ಭಾರತೀಯ ಮಾಹಿತಿ ತಂತ್ರಜ್ಞಾನ ಸಂಸ್ಥೆ ರಾಯಚೂರು
भारतीय सूचना प्रौद्योगिकी संस्थान रायचूर
Indian Institute of Information Technology Raichur

Lab - 6

Introduction to Programming (ID110)
Date: December 1, 2024
Topics: Dynamics arrays

Time: 1.5 Hr | CSE'24, Semester - I | Max marks: 10

Instructions:

- The lab session consists of **two programming questions**, and **both are mandatory**.
- External materials (e.g., notes, books) and electronic devices (e.g., mobile phones, smart watch, bluetooth) are **strictly prohibited**. Only a **blank sheet of paper** and a **pen** may be used for rough work.
- Internet usage is **not allowed** under any circumstances. Any violations will lead to **serious academic consequences**, including potential disqualification from the lab.
- Any form of **plagiarism or academic dishonesty** will be treated with the utmost seriousness and may result in severe penalties, including a zero for the lab or further disciplinary actions.
- Code must be written from scratch during the session. Pre-written code snippets or solutions will not be accepted. Use meaningful variable names and add appropriate comments where necessary.
- Upon completion, **two code files** named after **roll no.** (e.g., "CS24B1001-Lab6-p1.c" and "CS24B1001-Lab6-p2.c") must be submitted on **Google Classroom**. Not follow- ing the naming convention will lead to **minus marking**. The submission will only be accepted if done in the presence of TA.
- The input and output format should be the same as the question paper or else you will lose the marks.

---

1. Write a c Program lets the user enter the values for each element and dynamically distributive 3-D array of integers according to user supplied dimensions. The program should calculate the sum of all elements, find the max and min values of the array and display the 3-D array in a structure.

   Input Format:

   - First, an integer *x* representing the depth of elements in the array.
   - Secondly, an integer *y* representing the row of elements in the array.
   - Thirdly, an integer *z* representing the column of elements in the array.
   - Followed by *x,y,z* integers, representing the elements of the array.

   Output Format:

   - If the array is not empty, calculate the **sum of all elements** in the 3D-array and **print it**.
   - If the array is not empty, print:
     - Print: **maximum element index value.**
     - Print: **maximum element.**
   - **print the 3D-array**.
   - If the array is empty, print: **Array is empty**

Examples:

- **Input:**

```
2
3
4
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17
18 19 20 21 22 23 24
```

Output:

```
sum: 300
index: 1 2 3
Max element is: 24
slice-0:
  1  2   3   4
  5  6   7   8
  9 10 11 12
slice-1:
  13 14 15 16
  17 18 19 20
  21 22 23 24
```

- **Input:**

```
3
2
3
3 5 7 9 11 13 2 4 6 8 10 12 1 3 5 7 9 11
```

Output:

```
sum: 138
index: 0 1 2
Max element is: 13
slice-0:
  3  5   7
  9 11 13
slice-1:
  2  4   6
  8 10 12
slice-2:
  1 3  5
  7 9 11
```

- **Input:**

```
0
0
0
```

Output:

```
Array is empty
```

(5 marks)

2. Write a c programming code to print the ND-Arrays using Dynamic arrays. Now the user gives the dimension and then enters the n different depths of the ND-Array then print the array in your terminal as mentioned in the test cases:

   Input Format:

   - An Integer representing the **Dimension(n)**.
   - An 1D-Array of Integers representing the **Depths** of the ND-Array.
   - An *Integer* representing the **elements of the ND-Array**.

   Output Format:

   - Print the **elements of the ND-array.**

   Examples:

   - **Input:**

     ```
     3
     2 2 2
     66 92 96 100 9 39 67 28
     ```

     Output:

     ```
     [[66  92
     96  100 ]
     [9  39
     67  28 ]]
     ```

   - **Input:**

     ```
     5
     2 3 2 3 2
     73 20 4 99 22 16 67 39 21 27 84 40 9
     100 64 47 44 69 25 88 87 75 66 62 25 69
     37 88 77 38 72 59 29 3 81 1 96 74 85 68
     41 93 53 85 92 89 68 20 60 35 92 85 41
     16 79 14 62 4 7 28 82 22 11 87 99 74 86
     92 66 65 52 24
     ```

     Output:

     ```
     [[[[73  20
     4  99
     22  16 ]
     [67  39
     21  27
     84  40 ]]
     [[9  100
     64  47
     44  69 ]
     [25  88
     87  75
     66  62 ]]
     [[25  69
     37  88
     ```

77 38]
[72 59
29 3
81 1]]]
[[[96 74
85 68
41 93]
[53 85
92 89
68 20]]
[[60 35
92 85
41 16]
[79 14
62 4
7 28]]
[[82 22
11 87
99 74]
[86 92
66 65
52 24]]]]

- **Input:**

0

Output:

Array is empty

Explanation:

You can see 5 is the dimension

now,

When the dynamic array is sliced into 2 matrices and then slices into 3 matrices and then slices into 2 Matrices at the end it will form a 2 column × 3 row matrix.

(5 marks)

All the Best!