Object Oriented Programming Lab

Ashish Wilson
Roll no : 28
S2MCA_A

8. Program to create a generic stack and do the Push and Pop operations.

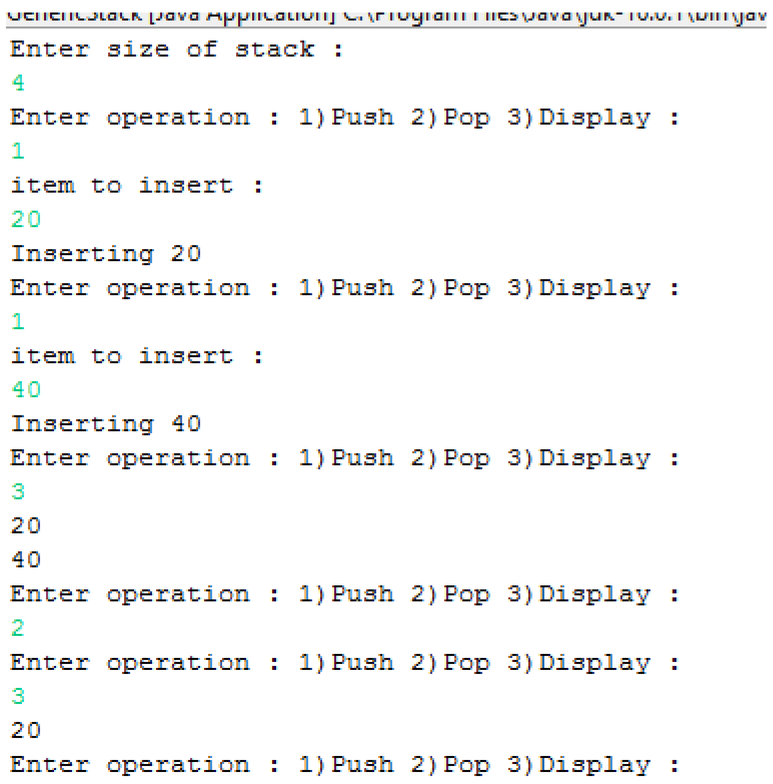
A stack class is provided by the Java collection framework and it implements the Stack data structure. The stack implements LIFO i.e. Last In First Out. This means that the elements pushed last are the ones that are popped first.

1. push() Method adds element x to the stack.
2. pop() Method removes the last element of the stack.
3. top() Method returns the last element of the stack.
4. empty() Method returns whether the stack is empty or not.

```
package javaprj;

import java.util.Scanner;

public class GenericStack {
    private int[] arr;
    private int top;
    private int capacity;

    // Creating a stack
    GenericStack(int size) {
        arr = new int[size];
        capacity = size;
        top = -1;
    }

    // Add elements into stack
    public void push(int x) {
        if (isFull()) {
            System.out.println("OverFlow");
        }
        else{
            System.out.println("Inserting " + x);
            arr[++top] = x;
        }

    }
    // Remove element from stack
    public int pop() {
        if (isEmpty()) {
            System.out.println("STACK EMPTY");
            return -1;
        }
        else {
            return arr[top--];
        }
    }

    // Utility function to return the size of the stack
    public int size() {
        return top + 1;
    }

    // Check if the stack is empty
    public Boolean isEmpty() {
        return top == -1;
    }

    // Check if the stack is full
    public Boolean isFull() {
        return top == capacity - 1;
    }

    public void printStack() {
        for (int i = 0; i <= top; i++) {
            System.out.println(arr[i]);
        }
    }

public static void main(String[] args) {

    Scanner scanner = new Scanner(System.in);
    System.out.println("Enter size of stack : ");
    int count = scanner.nextInt();
    GenericStack stack = new GenericStack(count);

    while (true) {
        System.out.println("Enter operation : 1)Push 2)Pop 3)Display : ");
        int choice = scanner.nextInt();
        switch (choice) {
            case 1:
                System.out.println("item to insert :");
                int item = scanner.nextInt();
```

```
                    stack.push(item);
                    break;
                case 2:
                    stack.pop();
                    break;
                case 3:
                    stack.printStack();
            }
        }
    }
}
```

OUTPUT



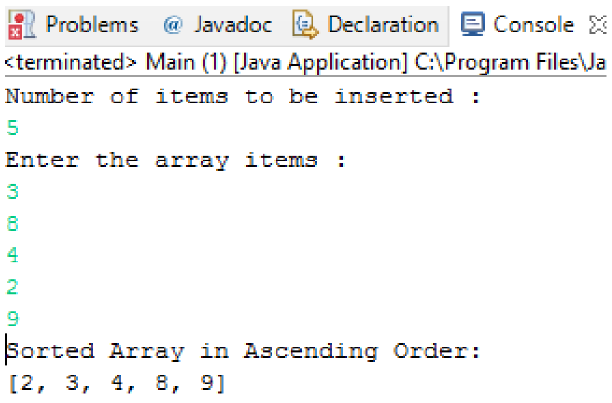
```
GenericStack [Java Application] C:\Program Files\Java\jdk-10.0.1\bin\jav
Enter size of stack :
4
Enter operation : 1)Push 2)Pop 3)Display :
1
item to insert :
20
Inserting 20
Enter operation : 1)Push 2)Pop 3)Display :
1
item to insert :
40
Inserting 40
Enter operation : 1)Push 2)Pop 3)Display :
3
20
40
Enter operation : 1)Push 2)Pop 3)Display :
2
Enter operation : 1)Push 2)Pop 3)Display :
3
20
Enter operation : 1)Push 2)Pop 3)Display :
```

9. Using generic method perform Bubble sort.

Bubble sort is a simple sorting algorithm. This sorting algorithm is a comparison-based algorithm in which each pair of adjacent elements is compared and the elements are swapped if they are not in order. This algorithm is not suitable for large datasets as its average and worst case complexity is of O(n2) where n is the number of items.

```
package javaprj;
import java.util.Arrays;
import java.util.Scanner;


public class Main {
        static void bubbleSort(int array[]) {
            int size = array.length;
            for (int i = 0; i < size - 1; i++)
                for (int j = 0; j < size - i - 1; j++)
                    if (array[j] > array[j + 1]) {

                        int temp = array[j];
                        array[j] = array[j + 1];
                        array[j + 1] = temp;
                    }
        }

        public static void main(String args[]) {

            Scanner scanner = new Scanner(System.in);


            System.out.println("Number of items to be inserted : ");
            int count = scanner.nextInt();

            int[] data = new int[count];

            System.out.println("Enter the array items : ");

            for(int i=0;i<count;i++)
            {
                data[i] = scanner.nextInt();
            }


            Main.bubbleSort(data);

            System.out.println("Sorted Array in Ascending Order:");
            System.out.println(Arrays.toString(data));
        }
}
```

OUTPUT



```
<terminated> Main (1) [Java Application] C:\Program Files\Ja
Number of items to be inserted :
5
Enter the array items :
3
8
4
2
9
Sorted Array in Ascending Order:
[2, 3, 4, 8, 9]
```

10. Maintain a list of Strings using ArrayList from collection framework, perform built-in operations.

The ArrayList class extends AbstractList and implements the List interface. ArrayList supports dynamic arrays that can grow as needed.

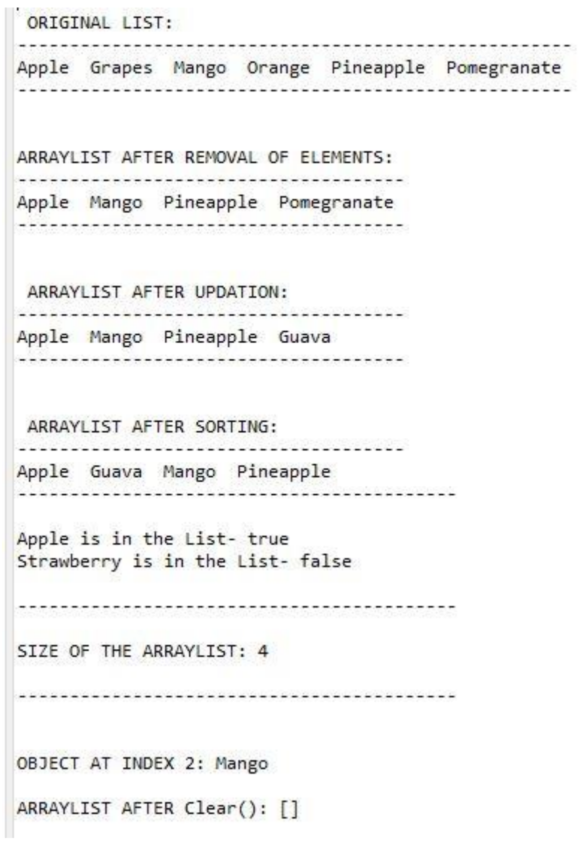
Standard Java arrays are of a fixed length. After arrays are created, they cannot grow or shrink, which means that you must know in advance how many elements an array will hold.

Array lists are created with an initial size. When this size is exceeded, the collection is automatically enlarged. When objects are removed, the array may be shrunk.

```
package javaprj;
import java.util.*;
public class arrayList {
	public static void main(String[] args) {

		// Creating ArrayList of type "String" which means we can only add "String"
elements

		 ArrayList<String> fruits = new ArrayList<String>();

		 //adding elements to an ArrayList
```

```
fruits.add("Apple");
fruits.add("Grapes");
fruits.add("Mango");
fruits.add("Pineapple");
fruits.add("Pomegranate");
fruits.add(3, "Orange");


// Displaying elements

System.out.println("\n ORIGINAL LIST:");
System.out.println("-----------------------------------------");
for(String str : fruits)
   System.out.printf(str+"  ");

//Remove elements from ArrayList

fruits.remove("Grapes");
fruits.remove(2);

// Displaying elements
System.out.println("\n---------------------------------------");
System.out.println("\n\nARRAYLIST AFTER REMOVAL OF ELEMENTS:");
System.out.println("-------------------------------------");
for(String str : fruits )
        System.out.printf(str+"  ");


//Updating the ArrayList

fruits.set(3,"Guava");
System.out.println("\n-------------------------------------");
System.out.println("\n\n ARRAYLIST AFTER UPDATION:");
System.out.println("-------------------------------------");
for(String str : fruits )
        System.out.printf(str+"  ");
System.out.println("\n-------------------------------------");
//Sorting the ArrayList

Collections.sort(fruits);

System.out.println("\n\n ARRAYLIST AFTER SORTING:");
System.out.println("-------------------------------------");
for (String str : fruits)
        System.out.printf(str+"  ");
        // Checks whether the object is in the ArrayList
    System.out.println("\n------------------------------------");
    System.out.println("\nApple is in the List- "+ fruits.contains("Apple"));
    System.out.println("Strawberry is in the Lis"+fruits.contains("Strawberry"));


        //Size of the ArrayList

        System.out.println("\n----------------------------------------");
        System.out.println("\nSIZE OF THE ARRAYLIST: "+ fruits.size());


        //returns the object of list which is present at the specified index
        System.out.println("\n----------------------------------------");
        System.out.println("\n\nOBJECT AT INDEX 2: "+ fruits.get(2));


        // removing all the elements of the ArrayList

        fruits.clear();

        System.out.println("\nARRAYLIST AFTER Clear(): "+ fruits);

    }

}
```

OUTPUT



```
 ORIGINAL LIST:
-----------------------------------------------------
Apple  Grapes  Mango  Orange  Pineapple  Pomegranate
-----------------------------------------------------


ARRAYLIST AFTER REMOVAL OF ELEMENTS:
-------------------------------------
Apple  Mango  Pineapple  Pomegranate
-------------------------------------


 ARRAYLIST AFTER UPDATION:
-------------------------------------
Apple  Mango  Pineapple  Guava
-------------------------------------


 ARRAYLIST AFTER SORTING:
-------------------------------------
Apple  Guava  Mango  Pineapple
------------------------------------------

Apple is in the List- true
Strawberry is in the List- false

------------------------------------------

SIZE OF THE ARRAYLIST: 4

------------------------------------------


OBJECT AT INDEX 2: Mango

ARRAYLIST AFTER Clear(): []
```