LAB SHEET -1

AIM 1: Understanding the concept of Array and its Applications (5 points)

An array is a collection of similar data elements. These data elements have the same data type. The elements of the array are stored in consecutive memory locations and are referenced by an index (also known as the subscript). The subscript is an ordinal number which is used to identify an element of the array.

Operations on Arrays

There are a number of operations that can be preformed on arrays. These operations include:

- Traversing an array Inserting an element in an array
- Searching an element in an array
- Deleting an element from an array
- Merging two arrays Sorting an array in ascending or descending order

1) Implement a program for inserting a new element to the specified position of an array.

```
1  package DSA;
2  import java.io.*;
3  import java.lang.*;
4  import java.util.*;
5
6  class InsertElement {
7     public static int[] insertX(int n, int arr[],
8                                 int x, int pos)
9     {
10       int i;
11
12       // creating a new array of size n+1
13       int newarr[] = new int[n + 1];
14       for (i = 0; i < n + 1; i++) {
15          if (i < pos - 1)
16             newarr[i] = arr[i];
17          else if (i == pos - 1)
18             newarr[i] = x;
19          else
20             newarr[i] = arr[i - 1];
21       }
22       return newarr;
23    }
24
25    // Driver code
26    public static void main(String[] args)
27    {
```

Problems @ Javadoc Declaration Console ×

<terminated> InsertElement [Java Application] C:\Program Files\Java\jdk-18.0.2.1\bin\javaw.exe (05

```java
    // Driver code
    public static void main(String[] args)
    {

        int n = 10;
        int i;

        // initial array of size 15
        int arr[]
            = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10,11,12,13,14,15};

        // printing  the original array
        System.out.println("Initial Array:\n"
                           + Arrays.toString(arr));

        // element to be inserted
        int x = 60;

        // position at which element is to be inserted is

        int pos = 4;

        // calling  the method to insert x
        arr = insertX(n, arr, x, pos);

        // print the modified  array here
        System.out.println("\nArray with " + x       + " inserted at position " + pos + ":\n" + Arrays.toString(arr));
```

```java
// Driver code
public static void main(String[] args)
{

    int n = 15;
    int i;
    // initial array of size 15
    int arr[]
        = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15};
    // printing  the original array
    System.out.println("Initial Array:\n"
                       + Arrays.toString(arr));
    // element to be inserted
    int x = 60;
    // position at which element is to be inserted is
    int pos = 4;
    // calling  the method to insert x
    arr = insertX(n, arr, x, pos);
    // print the modified  array here
    System.out.println("\nArray with " + x       + " inserted at position " + pos + ":\n" + Arrays.toString(arr));
}
}
```

```
Problems  @ Javadoc  Declaration  Console ×
<terminated> InsertElement [Java Application] C:\Program Files\Java\jdk-18.0.2.1\bin\javaw.exe  (05-Feb-2023, 2:10:45 pm – 2:10:46 pm) [pid: 2056]
Initial Array:
[1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15]

Array with 60 inserted at position 4:
[1, 2, 3, 60, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15]
```

2) Implement a program for deleting an element from the specified position of an array.

```java
package DSA;
import java.util.Arrays;
public class Removearray {
public static void main(String[] args) {
        int[] arr = new int[]{1,2,3,4,5,6,12};
        int[] arr_new = new int[arr.length-1];
        int j=5;
        for(int i=0, k=0;i<arr.length;i++){
           if(i!=j){
              arr_new[k]=arr[i];
              k++;
           }
        }
        System.out.println("Before deletion :" + Arrays.toString(arr));
        System.out.println("After deletion :" + Arrays.toString(arr_new));
    }
}
```

Problems @ Javadoc Declaration Console ×

<terminated> Removearray [Java Application] C:\Program Files\Java\jdk-18.0.2.1\bin\javaw.exe (27-Jan-2023, 2:46:51 pm – 2:

```
Before deletion :[1, 2, 3, 4, 5, 6, 12]
After deletion :[1, 2, 3, 4, 5, 12]
```

3) Implement a program for sorting a given set of numbers.

```java
package DSA;

   import java.util.Arrays;
   public class SortingArray {

   public static void main(String[] args)
   {
   int [] array = new int[] {101,76,89,45,34,8,42,12};
   Arrays.sort(array);
   System.out.println("Elements of array sorted in ascending order: ");

   for (int i = 0; i < array.length; i++)
   {
   System.out.println(array[i]);

   }
   }
   }
```

Problems @ Javadoc Declaration Console

<terminated> SortingArray (1) [Java Application] C:\Program Files\Java\jdk-18.0.2.1\bin\javaw.exe (19-Jan-2023,

```
Elements of array sorted in ascending order:
8
12
34
42
45
76
89
101
```

AIM 2: Understanding the concepts of stack, its implementations and applications.

(5 points)

- Stack is linear data structure in which addition or deletion takes place at the same end. This end is called the top of stack. Examples of stack are: Stack of plates, Stack of Books etc. Stack is a sequence of items, which can be added and removed from one end only.
- Stack is known as LIFO (last in first out).
- Insert Operation (PUSH) Stacks can be implemented using arrays by defining a structure containing an array and variable to indicate the position of top of stack. PUSH – add data x to stack Increment top and then set data[top]= x

- Delete Operation (POP) POP-remove and return data from stack Return data[top] and decrement top

1) Implement a program for creating a new stack, adding element to the stack,removing elements from stack.

```
1  package DSA;
2
3  public class Stack {
4
5
6        // storing  elements of stack
7         private int arr[];
8
9         private int top;
10
11        private int capacity;
12
13        // Creating a stack!!
14        Stack(int size) {
15
16          arr = new int[size];
17          capacity = size;
18          top = -1;
19        }
20
21
22        public void push(int x) {
```

Problems @ Javadoc Declaration Console ×

<terminated> Stack [Java Application] C:\Program Files\Java\jdk-18.0.2.1\bin\javaw.exe (27-Jan-2

```
Inserting 16
Inserting 24
Inserting 36
Inserting 98
Stack: 16, 24, 36, 98,
After popping out the elements
16, 24, 36,
```

```
22        public void push(int x) {
23          if (isFull()) {
24            System.out.println("Stack OverFlow")
25
26
27            System.exit(1);
28          }
29
30
31          System.out.println("Inserting " + x);
32          arr[++top] = x;
33        }
34
35
36        public int pop() {
37
38          // if stack is empty
39          // no element to pop
40          if (isEmpty()) {
41            System.out.println("Stack is  empty!!"
42            // terminates the program
43            System.exit(1);
```

Problems @ Javadoc Declaration Console

<terminated> Stack [Java Application] C:\Program Files\Java\jdk-18.0.2.1\

```
Inserting 16
Inserting 24
Inserting 36
Inserting 98
Stack: 16, 24, 36, 98,
After popping out the elements
16, 24, 36,
```

```
39          // no element to pop
40          if (isEmpty()) {
41            System.out.println("Stack is  empty!!");
42            // terminates the program
43            System.exit(1);
44          }
45          return arr[top--];
46        }
47        public int getSize() {
48          return top + 1;
49        }
50        // checking  if the stack is empty
51        public Boolean isEmpty() {
52          return top == -1;
53        }
54        // checking  if the stack is full or not
55        public Boolean isFull() {
56          return top == capacity - 1;
57        }
58        public void printStack() {
59          for (int i = 0; i <= top; i++) {
60            System.out.print(arr[i] + ", ");
61          }
62        }
63        public static void main(String[] args) {
64          Stack stack = new Stack(5);
65          stack.push(16);
```

```
        // checking if the stack is empty
        public Boolean isEmpty() {
          return top == -1;
        }
        // checking  if the stack is full or not
        public Boolean isFull() {
          return top == capacity - 1;
        }
        public void printStack() {
          for (int i = 0; i <= top; i++) {
            System.out.print(arr[i] + ", ");
          }
        }
        public static void main(String[] args) {
          Stack stack = new Stack(5);
          stack.push(16);
          stack.push(24);
          stack.push(36);
          stack.push(98);
          System.out.print("Stack: ");
          stack.printStack();
          // removing  element from the  stack
          stack.pop();
          System.out.println("\nAfter popping out the elements");
          stack.printStack();
        }
      }
```

2) Implement a program to reverse a given string using stack.

```java
package DSA;
import java.util.Stack;

public class StackReversal {

    public static Stack<Character> reverseStack(Stack<Character> stack) {
        if (stack.isEmpty()) return stack;
        char element = stack.pop();
        reverseStack(stack);
        insertAtBottom(stack, element);
        return stack;
    }

    private static void insertAtBottom(Stack<Character> stack, char element) {
        if (stack.isEmpty()) {
            stack.push(element);
            return;
        }
        char popped = stack.pop();
        insertAtBottom(stack, element);
        stack.push(popped);
    }

    public static void main(String[] args) {
        Stack<Character> stack = new Stack<>();
        stack.push('A');
        stack.push('B');
```

```java
    public static Stack<Character> reverseStack(Stack<Character> stack) {
        if (stack.isEmpty()) return stack;
        char element = stack.pop();
        reverseStack(stack);
        insertAtBottom(stack, element);
        return stack;
    }
    private static void insertAtBottom(Stack<Character> stack, char element) {
        if (stack.isEmpty()) {
            stack.push(element);
            return;
        }
        char popped = stack.pop();
        insertAtBottom(stack, element);
        stack.push(popped);
    }
    public static void main(String[] args) {
        Stack<Character> stack = new Stack<>();
        stack.push('A');
        stack.push('B');
        stack.push('C');
        stack.push('D');
        Stack<Character> reversedStack = reverseStack(stack);
        System.out.println(reversedStack);
    }
}
```

Problems Javadoc Declaration Console

```
<terminated> StackReversal [Java Application] C:\Program Files\Java\jdk-18.0.2.1\bin\javaw.exe (31-Jan-2023, 1:32:34 pm – 1:32:35 pm) [pid: 7056]
[D, C, B, A]
```