American International University-Bangladesh

Data Structure Lab
Fall 2023-2024

Marks: 20
Name: RATUL, Mahir Shariar Id: 22-47178-1
Time: 2 hours

---

1) Initialize an integer array of **8 elements** from the user and print the **third smallest** and **third larger** numbers in that array.
Sample output:

Elements of array: **4 1 3 9 6 2 7 5**
Third smallest number: **3**
Third larger number: **6** 2)

Print the following pattern usings **loops**.

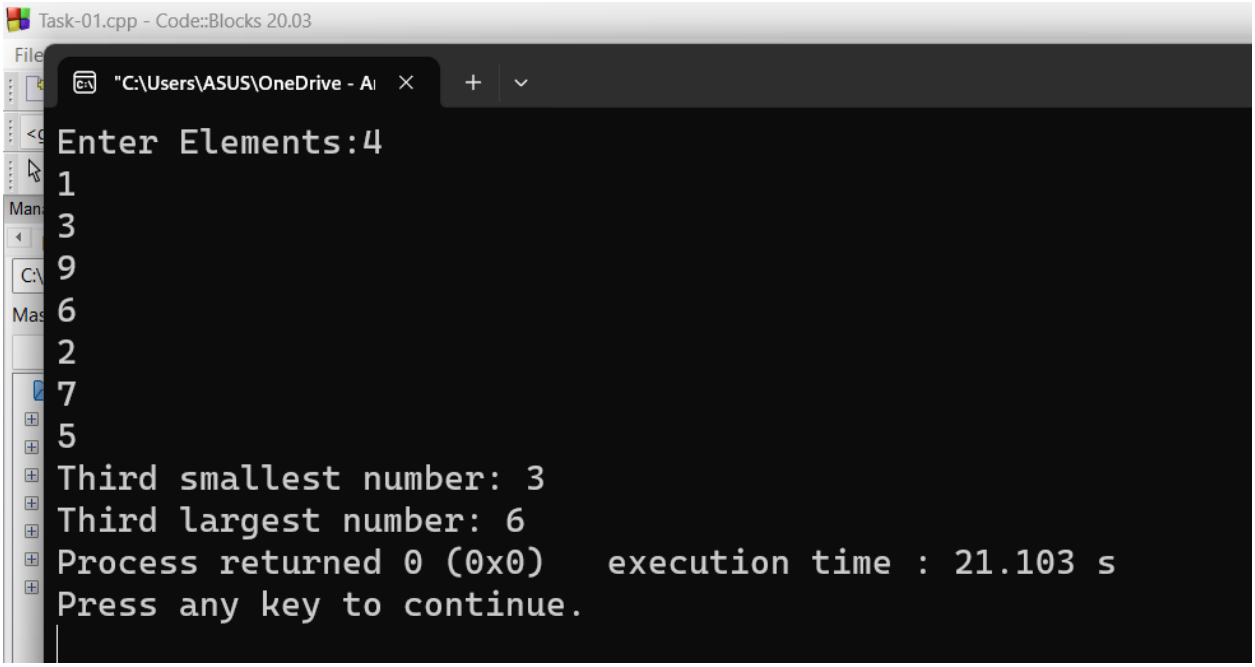
D
DSL
DSLAB
DSLABEX
DSLABEXAM

3) Implement basic stack operations of 20 size array using **class** and **struct**. Necessary functions:
void isEmpty() void isFull() void Push(int element) void pop() void topElement() void show()

Ans to the ques-01:

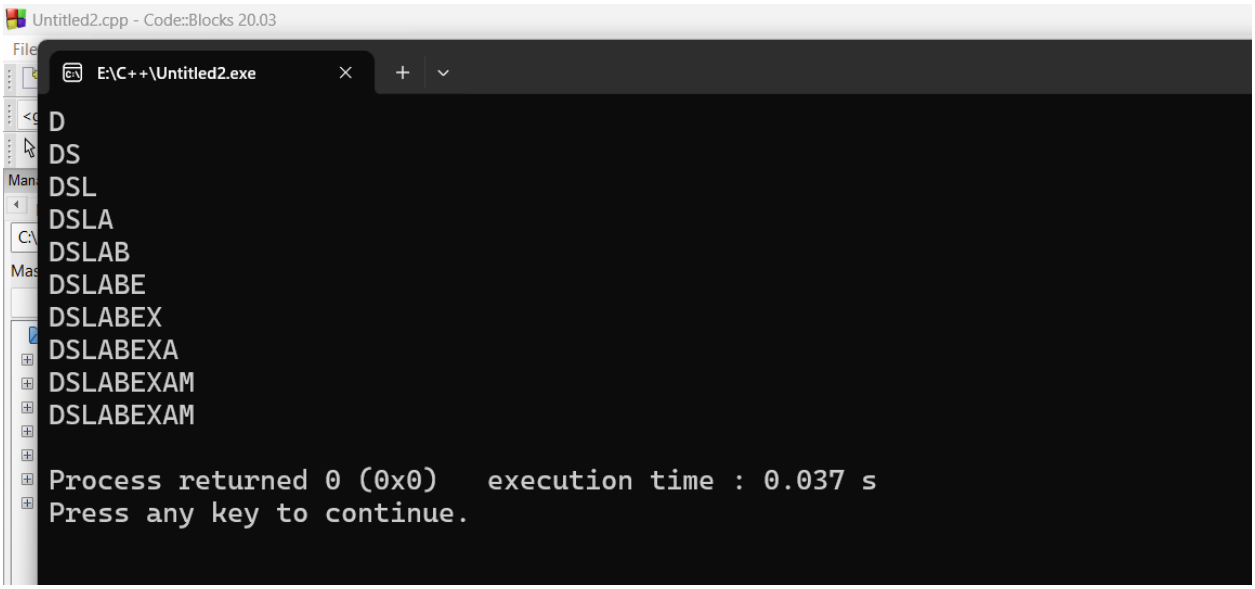
```
#include <iostream>
using namespace std;

int main() {
   int n = 8;
   int arr[n];


  cout<<"Enter Elements:";
   for (int i = 0; i < n; i++) {
```

```
        cin >> arr[i];
    }


    for (int i = 0; i < n - 1; i++) {
        int minIndex = i;
        for (int j = i + 1; j < n; j++) {
            if (arr[j] < arr[minIndex]) {
                minIndex = j;
            }
        }
        if (minIndex != i) {
            int temp = arr[i];
            arr[i] = arr[minIndex];
            arr[minIndex] = temp;
        }
    }


   cout << "Third smallest number: " << arr[2]<<endl;
    cout << "Third largest number: " << arr[n - 3] ;

    return 0;
}
```



Ans to the ques-02:

```
#include <iostream>
using namespace std;

int main() {
    char* baseString = "DSLABEXAM";
    int rows = 10;

    for (int i = 0; i < rows; ++i) {
        for (int j = 0; j <= i; ++j) {
            cout << baseString[j];
        }
        cout << endl;
    }

    return 0;
}
```



Ans to the ques-03:

```
#include <iostream>
using namespace std;

const int MAX_SIZE = 20;

class Stack {
private:
    int data[MAX_SIZE];
    int top;

public:
    Stack() : top(-1) {}

    void isEmpty() {
        cout << (top == -1 ? "Stack is empty" : "Stack is not empty") << endl;
    }

    void isFull() {
        cout << (top == MAX_SIZE - 1 ? "Stack is full" : "Stack is not full") << endl;
    }

    void push(int element) {
```

```
    if (top == MAX_SIZE - 1) {
      cout << "Stack overflow. Cannot push element." << endl;
    } else {
      data[++top] = element;
      cout << "Element " << element << " pushed onto the stack." << endl;
    }
  }

  void pop() {
    if (top == -1) {
      cout << "Stack underflow. Cannot pop element." << endl;
    } else {
      cout << "Element " << data[top--] << " popped from the stack." << endl;
    }
  }

  void topElement() {
    if (top == -1) {
     cout << "Stack is empty. No top element." << endl;
    } else {
      cout << "Top element: " << data[top] << endl;
    }
  }

  void show() {
    if (top == -1) {
      cout << "Stack is empty." << endl;
    } else {
      cout << "Stack elements: ";
      for (int i = 0; i <= top; ++i) {
        cout << data[i] << " ";
      }
      cout <<endl;
    }
  }
};
```

```
int main() {
    Stack stack;

    stack.isEmpty();
    stack.isFull();

    stack.push(10);
    stack.push(20);
    stack.push(30);

    stack.show();
    stack.topElement();

    stack.pop();
    stack.show();
    stack.topElement();

    return 0;
}
```