LAB - 3

Java Programming Lab

Topics Covered

Arrays, Methods, Method Overloading

Syeda Reeha Quasar
14114802719
4C7

EXPERIMENT – 3.1

Aim:

Write a program to find out the array index or position where sum of numbers preceding the index is equals to sum of numbers succeeding the index.

Theory:

Given, an array of size n. Find an element index that divides the array into two sub-arrays with equal sums.

Examples:

Input: 1 4 2 5

Output: 2

Explanation: If 2 is the partition, subarrays are : {1, 4} and {5}

Concepts:

- **class** keyword is used to declare a class in Java.
- **public** keyword is an access modifier that represents visibility. It means it is visible to all.
- **static** is a keyword. If we declare any method as static, it is known as the static method. The core advantage of the static method is that there is no need to create an object to invoke the static method. The main() method is executed by the JVM, so it doesn't require creating an object to invoke the main() method. So, it saves memory.
- **void** is the return type of the method. It means it doesn't return any value.
- **main** represents the starting point of the program.
- **String[] args** or **String args[]** is used for command line argument. We will discuss it in coming section.
- **System.out.println()** is used to print statement. Here, System is a class, out is an object of the PrintStream class, println() is a method of the PrintStream class. We will discuss the internal working of System.out.println() statement in the coming section.

- **Java Utils:** Resources Job Search Discussion. Java. util package contains the **collections framework, legacy collection classes, event model, date and time facilities, internationalization, and miscellaneous utility classes**. This reference will take you through simple and practical methods available in java.
- util. Java util package contains **collection framework, collection classes**, classes related to date and time, event model, internationalization, and miscellaneous utility classes. ... On importing this package, you can access all these classes and methods.
- **Scanner** is a class in **java. util package used for obtaining the input of** the primitive types like int, double, etc. and strings. ... To create an object of Scanner class, we usually pass the predefined object System.in, which represents the standard input stream.
- A switch statement **allows a variable to be tested for equality against a list of values**. Each value is called a case, and the variable being switched on is checked for each case.
- The input is **the data that we give to the program**. The output is the data what we receive from the program in the form of result. Stream represents flow of data or the sequence of data.

```
Input: 1 4 2 5
Output: 2
Explanation: If 2 is the partition,
      subarrays are : {1, 4} and {5}

Input: 2 3 4 1 4 5
Output: 1
Explanation: If 1 is the partition,
 Subarrays are : {2, 3, 4} and {4, 5}
```

Source Code:

```
// Program to find an element such that sum of right side element is equal to sum
of left side
public class sumFragment {

    // Finds an element in an array such that
    // left and right side sums are equal
    static int findElement(int arr[], int n) {
        // Forming prefix sum array from 0
        int[] prefixSum = new int[n];
        prefixSum[0] = arr[0];
        for (int i = 1; i < n; i++)
            prefixSum[i] = prefixSum[i - 1] + arr[i];

        // Forming suffix sum array from n-1
        int[] suffixSum = new int[n];
        suffixSum[n - 1] = arr[n - 1];
        for (int i = n - 2; i >= 0; i--)
            suffixSum[i] = suffixSum[i + 1] + arr[i];

        // Find the point where prefix and suffix
        // sums are same.
        for (int i = 1; i < n - 1; i++)
            if (prefixSum[i] == suffixSum[i])
                return i; // idex where equal

        return -1;
    }

    public static void main(String args[]) {
        int arr[] = { 1, 4, 3, 5 };
        int n = arr.length;
        System.out.println(findElement(arr, n));
    }
}
```

Output:

```
PS E:\sem 5\java> javac .\sumFragment.java
PS E:\sem 5\java> java sumFragment
2
```

```
int arr[] = { 1, 4, 2, 3, 5 };
```

```
PS E:\sem 5\java> javac .\sumFragment.java
PS E:\sem 5\java> java sumFragment
-1
```

EXPERIMENT – 3.2

Aim:

Write a program that creates and initializes a four-element int array. Calculate and display the average of its values.

Theory:

Enter size of array and then enter all the elements of that array. Now using for loop we calculate sum of elements of array and hence we divide it by number of elements in array to get average.

Here is the source code of the Java Program to Calculate Sum & Average of an Array. The Java program is successfully compiled and run on a Windows system.

Concepts:

- **class** keyword is used to declare a class in Java.
- **public** keyword is an access modifier that represents visibility. It means it is visible to all.
- **static** is a keyword. If we declare any method as static, it is known as the static method. The core advantage of the static method is that there is no need to create an object to invoke the static method. The main() method is executed by the JVM, so it doesn't require creating an object to invoke the main() method. So, it saves memory.
- **void** is the return type of the method. It means it doesn't return any value.
- **main** represents the starting point of the program.
- **String[] args** or **String args[]** is used for command line argument. We will discuss it in coming section.
- **System.out.println()** is used to print statement. Here, System is a class, out is an object of the PrintStream class, println() is a method of the PrintStream class. We will discuss the internal working of System.out.println() statement in the coming section.
- **Scanner** is a class in **java. util package used for obtaining the input of** the primitive types like int, double, etc. and strings. ... To create an object of Scanner

class, we usually pass the predefined object System.in, which represents the standard input stream.

Source Code:

```
import java.util.Scanner;


public class avgArray {

  public static void main(String[] args) {
    double[] arr = new double[4];
    double total = 0;

    Scanner sc = new Scanner(System.in);

    System.out.println("Enter 4 elements for which you want to calculate average:
");
    for (int i = 0; i < arr.length; i++) {
      arr[i] = sc.nextDouble();
      total = total + arr[i];
    }

    double average = total / arr.length;

    /*
     * This is used for displaying the formatted output if you give %.4f then the
     * output would have 4 digits after decimal point.
     */
    System.out.format("The average is: %.3f", average);
  }
}
```

Output:

```
PS E:\sem 5\java> javac .\avgArray.java
PS E:\sem 5\java> java avgArray
Enter 4 elements for which you want to calculate average:
12 3 4 5
The average is: 6.000
```

```
PS E:\sem 5\java> java avgArray
Enter 4 elements for which you want to calculate average:
20
40
23
87
The average is: 42.500
```

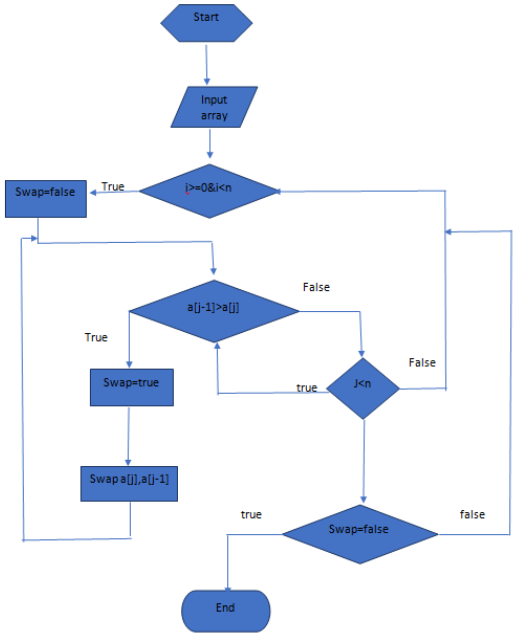
EXPERIMENT – 3.3

Aim:

WAP using Bubble sort for sorting in ascending Order.

Theory:

Bubble Sort is the simplest sorting algorithm that works by repeatedly swapping the adjacent elements if they are in wrong order.



i = 0	j	0	1	2	3	4	5	6	7
	0	5	3	1	9	8	2	4	7
	1	3	5	1	9	8	2	4	7
	2	3	1	5	9	8	2	4	7
	3	3	1	5	9	8	2	4	7
	4	3	1	5	8	9	2	4	7
	5	3	1	5	8	2	9	4	7
	6	3	1	5	8	2	4	9	7
i =1	0	3	1	5	8	2	4	7	9
	1	1	3	5	8	2	4	7	
	2	1	3	5	8	2	4	7	
	3	1	3	5	8	2	4	7	
	4	1	3	5	2	8	4	7	
	5	1	3	5	2	4	8	7	
i = 2	0	1	3	5	2	4	7	8	
	1	1	3	5	2	4	7		
	2	1	3	5	2	4	7		
	3	1	3	2	5	4	7		
	4	1	3	2	4	5	7		
i = 3	0	1	3	2	4	5	7		
	1	1	3	2	4	5			
	2	1	2	3	4	5			
	3	1	2	3	4	5			
i = 4	0	1	2	3	4	5			
	1	1	2	3	4				
	2	1	2	3	4				
i = 5	0	1	2	3	4				
	1	1	2	3					
i = 6	0	1	2	3					
		1	2						

Source Code:

```
import java.util.Scanner;


public class bubbleSort {
    void sort(int[] arr, int n) {
        int temp;
        for (int i = 0; i < n - 1; i++)
            for (int j = 0; j < n - i - 1; j++)
                if (arr[j] > arr[j + 1]) {
                    temp = arr[j];
                    arr[j] = arr[j + 1];
                    arr[j + 1] = temp;
                }
        System.out.println("Sorted Array:");
        for (int i = 0; i < n; i++)
            System.out.print(arr[i] + " ");

    }


    public static void main(String[] args) {
        bubbleSort sort = new bubbleSort();
        int n;
        Scanner sc = new Scanner(System.in);
        System.out.print("Enter the size of array ");
        n = sc.nextInt();
        var arr = new int[n];
        System.out.println("Enter the array");
        for (int i = 0; i < n; i++)
            arr[i] = sc.nextInt();
        sc.close();
        sort.sort(arr, n);
    }

}
```

Output:

```
PS E:\sem 5\java> javac .\bubbleSort.java
PS E:\sem 5\java> java bubbleSort
Enter the size of array 10
Enter the array
12 43 1 100 23 90 22 15 4 7
Sorted Array:
1 4 7 12 15 22 23 43 90 100
```

```
PS E:\sem 5\java> java bubbleSort
Enter the size of array 5
Enter the array
23
7
80
10
12
Sorted Array:
7 10 12 23 80
```

EXPERIMENT – 3.4

Aim:

Create a java program to implement stack and queue concept.

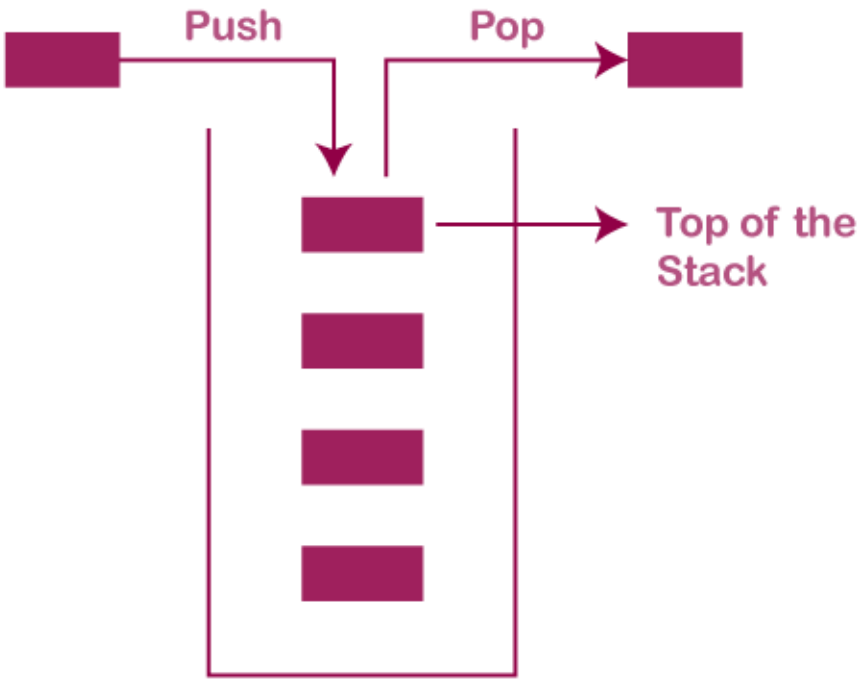
Theory:

Java Stack

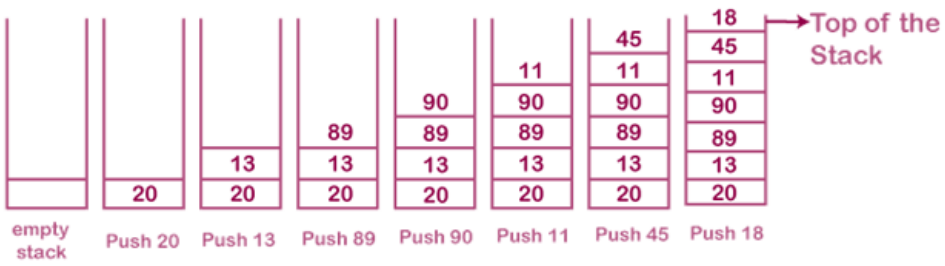
The stack is a linear data structure that is used to store the collection of objects. It is based on Last-In-First-Out (LIFO). Java collection framework provides many interfaces and classes to store the collection of objects. One of them is the Stack class that provides different operations such as push, pop, search, etc.

In this section, we will discuss the Java Stack class, its methods, and implement the stack data structure in a Java program. But before moving to the Java Stack class have a quick view of how the stack works.

The stack data structure has the two most important operations that are push and pop. The push operation inserts an element into the stack and pop operation removes an element from the top of the stack. Let's see how they work on stack.

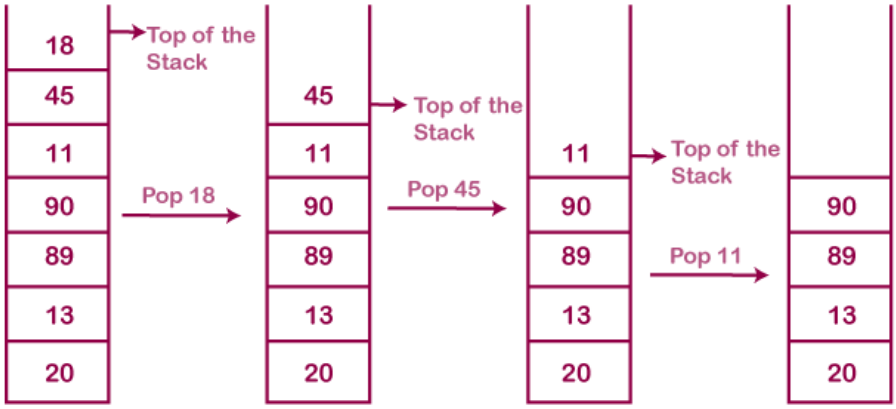


Let's push 20, 13, 89, 90, 11, 45, 18, respectively into the stack.



Push operation

Let's remove (pop) 18, 45, and 11 from the stack.



Pop operation

Queue Interface In Java

The Queue interface present in the java.util package and extends the Collection interface is used to hold the elements about to be processed in FIFO(First In First Out) order. It is an ordered list of objects with its use limited to insert elements at the end of the list and deleting elements from the start of the list, (i.e.), it follows the FIFO or the First-In-First-Out principle.

Being an interface the queue needs a concrete class for the declaration and the most common classes are the PriorityQueue and LinkedList in Java.It is to be noted that both the implementations are not thread safe. PriorityBlockingQueue is one alternative implementation if thread safe implementation is needed.

Source Code:

```
import java.util.ArrayList;
import java.util.List;
import java.util.Scanner;
public class stackUsingArrayList {
    public static void main(String[] args) {
        Scanner sc=new Scanner(System.in);
        List<Integer> arrayList=new ArrayList<>();
        int val;
        System.out.println("Enter your choice: (1-Stack, 2-Queue)");
        val=sc.nextInt();
        if(val==1){
        System.out.println("Push 3 values for Stack");
        for(int i=0;i<3;i++){
            val=sc.nextInt();
            arrayList.add(val);
        }
        System.out.println("Popping element from stack");
        int valuePop=arrayList.get(arrayList.size()-1);
        arrayList.remove(arrayList.size()-1);
        System.out.println(valuePop+" popped from stack");
    }
        else{
            System.out.println("Enter 3 values for Queue");
        for(int i=0;i<3;i++){
            val=sc.nextInt();
            arrayList.add(val);
```

```
        }
        System.out.println("Removing element from queue");
        int valuePop=arrayList.get(0);
        arrayList.remove(0);
        System.out.println(valuePop+" removed from queue");
        }
        sc.close();
    }
}
```

Output:

```
Enter your choice: (1-Stack, 2-Queue)
1
Push 3 values for Stack
55
24
35
Popping element from stack
35 popped from stack
PS C:\Users\Volted User\OneDrive\Desktop\practical>
```

```
Enter your choice: (1-Stack, 2-Queue)
2
Enter 3 values for Queue
22
41
36
Removing element from queue
22 removed from queue
PS C:\Users\Volted User\OneDrive\Desktop\practical>
```

EXPERIMENT – 3.5

Aim:

Using the concept of method overloading Write method for calculating the area of triangle, circle and rectangle.

Theory:

This is a Java Program to Find Area of Square, Rectangle and Circle using Method Overloading.

We declare three methods of same name but with different number of arguments or with different data types. Now when we call these methods using objects, corresponding methods will be called as per the number of arguments or their datatypes.

Here is the source code of the Java Program to Find Area of Square, Rectangle and Circle using Method Overloading. The Java program is successfully compiled and run on a Windows system. The program output is also shown below.

In this program, we will see how to find the area of a square, rectangle, and circle using Method Overloading.

Algorithm:

1. Start
2. Declare three different classes for rectangle, square, and circle.
3. Declare two methods of the same name but with a different number of arguments or with different data types.
4. Call these methods using objects.
5. Call the corresponding methods as per the number of arguments or their data types.
6. Display the result.
7. Stop.

To understand this example, you should have the knowledge of the following Java programming topics:

- Java if...else Statement
- Java Scanner Class
- **Java Utils:** Resources Job Search Discussion. Java. util package contains the **collections framework, legacy collection classes, event model, date and time facilities, internationalization, and miscellaneous utility classes**. This reference will take you through simple and practical methods available in java.
- util. Java util package contains **collection framework, collection classes**, classes related to date and time, event model, internationalization, and miscellaneous utility classes. ... On importing this package, you can access all these classes and methods.
- **Scanner** is a class in **java. util package used for obtaining the input of** the primitive types like int, double, etc. and strings. ... To create an object of Scanner class, we usually pass the predefined object System.in, which represents the standard input stream.

Source Code:

```
import java.util.Scanner;

public class p5 {

    public float area(int a,int b) {

        return (float)a*b;    }

    public float area(float r) {

            return (22*r*r/7); }

    public float area(float h,float b) {

                return h*b/2;      }

    public static void main(String[] args) {

        p5 ar=new p5();

        Scanner sc=new Scanner(System.in);

        int ch;

        System.out.println("Enter choice of area (1-Triangle , 2-Circle , 3-
Rectangle) ");

        ch=sc.nextInt();
```

```
        switch(ch){
            case 1:float h,b;
                System.out.print("Enter height and base of triangle: ");
                h=sc.nextFloat();
                b=sc.nextFloat();
                System.out.println("Area of Traingle: "+ar.area(h,b));
                break;
            case 2:float r;
                System.out.print("Enter radius of circle: ");
                r=sc.nextFloat();
                System.out.println("Area of Circle: "+ar.area(r));
                break;
            case 3:int a,c;
                System.out.print("Enter length and breadth of triangle: ");
                a=sc.nextInt();
                c=sc.nextInt();
                System.out.println("Area of Rectangle: "+ar.area(a,c));
                break;
            }
        sc.close();
    }
}
```

Output:

```
Enter choice of area (1-Triangle , 2-Circle , 3-Rectangle)
1
Enter height and base of triangle: 5
3
Area of Traingle: 7.5
PS C:\Users\Volted User\OneDrive\Desktop\practical>
```

```
Enter choice of area (1-Triangle , 2-Circle , 3-Rectangle)
2
Enter radius of circle: 4
Area of Circle: 50.285713
PS C:\Users\Volted User\OneDrive\Desktop\practical>
```

```
Enter choice of area (1-Triangle , 2-Circle , 3-Rectangle)
3
Enter length and breadth of Rectangle: 6
3
Area of Rectangle: 18.0
PS C:\Users\Volted User\OneDrive\Desktop\practical>
```

Viva Questions

1. What is difference between Path and Class path?

Ans.

Path and Class path are operating system level environment variables. Path is used define where the system can find the executables(.exe) files and class path is used to specify the location .class files.

2. What are local variables?

Ans.

Local variables are those which are declared within a block of code like methods. Local variables should be initialized before accessing them.

3. Can a class declared as private be accessed outside its package?

Ans.

Not possible.

4. Can a class be declared as protected?

Ans.

A class can't be declared as protected. only methods can be declared as protected.

5. What is the access scope of a protected method?

Ans.

A protected method can be accessed by the classes within the same package or by the subclasses of the class in any package.