1. Java program for Matrix Addition.

**package** com.rsc.babystepprogram;

**import** java.util.\*;

**public** **class** MatrixAddition {

Scanner scan = **new** Scanner(System.***in***);

**public** **int**[][] readMatrix(**int** rows, **int** columns) {

**int** matrix[][] = **new** **int**[rows][columns];

String input[] = scan.nextLine().split(" ");

**for**(**int** k=0,i=0; i<rows; i++)

**for**(**int** j=0; j<columns; j++)

matrix[i][j] = Integer.*parseInt*(input[k++]);

**return** matrix;

}

**public** **int**[][] addMatrices(**int**[][] matrixA, **int**[][] matrixB, **int** rows, **int** columns) {

**for**(**int** i=0; i<rows; i++)

**for**(**int** j=0; j<columns; j++)

matrixA[i][j] += matrixB[i][j];

**return** matrixA;

}

**public** **void** printMatrix(**int**[][] matrixC, **int** rows, **int** columns) {

**for**(**int**[] row: matrixC)

System.***out***.println(Arrays.*toString*(row));

}

**public** **static** **void** main(String[] args) {

Scanner scan = **new** Scanner(System.***in***);

**try** {

System.***out***.print("Enter the number of rows: ");

**int** rows = scan.nextInt();

System.***out***.print("Enter the number of columns: ");

**int** columns = scan.nextInt();

**if**(rows > 0 && columns > 0) {

System.***out***.println("Enter elements of matrix A: ");

**int**[][] matrixA = **new** MatrixAddition().readMatrix(rows, columns);

System.***out***.println("Matrix A: ");

**new** MatrixAddition().printMatrix(matrixA, rows, columns);

System.***out***.println("Enter elements of matrix B: ");

**int**[][] matrixB = **new** MatrixAddition().readMatrix(rows, columns);

System.***out***.println("Matrix B: ");

**new** MatrixAddition().printMatrix(matrixB, rows, columns);

System.***out***.println("Sum of two matrices: ");

**int**[][] matrixC = **new** MatrixAddition().addMatrices(matrixA, matrixB, rows, columns);

**new** MatrixAddition().printMatrix(matrixC, rows, columns);

}

**else** {

System.***out***.println("Matrix dimensions must be greater than 0");

}

}

**catch**(Exception e) {

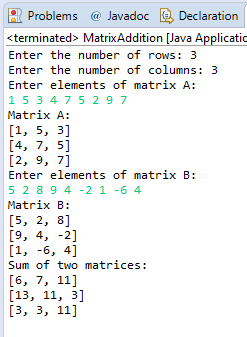
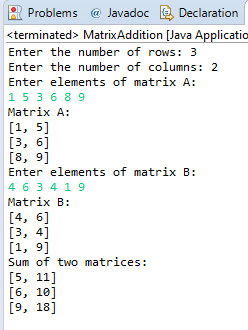
System.***out***.println("Input mismatch exception: "+e);

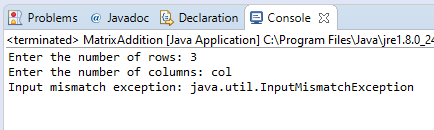
}

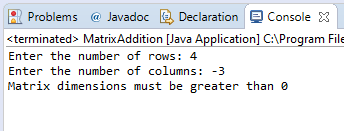
scan.close();

}

}





1. Java program for Matrix Multiplication.

**package** com.rsc.babystepprogram;

**import** java.util.\*;

**public** **class** MatrixMultiplication {

Scanner scan = **new** Scanner(System.***in***);

**public** **int**[][] readMatrix(**int** rows, **int** columns) {

**int** matrix[][] = **new** **int**[rows][columns];

String input[] = scan.nextLine().split(" ");

**for**(**int** k=0,i=0; i<rows; i++)

**for**(**int** j=0; j<columns; j++)

matrix[i][j] = Integer.*parseInt*(input[k++]);

**return** matrix;

}

**public** **void** printMatrix(**int**[][] matrixC, **int** rows, **int** columns) {

**for**(**int**[] row: matrixC)

System.***out***.println(Arrays.*toString*(row));

}

**Publ**

**ic** **int**[][] productMatrices(**int**[][] matrixA, **int**[][] matrixB, **int** r1,

**int** c1, **int** r2, **int** c2) {

**int** product[][] = **new** **int**[r1][c2];

**if**(c1 == r2) {

**for**(**int** i=0; i<r1; i++)

**for**(**int** j=0; j<c2; j++)

**for**(**int** k=0; k<r2; k++)

product[i][j] += matrixA[i][k]\*matrixB[k][j];

} **else**

System.***out***.println("matrix multiplication is not possible");

**return** product;

}

**public** **static** **void** main(String[] args) {

Scanner scan = **new** Scanner(System.***in***);

**try** {

System.***out***.print("Enter the number of rows of matrix A: ");

**int** rowsA = scan.nextInt();

System.***out***.print("Enter the number of columns of matrix A: ");

**int** columnsA = scan.nextInt();

System.***out***.print("Enter the number of rows of matrix B: ");

**int** rowsB = scan.nextInt();

System.***out***.print("Enter the number of columns of matrix B: ");

**int** columnsB = scan.nextInt();

**if**(rowsA > 0 && columnsA > 0 && rowsB > 0 && columnsB > 0) {

**if**(columnsA == rowsB) {

System.***out***.println("Enter elements of matrix A: ");

**int**[][] matrixA = **new** MatrixMultiplication().readMatrix(rowsA, columnsA);

System.***out***.println("Matrix A: ");

**new** MatrixMultiplication().printMatrix(matrixA, rowsA, columnsA);

System.***out***.println("Enter elements of matrix B: ");

**int**[][] matrixB = **new** MatrixMultiplication().readMatrix(rowsB, columnsB);

System.***out***.println("Matrix B: ");

**new** MatrixMultiplication().printMatrix(matrixB, rowsB, columnsB);

System.***out***.println("Product of two matrices: ");

**int**[][] matrixC = **new** MatrixMultiplication().productMatrices(matrixA, matrixB, rowsA, columnsA, rowsB, columnsB);

**new** MatrixMultiplication().printMatrix(matrixC, rowsA, columnsB);

} **else**

System.***out***.println("matrix multiplication is not possible");

} **else** {

System.***out***.println("Matrix dimensions must be greater than 0");

}

}

**catch**(Exception e) {

System.***out***.println("Input mismatch exception: "+e);

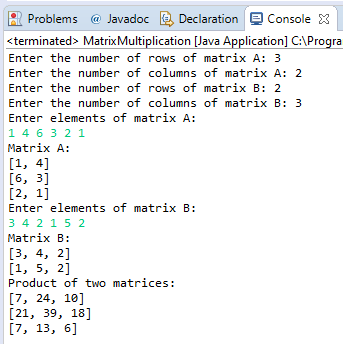
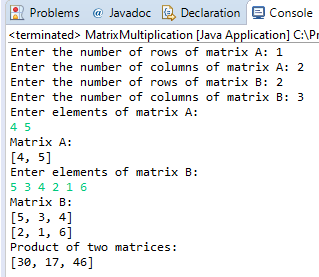
}

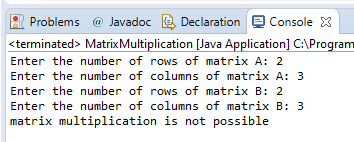
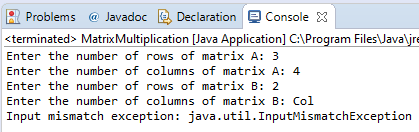
scan.close();

}

}

Output:

1. Java program to demonstrate method overloading.

**package** com.rsc.babystepprogram;

**import** java.util.\*;

**public** **class** MethodOverloadingExample {

**public** **int** findVolume(**int** side) {

**return** side\*side\*side;

}

**public** **float** findVolume(**float** l, **float** b, **float** h) {

**return** l\*b\*h;

}

**public** **double** findVolume(**float** r, **float** l) {

**return** Math.***PI***\*r\*r\*l;

}

**public** **static** **void** main(String[] args) {

Scanner scan =**new** Scanner(System.***in***);

MethodOverloadingExample cube = **new** MethodOverloadingExample();

MethodOverloadingExample cuboid = **new** MethodOverloadingExample();

MethodOverloadingExample cylinder = **new** MethodOverloadingExample();

**try** {

System.***out***.println("Enter the side of the square: ");

**int** side = scan.nextInt();

scan.nextLine();

System.***out***.println("Enter the dimensions of the cuboid (l/b/h): ");

String input[] = scan.nextLine().split(" ");

**float** length = Float.*parseFloat*(input[0]);

**float** breadth = Float.*parseFloat*(input[1]);

**float** height = Float.*parseFloat*(input[2]);

System.***out***.println("Enter the radius and height of cylinder: ");

String input1[] = scan.nextLine().split(" ");

**float** radius = Float.*parseFloat*(input1[0]);

**float** len = Float.*parseFloat*(input[1]);

System.***out***.println("Volume of the cube : "+cube.findVolume(side));

System.***out***.println("Volume of the cuboid : "+cuboid.findVolume(length, breadth, height));

System.***out***.println("Volume of the cylinder : "+cylinder.findVolume(radius, len));

scan.close();

} **catch**(Exception e) {

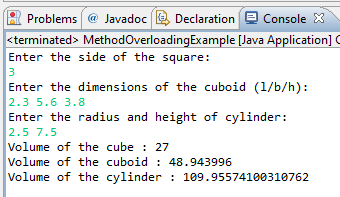
System.***out***.println(e);

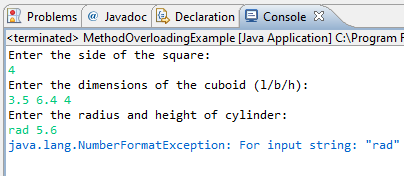
}

}

}

Output:





1. Java program to create a class Point with two data members x & y. Include all constructors and display().

**package** com.rsc.babystepprogram;

**import** java.util.\*;

**public** **class** Point {

**int** x, y;

Point() {

**this**.x = 0;

**this**.y = 0;

}

Point(**int** x) {

**this**.x = x;

**this**.y = 0;

}

Point(**int** x, **int** y) {

**this**.x = x;

**this**.y = y;

}

**public** **void** display() {

System.***out***.println("( "+**this**.x +", "+ **this**.y+" )");

}

**public** **static** **void** main(String[] args) {

Scanner scan = **new** Scanner(System.***in***);

Point p1 = **new** Point();

p1.display();

Point p2 = **new** Point(5);

p2.display();

Point p3 = **new** Point(3,6);

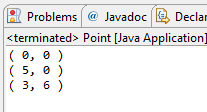
p3.display();

scan.close();

}

}

Output:



1. Java program using static method.

**package** com.rsc.babystepprogram;

**import** java.util.\*;

**class** Student {

**int** id;

String name;

**static** String *college* = "GIET";

Student(**int** id, String name) {

**this**.id = id;

**this**.name = name;

}

**static** String getCollege() {

**return** *college*;

}

}

**public** **class** StaticMethodExample {

**public** **static** **void** main(String[] args) {

Scanner scan = **new** Scanner(System.***in***);

System.***out***.println("Enter the pin and name of the student : ");

String input[] = scan.nextLine().split(" ");

**try** {

**int** id = Integer.*parseInt*(input[0]);

String name = input[1];

Student s1 = **new** Student(id, name);

System.***out***.println("Student id: "+ s1.id);

System.***out***.println("Student name: "+ s1.name);

System.***out***.println("College name: "+ Student.*getCollege*());

scan.close();

} **catch**( Exception e) {

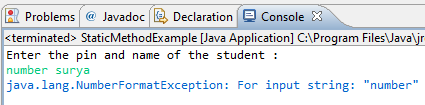
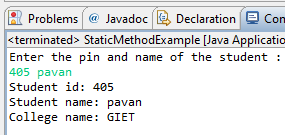
System.***out***.println(e);

}

}

}

Output:

1. What is conditional statement?
2. Conditional statements are used to check whether the given condition is satisfied or not.

These are also called Decision control statements. Java supports different conditional statements like

* Simple if
* If else
* Nested if else
* If else ladder

1. Write the syntax of switch case statement?
2. Syntax of switch statement:

switch(expression) {

case a: block-1

Break;

case b: block-2

Break;

.... ...

default: block-n

1. Write the difference between break and continue statements?
2. Break and continue statements come under conditional jump statements.
   1. Break statement is used to halt the current iteration and come out of the loop when any particular condition is met.
   2. Continue statement is used to skip the current iteration of the loop and go to the next iteration on a particular condition.
3. What is looping statements?
4. Looping statements are used to execute a block of statements over a finite number of iterations without writing them multiple times.

There are 3 types of looping statements

1. While b. Do-while c. For
2. Write the difference between while and do while statements?
3. While:
4. The statements in the while loop are executed if and only if the condition is true.
5. While loop is called entry controlled loop since the condition is checked prior to the execution.
6. Minimum iterations a while loop can run in zero, when condition is false.

Do-while:

1. The statements in the do while loop are executed for the first time and then the condition is checked for the next iteration.
2. Do while loop is called exit controlled loop since the condition is checked at the end of the loop.
3. Minimum number of times the do while loop can execute is 1.
4. What is array. How it is created?
5. An array is a non primitive data type which is used to store elements of similar data type. All the elements in the array are stored in the contiguous memory locations and are accessed using their index.

The syntax for creating an array is:

datatype array-name[] = new datatype [size];

ex: int array[] = new int[10];

above is an array of type integer which stores 10 integer elements.

1. What is class?
2. A class is virtual entity in programming world, which doesn’t exist in reality. In general a class in a blueprint, from which we can make multiple number of objects. It is a collection of data members(data) and member functions(code).
3. What is constructor?
4. A constructor is a special method of class which is invoked when an object of that class is instantiated. It is used to initialize the object. The name of the constructor must be same as the class name and it doesn’t have return type.
5. What is the use of copy constructor?
6. Copy constructor is used to create an object with the help of existing object of the same class. It is useful in case of copying a complex object to another without instantiating with all the fields of the data.
7. What is the use of this keyword?
8. a. The keyword this is used to refer to the current instance variable in java.
9. It is used to differentiate between the instance variable and arguments.
10. Used to call methods of same class implicitly.
11. What is method overloading?
12. Method overloading is a concept which allows a class to have one or more methods with a same name, provided that they are differentiated in the number of arguments passed in. We can overload a method in java by changing either the number of parameters or the data type of the parameters.
13. What is static variable?
14. Static is a keyword which is related to class. If a variable is declared static then it is said to be a static variable. It is the common property for all the objects of a class and can be initialized only once, keeping in view of memory management.
15. What is access modifier?
16. Access modifiers are used to specify the scope of the variable, method, object, constructor or class. There are 4 types of access modifiers in java.
17. Public
18. Private
19. Protected
20. Default
21. Write the difference between instance and static methods?
22. Instance methods:

* Instance methods are related to the objects of the class
* They are called with the help of the object
* Instance methods are different for different objects.

Static method:

* Static methods are related to the class.
* These methods does not require an object to invoke them, instead uses class name.
* Static methods are same for all the objects of a class.

1. What is object. How it is created?
2. An object is an instance or simply a copy of a class. Every object of a class will acquire all the attributes and behaviours of the class. Each object has its own space in the heap memory. The process of creating an object is called Instantiation.

Ex. ClassName ObjectName = new ClassName();

Here the new operator will create a new instance of a class which is called an object.