CSE 295

L-2/T-1/IPE

Date: 13/09/2025

BANGLADESH UNIVERSITY OF ENGINEERING AND TECHNOLOGY, DHAKA

L-2/T-1 B. Sc. Engineering Examinations 2023-2024

Sub: **CSE 295** (Computer Programming Techniques)

Full Marks: 210 Time: 3 Hours

The figures in the margin indicate full marks.

USE SEPARATE SCRIPTS FOR EACH SECTION

SECTION – A

There are **FOUR** questions in this section. Answer **question No. 1** and any **TWO** from the rest.

1. (a) Why is learning programming essential in IPE (industrial Production Engineering)? Give two practical examples. **(10)**

(b) What are the primary and the user-defined data types in C? Explain with relevant examples. **(10)**

(c) Write a code segment in C to calculate the sum of n terms of the series: **(10)**

$$1 + 3/2! + 5^2/3! + 7^3/4! + \ldots..$$

(d) What will be the output if you execute the following C code? Explain how. **(05)**

```
#include <stdio.h>

int main()
{
    int end = 5;
    int i = 1;

    while (i <= end) {
        printf("\n");
        int j = 1;
        while (j <= i) {
            printf("%d ", j);
            j = j + 1;
        }
        i = i + 1;
    }
    return 0;
}
```

2. (a) Convert this switch-case into an equivalent expression with ternary operations: **(05)**

```
switch (x) {
  case 1: printf("One"); break;
  case 2: printf("Two"); break;
  default: printf("Invalid");
}
```

(b) What is the difference between while and do-while loops? When would you prefer one over the other? **(10)**

Contd P/2

CSE 295/ IPE

(Contd ... Q. No. 2)

(c) Given a number (num) write a C program to check whether it is a Neon Number (i.e. a number where the sum of digits of the square of the number is equal to the number) and return "true" or "false" accordingly. **(10)**

Example:

Input: num = 9

Output: true

Explanation: square of 9 is 9 * 9 = 81, sum of digit of square is 8 + 1 = 9 (i. e equal to given number).

(d) Given a sorted array arr and an integer k, write a function to find the position (0-based indexing) at which k is present in the array using binary search. If multiple occurrences are there, return the smallest index. **(10)**

Example:

Input: arr[] = [1, 2, 3, 4, 5], k = 4

Output: 3

3. (a) What is the difference between function arguments and parameters? What are the actual and formal parameters? **(10)**

(b) Write a C program to find the Sum to natural numbers up to N using Recursion. N will be given as input. **(10)**

(c) Given two unsorted input arrays write a C program to combine the elements of both arrays into a single sorted array. **(15)**

4. (a) What is mutual recursion? Explain with examples. **(10)**

(b) Given two arrays, write a C Program to find common array elements between those two arrays. **(10)**

Input:

array1 [] = {8, 2, 3, 4, 5, 6, 7, 1}

array2 [] = {4, 5, 7, 11, 6, 1}

Output:

Common elements are: 4 5 6 7 1

(c) Given a string Str, write a C program to check if Str is a Pangram or not. A pangram is a sentence containing every letter in the English Alphabet. **(15)**

Examples:

Input: "The quick brown fox jumps over the lazy dog"

Output: is a Pangram

Explanation: Contains all the characters from 'a' t 'z'

Contd P/3

55

CSE 295/ IPE

SECTION – B

There are FOUR questions in this section. Answer question No. 5 and any TWO from the rest.

5. (a) In many engineering fields, data is often stored and processed in matrices. A square matrix is symmetric if it looks the same when you flip it along its main diagonal (from top-left to bottom-right). This means the element at row i, column j is always the same as the element at row j, column i ($A_{ij} = A_{ji}$). Because of this property, symmetric matrices contain redundant information, which gives us an opportunity for efficient storage by only saving the unique elements. **(11+10)**

With this in mind, you are required to answer the following questions. You **MUST NOT** rely on global variables for solving any of them.

i. Design a function `compressSymmetric` that takes as parameters a symmetric square matrix `mat` and an integer `n` representing its size. Using the symmetry property, the function should store only the minimum number of elements necessary to allow complete reconstruction of the original matrix. It should return a dynamically allocated array containing these essential elements. Choose an appropriate return type based on your method. You may introduce additional parameters if needed.

ii. Demonstrate the functionality of your solution in i. by writing a function `demo` that takes as parameters a symmetric square matrix `mat` and an integer `n` representing its size. The function should compress the matrix using `compressSymmetric`, and then reconstruct the original matrix directly from the array returned by `compressSymmetric` function. Finally, the `demo` function must properly free all dynamically allocated memory used during the compression and reconstruction steps. You may introduce any additional helper functions or parameters as needed.

(b) Analyze the program in listing 1. Identify and briefly explain all errors/bugs in the program. **(14)**

6. (a) Consider the C program shown in listing 2. Assume that arrays are stored in memory as follows: `a[0]` starts at address 1000, `b[0]` at 2000, `c[0]` at 3000, and `parr[0]` at 4000. Each integer and integer pointer occupy 4 bytes. **(10+16=26)**

i. Draw a schematic memory layout showing how the arrays `a`, `b`, `c`, and `parr` are stored in memory. You do not need to show each individual byte; instead, consider each cell to be 4 bytes wide (i.e., one int). Label each cell with the address of its first byte and include the contents stored in that cell. For pointer-type variables (such as elements of parr), use arrows to indicate the cells they point to.

Contd P/4

CSE 295/ IPE (Contd ... Q. No. 6)

ii. Evaluate values of the following expressions

(a) `*parr[0]`

(b) `(parr - 2)`

(c) `*(*(parr + 2))`

(d) `*(parr + 3)`

(e) `*parr[3]`

(f) `*(parr[3] - 1)`

(g) `*(*(parr + 1) + 2)`

(h) `*(parr + 2)[1]`

(b) Briefly explain the purpose of `#include <stdio.h>` in a C program and how the C preprocessor processes this line. (9)

7. You are required to design a program to manage students' performance in a course. Each student has a student ID, name, marks in four class tests (CTs), an attendance percentage, and marks in the term final exam. The term final is divided into two sections: Section A and Section B, each out of 105 marks. Students are categorized into three groups based on their registration and attendance: regular, incomplete, and withdrawn. The course can have at most 120 students.

(a) Design appropriate structs, unions, and enums to fully capture and store all necessary details of each student in the course. The defined structures should include one named `Student` that stores all necessary information for a student. (10)

(b) Write a function `ctMarks` that takes a pointer to a `Student` structure and returns the sum of the best 3 out of 4 CT marks. (8)

(c) Write a function `computeFinalScore` that takes a pointer to a `Student` structure and returns the total marks in percentage. The final score should be computed as follows: 20% from CT marks (sum of best 3), 10% from attendance, and 70% from term final marks. (5)

(d) Write a function `statistics` that takes an array of `Student` and the size of the array, and prints the percentage of regular students, average attendance, average of each CT, average marks in Section A and Section B of the term final, and average final score percentage. Only regular students should be considered in the average calculations. (12)

8. (a) Write a C program that reads student records from a file named `students.txt`, where each line contains a student's full name, student ID, and email address separated by commas. A record is considered valid only if both of the following conditions hold: (28)

- The email address ends with `@ipe.buet.ac.bd` (case-insensitive).
- The student ID is exactly 9 digits long, where:
  - The first four digits represent a year starting with `20` (e.g., `2021`, `2023`).
  - The next two digits are `08`.
  - The last three digits form a number less than 121.

Contd P/5

57

CSE 295/ IPE

(Contd ... Q. No. 8)

Your program should write all valid records to `valid.txt`, write all invalid records to `invalid.txt` appending the reason(s) for invalidity (e.g., "Invalid email", "Invalid ID", or both), and generate a `summary.txt` file containing:

- Total students processed.
- Number of valid and invalid records.
- Percentage of valid and invalid entries (rounded to two decimal places).

(b) Briefly mention the pros and cons of using a macro like `#define SQUARE(X) (X * X)` instead of a function. (7)

Listing 1: C program for question 5(b)

```
1
2  #include <stdio.h>
3  #define SIZE 5
4  int main() {
5      int arr[SIZE];
6      int i;
7      printf("Enter %d numbers:\n", SIZE);
8      for (i = 0; i <= SIZE; i++) {
9          scanf("%d", arr[i]);
10     }
11     int sum;
12     for (i = 0; i < SIZE; i++);
13         sum += arr[i];
14     float avg = sum / SIZE;
15     printf("Average = %f\n", avg);
16     return 0;
17 }
```

Listing 2: C program for question 6(a)

```
1  #include <stdio.h>
2  int main(){
3      int a[] = {0, 1};
4      int b[] = {10, 11, 12};
5      int c[] = {20, 21, 22, 23, 24};
6      int* parr[] = {a, c, b, &c[1]};
7
8      return 0;
9
10 }
```