 <= N <= 100) from the user to compute and display the future date corresponding to 'N' days after the generated date. Display an error message if the value of the day number, year and N are not within the limit or not according to the condition specified.

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

Example 1

INPUT: DAY NUMBER: 255
YEAR: 2018
DATE AFTER (N DAYS): 22

OUTPUT: DATE: 12 TH SEPTEMBER, 2018
DATE AFTER 22 DAYS: 4 TH OCTOBER, 2018

Example 2

INPUT: DAY NUMBER: 360
YEAR: 2018
DATE AFTER (N DAYS): 45

OUTPUT: DATE: 26 TH DECEMBER, 2018
DATE AFTER 45 DAYS: 9 TH FEBRUARY, 2019

Example 3

INPUT: DAY NUMBER: 500
YEAR: 2018
DATE AFTER (N DAYS): 33

OUTPUT: DAY NUMBER OUT OF RANGE.

Example 4

INPUT: DAY NUMBER: 150
YEAR: 2018
DATE AFTER (N DAYS): 330

OUTPUT: DATE AFTER (N DAYS) OUT OF RANGE.

Question 2

Write a program to declare a single dimensional array a[ ] and a square matrix b[ ] [ ] of size N, where N>2 and N<10. Allow the user to input positive integers into the single dimensional array.

Perform the following tasks on the matrix:

(a) Sort the elements of the single dimensional array in ascending order using any standard sorting technique and display the sorted elements.

(b) Fill the square matrix b[ ][ ] in the following format.

If the array a[ ] = { 5, 2, 8, 1 } then, after sorting a[ ] = {1, 2, 5, 8}

Then, the matrix b[ ][ ] would fill as below:

1	2	5	8
1	2	5	1
1	2	1	2
1	1	2	5

(c) Display the filled matrix in the above format.

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

Example 1

INPUT: N = 3
ENTER ELEMENTS OF SINGLE DIMENSIONAL ARRAY: 3 1 7

OUTPUT: SORTED ARRAY: 1 3 7
FILLED MATRIX

1	3	7
1	3	1
1	1	3

Example 2

INPUT: N = 13

OUTPUT: MATRIX SIZE OUT OF RANGE

Example 3

INPUT: N = 5
ENTER ELEMENTS OF SINGLE DIMENSIONAL ARRAY: 10 2 5 23 6

OUTPUT: SORTED ARRAY: 2 5 6 10 23
FILLED MATRIX

2	5	6	10	23
2	5	6	10	2
2	5	6	2	5
2	5	2	5	6
2	2	5	6	10

Question 3

Write a program to accept a sentence which may be terminated by either '.' , '?' or '!' only. The words are to be separated by a single blank space and are in UPPER CASE.

Perform the following tasks:

(a) Check for the validity of the accepted sentence.

(b) Convert the non-palindrome words of the sentence into palindrome words by concatenating the word by its reverse (excluding the last character).

Example: The reverse of the word HELP would be LEH (omitting the last alphabet) and by concatenating both, the new palindrome word is HELPLEH. Thus, the word HELP becomes HELPLEH.

Note: The words which end with repeated alphabets, for example ABB would become ABBA and not ABBBA and XAZZZ becomes XAZZZAX.

[Palindrome word: Spells same from either side. Example: DAD, MADAM etc.]

(c) Display the original sentence along with the converted sentence.

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

Example 1

INPUT: THE BIRD IS FLYING.

OUTPUT: THE BIRD IS FLYING.
THEHT BIRDRIB ISI FLYINGNIYLF

Example 2

INPUT: IS THE WATER LEVEL RISING?

OUTPUT: IS THE WATER LEVEL RISING?
ISI THEHT WATERETAW LEVEL RISINGNISIR

Example 3

INPUT: THIS MOBILE APP LOOKS FINE.

OUTPUT: THIS MOBILE APP LOOKS FINE.
THISIHT MOBILELIBOM APPA LOOKSKOOL FINENIF

Example 4

INPUT: YOU MUST BE CRAZY#

OUTPUT: INVALID INPUT