Object Oriented Programming with Java
Practical-2
Basic Programming in Java

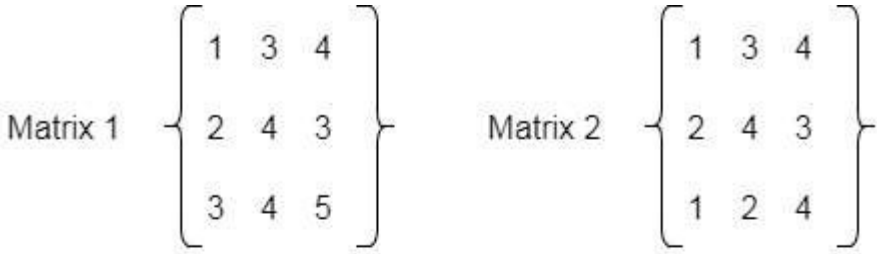
How to Get Size, Minimum, and Maximum Value of Data Types in Java?

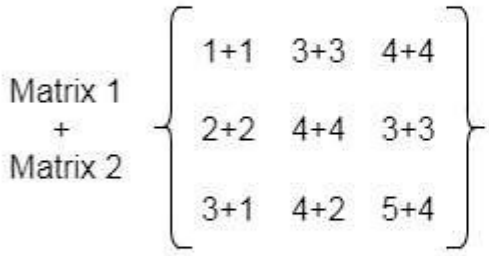
The size of a data type is given by (name of datatype).SIZE. The maximum value that it can store is given by (Name of data type).MAX_VALUE. The minimum value that it can store is given by (Name of data type).MIN_VALUE.

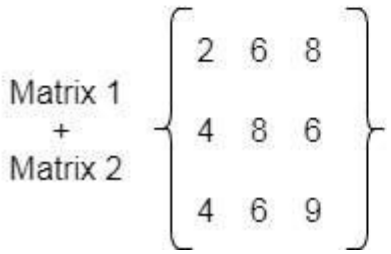
Always write first word of data type in capital.

Java Program to add two matrices

We can add two matrices in java using binary + operator. A matrix is also known as array of arrays. We can add, subtract and multiply matrices.







To subtract two matrices, use – operator and multiplication using * operator.

Java Scanner

Scanner class in Java is found in the java.util package. Java provides various ways to read input from the keyboard, the java.util.Scanner class is one of them.

The Java Scanner class breaks the input into tokens using a delimiter which is whitespace by default. It provides many methods to read and parse various primitive values.

The Java Scanner class provides nextXXX() methods to return the type of value such as nextInt(), nextByte(), nextShort(), next(), nextLine(), nextDouble(), nextFloat(), nextBoolean(), etc.

```
import java.util.*;
public class ScannerExample {
public static void main(String args[]){
        Scanner in = new Scanner(System.in);
        System.out.print("Enter your name: ");
        String name = in.nextLine();
        System.out.println("Name is: " + name);
        in.close();
        }
}
```

Program for addition of two matrices

```
import java.util.Scanner;

public class MatrixAddition {

   public static void main(String...args) {
        Scanner scanner = new Scanner(System.in);
        System.out.print("Enter number of rows in matrix : "); //rows and columns in matrix1 and matrix2 must
be same for addition.
        int rows = scanner.nextInt();
        System.out.print("Enter number of columns in matrix : ");
        int columns = scanner.nextInt();
        int[][] matrix1 = new int[rows][columns];
        int[][] matrix2 = new int[rows][columns];

        System.out.println("Enter the elements in first matrix :");
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < columns; j++) {
                matrix1[i][j] = scanner.nextInt();
            }
        }
        System.out.println("Enter the elements in second matrix :");
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < columns; j++) {
                matrix2[i][j] = scanner.nextInt();
            }
        }

        //addition of matrices.
        int[][] resultMatix = new int[rows][columns];
        for (int i = 0; i < rows; i++) {
```

```
            for (int j = 0; j < columns; j++) {
                resultMatix[i][j] = matrix1[i][j] + matrix2[i][j];
            }
        }

        System.out.println("\nFirst matrix is : ");
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < columns; j++) {
                System.out.print(matrix1[i][j] + " ");
            }
            System.out.println();
        }

        System.out.println("\nSecond matrix is : ");
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < columns; j++) {
                System.out.print(matrix2[i][j] + " ");
            }
            System.out.println();
        }

        System.out.println("\nThe sum of the two matrices is : ");
        for (int i = 0; i < rows; i++) {
            for (int j = 0; j < columns; j++) {
                System.out.print(resultMatix[i][j] + " ");
            }
            System.out.println();
        }
    }
}
```

Class, Data Member and Method

we have created a Student class which has two data members id and name. We are creating the object of the Student class by new keyword and printing the object's value.
Here, we are creating a main() method inside the class.

```
//Java Program to illustrate how to define a class and fields
//Defining a Student class.
public class Student{
 //defining fields
    int id;//field or data member or instance variable
    String name;
     //creating main method inside the Student class
    public static void main(String args[]){
    //Creating an object or instance
            Student s1=new Student();//creating an object of Student
    //Printing values of the object
            System.out.println(s1.id);//accessing member through reference variable
           System.out.println(s1.name);
     }
}
```

3 Ways to initialize object

There are 3 ways to initialize object in Java.

1. By reference variable
2. By method
3. By constructor

1) Object and Class Example: Initialization through reference

Initializing an object means storing data into the object. Let's see a simple example where we are going to initialize the object through a reference variable.

```
class Student1{
	int id;
	String name;
}
class TestStudent2{
	public static void main(String args[]){
	 Student s1=new  Student();
	 s1.id=101;
	 s1.name="Vivek";
	 System.out.println(s1.id+" "+s1.name);//printing members with a white space
 }
}
```

2) Object and Class Example: Initialization through method

In this example, we are creating the two objects of Student class and initializing the value to these objects by invoking the insertRecord method. Here, we are displaying the state (data) of the objects by invoking the displayInformation() method.

```
class Student{
	int rollno;
	String name;

	void insertRecord(int r, String n){
		rollno=r;
		name=n;
	}
	void displayInformation(){
		System.out.println(rollno+" "+name);
	}
}

class TestStudent4{
	public static void main(String args[]){
	Student s1=new Student();
	Student s2=new Student();

	s1.insertRecord(111,"Vivek");
	s2.insertRecord(222,"Vyas");
```

```
        s1.displayInformation();
        s2.displayInformation();
    }
}
```

3) Object and Class Example: Initialization through a constructor

---

Object and Class Example: Employee

Let's see an example where we are maintaining records of employees.

```
class Employee{
    int id;
    String name;
    float salary;
    Employee(int i, String n, float s) {
        id=i;
        name=n;
        salary=s;
    }
    void display(){System.out.println(id+" "+name+" "+salary);}
}
public class TestEmployee {
public static void main(String[] args) {
    Employee e1=new Employee(101,"vivek",45000);
    Employee e2=new Employee(102,"utsav",25000);
    Employee e3=new Employee(103,"himanshu",55000);
    e1.display();
    e2.display();
    e3.display();
}
}
```

Getter and Setter Method

```
public class GetterSetterExample
    {
        int salary;

        // a setter method that assign a
        // value to the salary variable
        void setSalary(int s)
        {
          salary = s;
        }


        // a getter mehtod to retrieve
        // the salary
        int getSalary()
        {
            return salary;
        }
```

```
    public void storeSalaryDB(int salary)
    {
        // code for storing the salary in the database
         System.out.println("Salary is "+salary);
    }

    // main method
    public static void main(String args[])
    {
        // creating an object of the class GetterSetterExample
        GetterSetterExample obj = new GetterSetterExample();

      obj.setSalary(5000);

      int salary = obj.getSalary();

      // storing salary in database
      obj.storeSalaryDB(salary);

    }
}
```