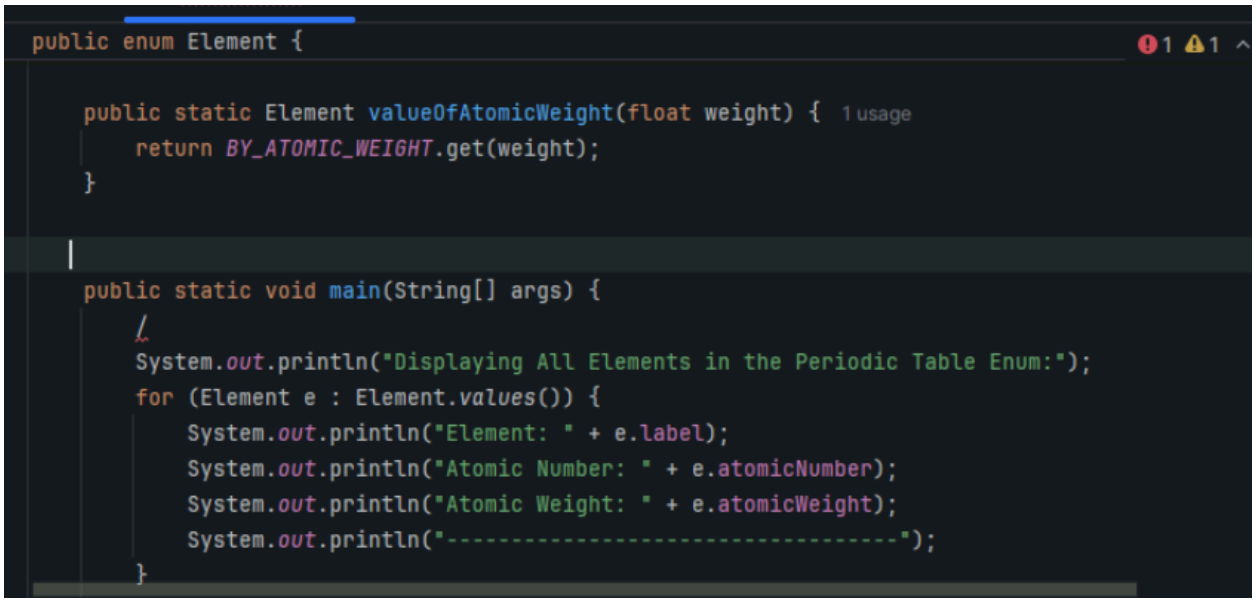
Enums

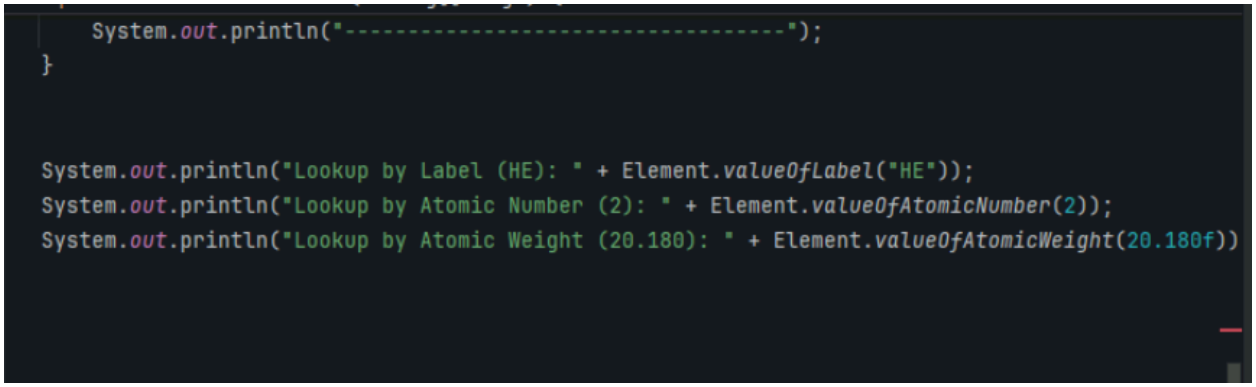
```
//Attaching Multiple values
public enum Element {
   H("Hydrogen", 1, 1.008f),
   HE("Helium", 2, 4.0026f),
   // ...
   NE("Neon", 10, 20.180f);

   private static final Map<String, Element> BY_LABEL = new HashMap<>();
   private static final Map<Integer, Element> BY_ATOMIC_NUMBER = new HashMap<>();
   private static final Map<Float, Element> BY_ATOMIC_WEIGHT = new HashMap<>();

   static {
      for (Element e : values()) {    //for each loop
         BY_LABEL.put(e.label, e);
         BY_ATOMIC_NUMBER.put(e.atomicNumber, e);
         BY_ATOMIC_WEIGHT.put(e.atomicWeight, e);
      }
   }

   public final String label;
   public final int atomicNumber;
   public final float atomicWeight;

   private Element(String label, int atomicNumber, float atomicWeight) {
      this.label = label;
      this.atomicNumber = atomicNumber;
      this.atomicWeight = atomicWeight;
   }

   public static Element valueOfLabel(String label) {
      return BY_LABEL.get(label);
   }

   public static Element valueOfAtomicNumber(int number) {
      return BY_ATOMIC_NUMBER.get(number);
   }

   public static Element valueOfAtomicWeight(float weight) {
      return BY_ATOMIC_WEIGHT.get(weight);
   }
}
```

Task 19:

Wap to display the content of the above enum.. (main needs to be added)

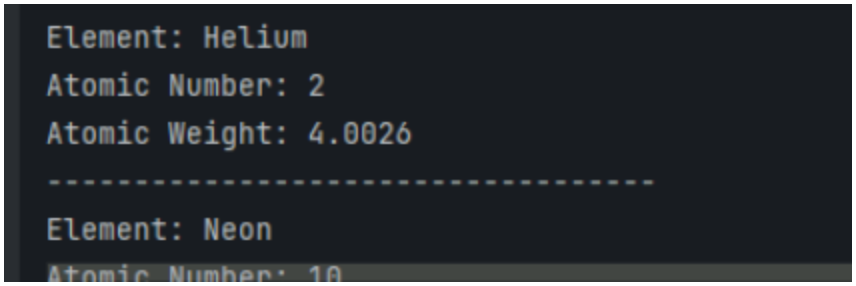


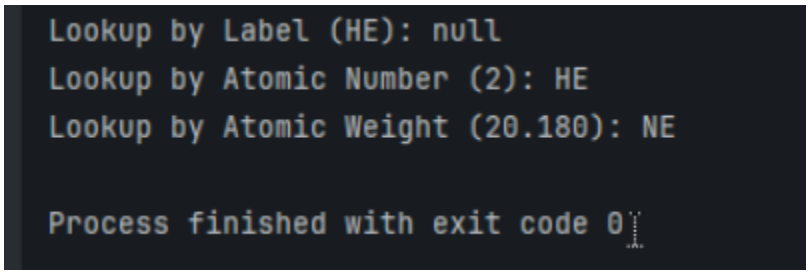
```
public enum Element {

    public static Element valueOfAtomicWeight(float weight) {  1 usage
        return BY_ATOMIC_WEIGHT.get(weight);
    }


    public static void main(String[] args) {
        /
        System.out.println("Displaying All Elements in the Periodic Table Enum:");
        for (Element e : Element.values()) {
            System.out.println("Element: " + e.label);
            System.out.println("Atomic Number: " + e.atomicNumber);
            System.out.println("Atomic Weight: " + e.atomicWeight);
            System.out.println("----------------------------------");
        }
```



```
    System.out.println("----------------------------------");
}


System.out.println("Lookup by Label (HE): " + Element.valueOfLabel("HE"));
System.out.println("Lookup by Atomic Number (2): " + Element.valueOfAtomicNumber(2));
System.out.println("Lookup by Atomic Weight (20.180): " + Element.valueOfAtomicWeight(20.180f))
```

```
External Libraries
Scratches and Consoles
    }
}

Run  Element
"C:\Program Files\Java\jdk-17\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2025.1\lib\idea_rt.jar=49227" -Dfile.encod
Displaying All Elements in the Periodic Table Enum:
Element: Hydrogen
Atomic Number: 1
Atomic Weight: 1.008
----------------------------------
Element: Helium
```





10.06 to 10.10

Task 020:

Create an array of your name

Hint : use

Char[] Name = {‘P’, “r’, ….}; // initializing an array

sout(Name);

Int n = Name.length; // size of your name

sout(“there are “+ n +”letters in my name”);

Use for loop to display each letter..

HInt: use ghe below code snippet…
// traversing array

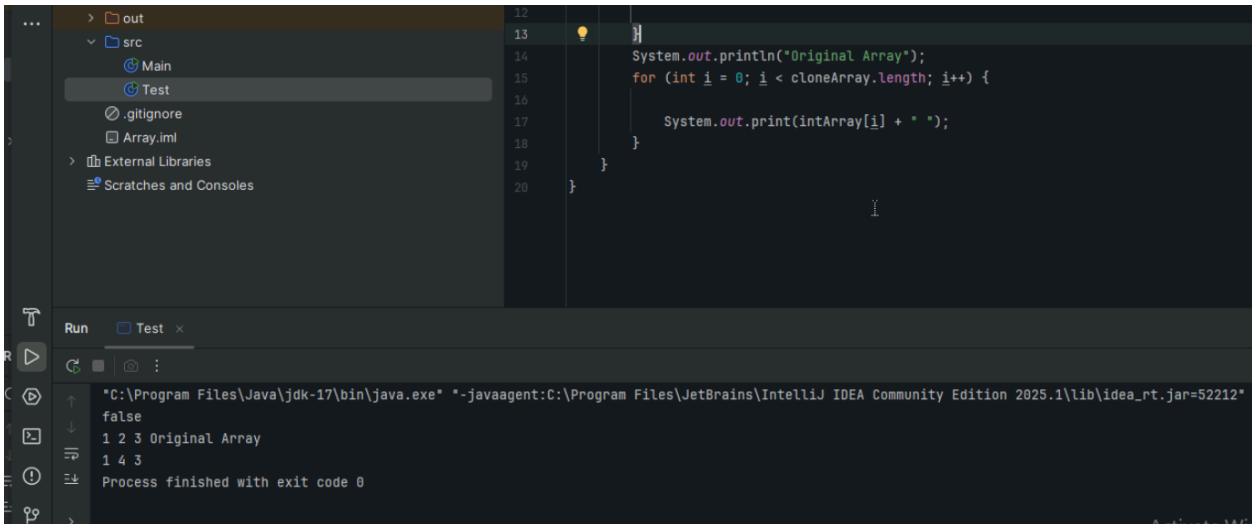
```
for (int i = 0; i < n; i++)
    System.out.print(Name[i] + " ");
```

```java
public class Task020 {
    public static void main(String[] args) {
        // Step 1: Initialize an array with your name (replace with your own name)
        char[] Name = {'M', 'e', 'e', 'r', 'a'};

        // Step 2: Print the array directly
        System.out.print("My name is: ");
        System.out.println(Name); // This prints the entire char array as a string

        // Step 3: Find the length of the array
        int n = Name.length;

        // Step 4: Print the number of letters in the name
        System.out.println("There are " + n + " letters in my name");

        // Step 5: Use for loop to display each letter with a space
        System.out.print("Letters in my name: ");
        for (int i = 0; i < n; i++) {
```

Task 29 home task

shallow copy vs deep copy

```java
class Test {
    public static void main(String args[])
    {
        int intArray[] = { 1, 2, 3 };
        int cloneArray[] = intArray.clone();
        intArray[1]= 4;
        // will print false as shallow copy is created
        System.out.println(intArray == cloneArray);
        for (int i = 0; i < cloneArray.length; i++) {
            System.out.print(cloneArray[i] + " ");


        }
        System.out.println("Original Array");
        for (int i = 0; i < cloneArray.length; i++) {
```

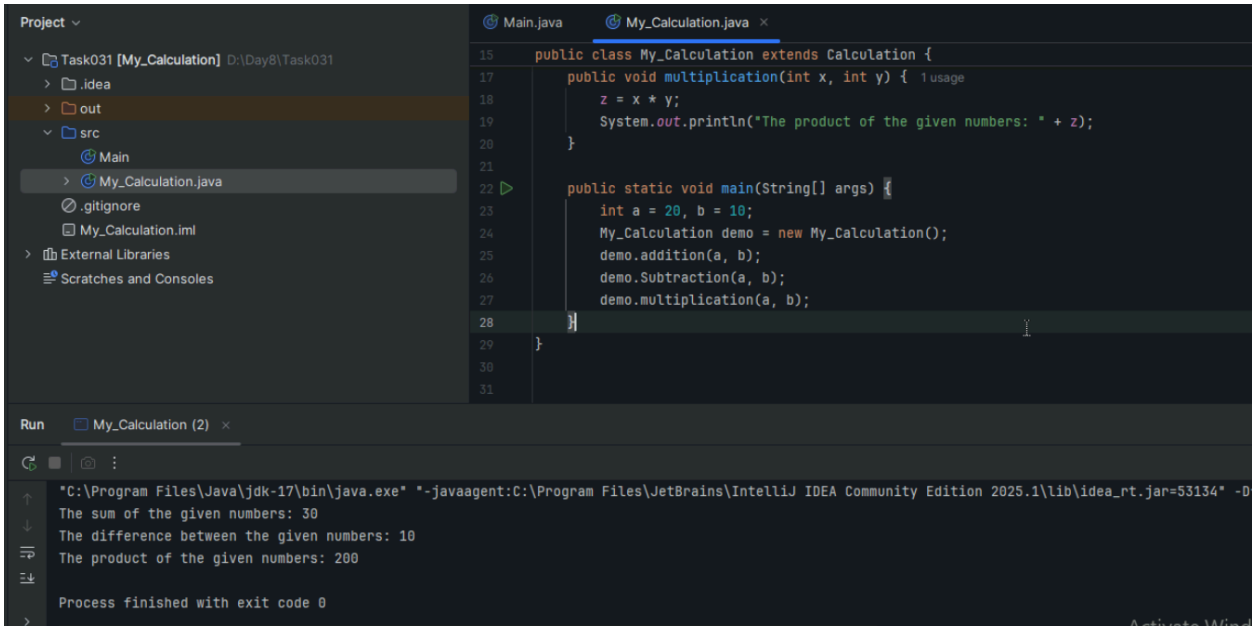


Task 031

```
class Calculation {
  int z;

  public void addition(int x, int y) {
    z = x + y;
    System.out.println("The sum of the given numbers:"+z);
  }
```

```
  public void Subtraction(int x, int y) {
    z = x - y;
    System.out.println("The difference between the given numbers:"+z);
  }
}

public class My_Calculation extends Calculation {
  public void multiplication(int x, int y) {
    z = x * y;
    System.out.println("The product of the given numbers:"+z);
  }

  public static void main(String args[]) {
    int a = 20, b = 10;
    My_Calculation demo = new My_Calculation();
    demo.addition(a, b);
    demo.Subtraction(a, b);
    demo.multiplication(a, b);
  }
}
```

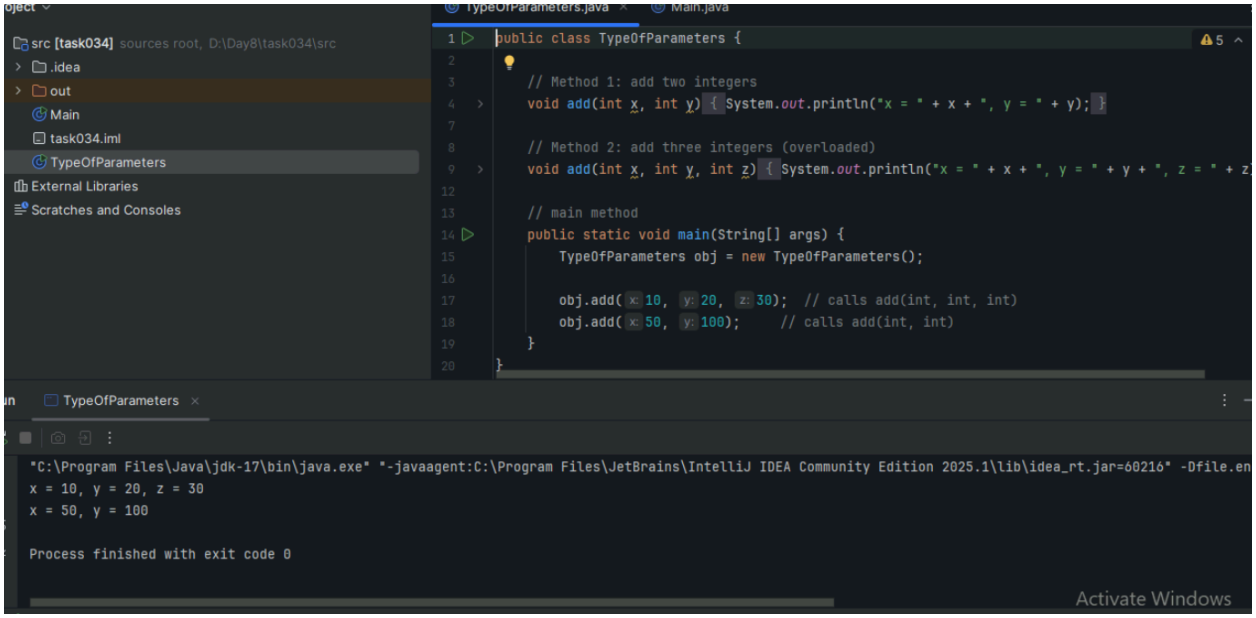
```
class Calculation {  1 usage  1 inheritor
    int z;  6 usages

    public void addition(int x, int y) {  1 usage
        z = x + y;
        System.out.println("The sum of the given numbers: " + z);
    }

    public void Subtraction(int x, int y) {  1 usage
        z = x - y;
        System.out.println("The difference between the given numbers: " + z);
    }
}

public class My_Calculation extends Calculation {
```



Task 034

```
Void add(int x, int y){
        Sout —> x and y values
}
Void add(int x, int y, int z){
        Sout    —-> x, y, z values

}
psvm(){
        add(10,20,30);
        add(50,100);
}
```

Type of parameters



Task 035

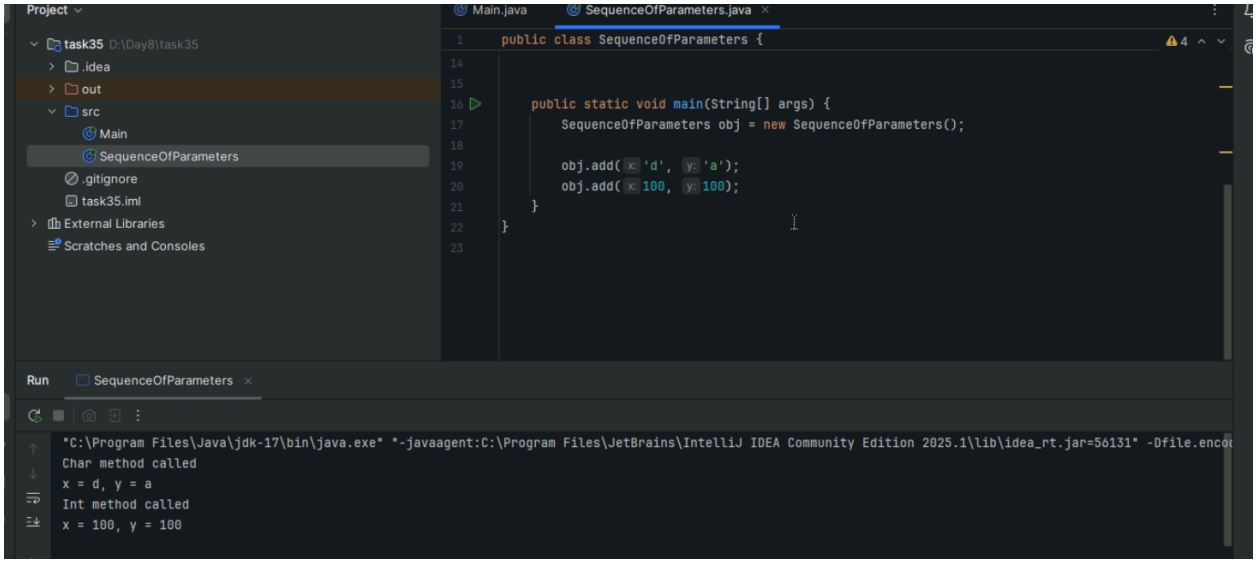
Void add(char x, char y){
Sout —-> x, y values
}

Void add(int x, int y) {
Sout —> x, y values
}

psvm(){
add('d', 'a');
add(100, 100);
}

Sequence of Parameters

```java
public class SequenceOfParameters {


    void add(char x, char y) {  1 usage
        System.out.println("Char method called");
        System.out.println("x = " + x + ", y = " + y);
    }


    void add(int x, int y) {  1 usage
        System.out.println("Int method called");
        System.out.println("x = " + x + ", y = " + y);
    }


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

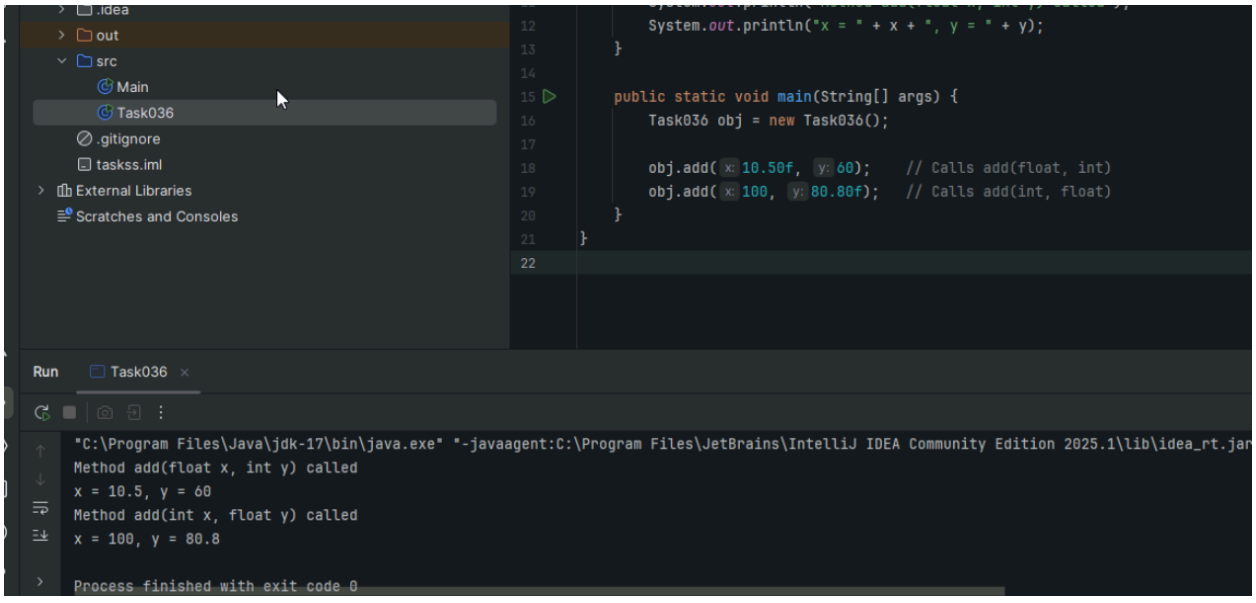


Task 036

Void add(int x, float y){
    Sout → x, y values
}
Void add(float x, int y){
    Sout → x, y
}

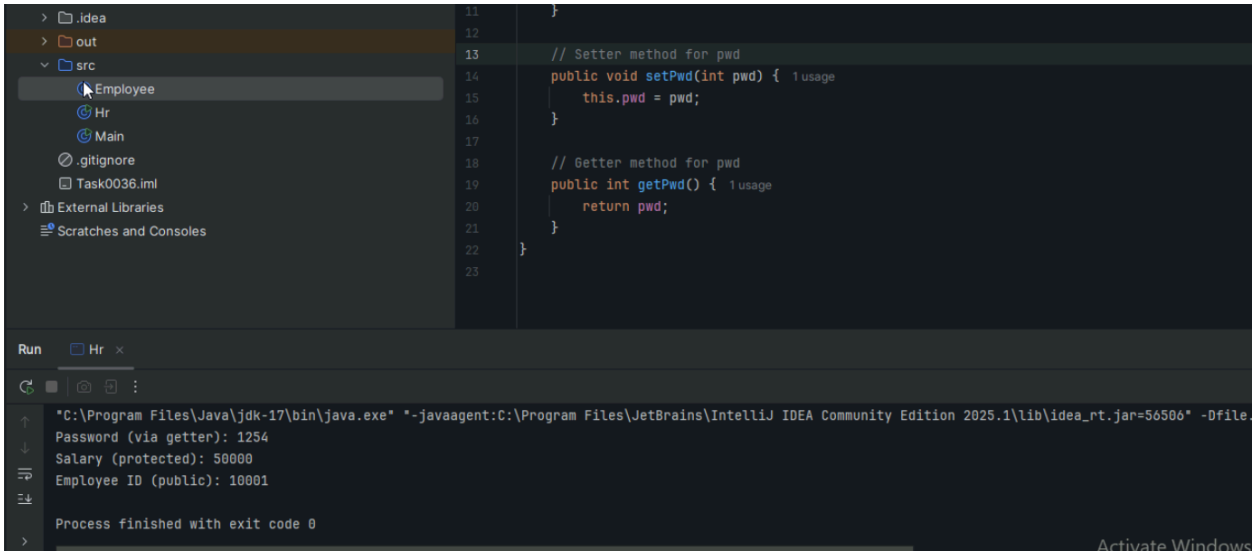
psvm(){
	add(10.50f, 60);
	add(100, 80.80f)
}

```java
public class Task036 {

    // Method with int first, then float
    void add(int x, float y) {  1 usage
        System.out.println("Method add(int x, float y) called");
        System.out.println("x = " + x + ", y = " + y);
    }

    // Method with float first, then int
    void add(float x, int y) {  1 usage
        System.out.println("Method add(float x, int y) called");
        System.out.println("x = " + x + ", y = " + y);
    }

    public static void main(String[] args) {
        Task036 obj = new Task036();
```



Task 037:

```
Class Employee{
        Private int pwd;
        Protected int Salary;
        Public int empid:

        employee(){ // constructors are methods having same name as class name  (we have in
c++)
        }
        ~employee(){// destructors used in c++ but not in java
        }

}
Class Hr extends Employee {
        super.pwd = 1254; //================>  ??????
        super.Salary = 50000; //==================>  ?
        Super.empid = 10001; // ======================>?
        psvm(){
        }
}
```



Task 038

/* File name : AbstractDemo.java */
Public class AbstractDemo {

public static void main(String [] args) {

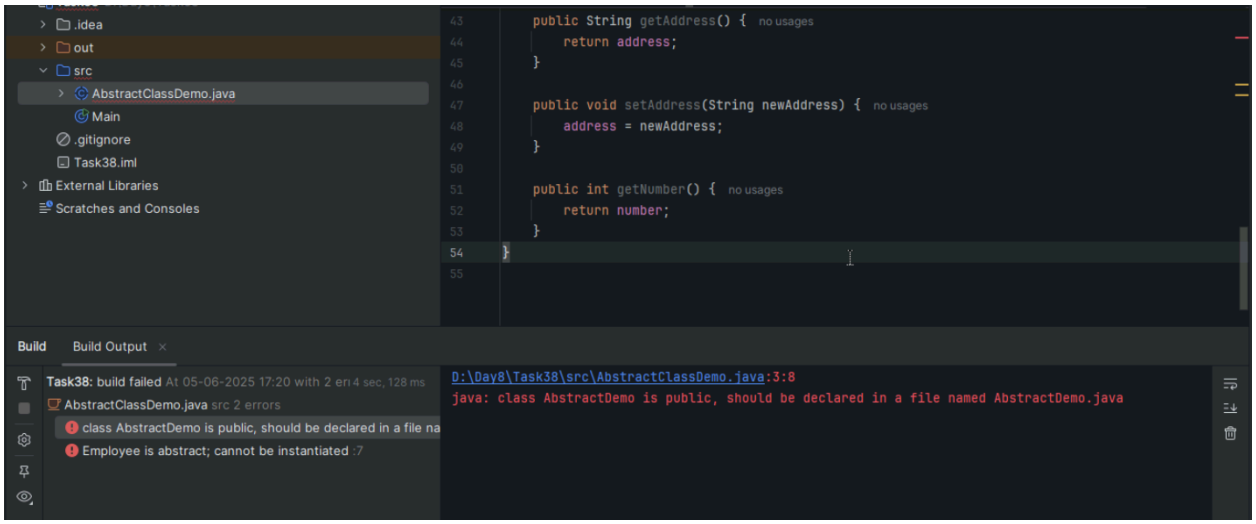
```
    /* Following is not allowed and would raise error */
    Employee e = new Employee("George W.", "Houston, TX", 43);
    System.out.println("\n Call mailCheck using Employee reference--");
    e.mailCheck();
  }
}

abstract class Employee {
  private String name;
  private String address;
  private int number;

  public Employee(String name, String address, int number) {
    System.out.println("Constructing an Employee");
    this.name = name;
    this.address = address;
    this.number = number;
  }

  public double computePay() {
   System.out.println("Inside Employee computePay");
   return 0.0;
  }

  public void mailCheck() {
    System.out.println("Mailing a check to " + this.name + " " + this.address);
  }

  public String toString() {
    return name + " " + address + " " + number;
  }

  public String getName() {
    return name;
  }

  public String getAddress() {
    return address;
  }

  public void setAddress(String newAddress) {
    address = newAddress;
  }
```

```
  public int getNumber() {
    return number;
  }
}
```

```
public class AbstractDemo {

    public static void main(String[] args) {

        Employee e = new Employee( name: "George W.", address: "Houston, TX", number: 43);

        System.out.println("\nCall mailCheck using Employee reference--");
        e.mailCheck();
    }
}

abstract class Employee {  2 usages
    private String name;  4 usages
    private String address;  5 usages
```

```
abstract class Employee {  2 usages
    private int number;  3 usages

    public Employee(String name, String address, int number) {  1 usage
        System.out.println("Constructing an Employee");
        this.name = name;
        this.address = address;
        this.number = number;
    }

    public double computePay() {  no usages
        System.out.println("Inside Employee computePay");
        return 0.0;
    }

    public void mailCheck() {  1 usage
```

```java
public void mailCheck() {  1 usage
    System.out.println("Mailing a check to " + this.name + ", " + this.address);
}

public String toString() {
    return name + " " + address + " " + number;
}

public String getName() {  no usages
    return name;
}

public String getAddress() {  no usages
```



Task 039

Rewrite the above code to give the output without errors

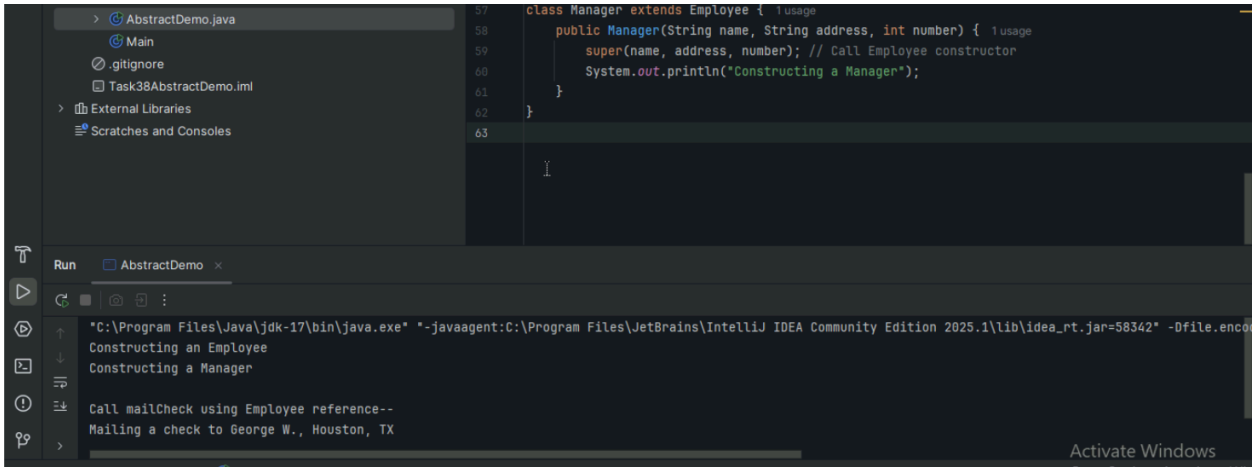
```java
// File: AbstractDemo.java

public class AbstractDemo {
    public static void main(String[] args) {
        // ✅ We are creating an object of Manager (not abstract)
        Employee e = new Manager( name: "George W.", address: "Houston, TX", number: 43);

        System.out.println("\nCall mailCheck using Employee reference--");
        e.mailCheck();
    }
}

// ✅ Abstract base class
abstract class Employee {  2 usages  1 inheritor
    private String name;  4 usages
    private String address;  5 usages
```

```java
abstract class Employee {  2 usages  1 inheritor
    private String name;  4 usages
    private String address;  5 usages
    private int number;  3 usages

    public Employee(String name, String address, int number) {  1 usage
        System.out.println("Constructing an Employee");
        this.name = name;
        this.address = address;
        this.number = number;
    }

    public double computePay() {  no usages
        System.out.println("Inside Employee computePay");
        return 0.0;
```

```
    return 0.0;
}

public void mailCheck() {  1 usage
    System.out.println("Mailing a check to " + this.name + ", " + this.address);
}

public String toString() {
    return name + " " + address + " " + number;
}

public String getName() {  no usages
    return name;
```

```
    public String getName() {  no usages
        return name;
    }

    public String getAddress() {  no usages
        return address;
    }

    public void setAddress(String newAddress) {  no usages
        address = newAddress;
    }

    public int getNumber() {  no usages
        return number;
```

```
abstract class Employee {  2 usages  1 inheritor
        return address;
    }

    public void setAddress(String newAddress) {  no usages
        address = newAddress;
    }

    public int getNumber() {  no usages
        return number;
    }
}

// ✅ Concrete subclass that extends Employee
class Manager extends Employee {  1 usage
    public Manager(String name, String address, int number) {  1 usage
```



Task 040

```
// Working of Abstraction in Java
abstract class Gadgets {
    abstract void turnOn();
    abstract void turnOff();
}
// Concrete class implementing the abstract methods
class TVRemote extends Gadgets {
    @Override
```

```
    void turnOn() {
        System.out.println("TV is turned ON.");
    }

    @Override
    void turnOff() {
        System.out.println("TV is turned OFF.");
    }
}
class ACRemote extends Gadgets {
    @Override
    void turnOn() {
        System.out.println("AC is turned ON.");
    }

    @Override
    void turnOff() {
        System.out.println("AC is turned OFF.");
    }
}

// Main class to demonstrate abstraction
public class Main {
    public static void main(String[] args) {
        Gadgets remote = new TVRemote();
        Gadgets remote = new ACRemote();
       remote.turnOn();
        remote.turnOff();

        Gadgets remote = new FanRemote();
        Gadgets remote = new CoolerRemote();
        remote.turnOn();
        remote.turnOff();
    }
}
```

```java
// Abstract base class
abstract class Gadgets {  5 usages
    abstract void turnOn();  4 usages  4 implementations
    abstract void turnOff();  4 usages  4 implementations
}

// Concrete subclass for TVRemote
class TVRemote extends Gadgets {  1 usage
    @Override  4 usages
    void turnOn() {
        System.out.println("TV is turned ON.");
    }

    @Override  4 usages
    void turnOff() {
        System.out.println("TV is turned OFF.");
```

```java
        System.out.println("TV is turned OFF.");
    }
```

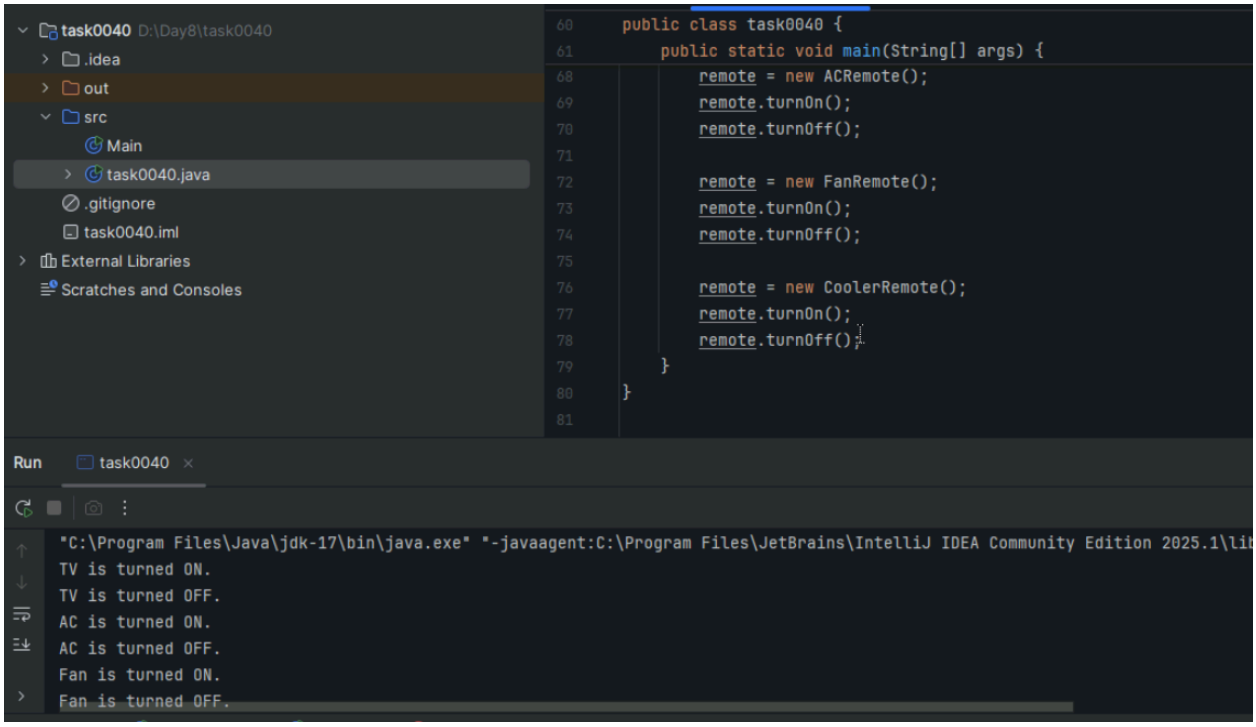
```java
void turnOff() {
    System.out.println("AC is turned OFF.");
}


Concrete subclass for FanRemote
ss FanRemote extends Gadgets {  1 usage
 @Override  4 usages
 void turnOn() {
    System.out.println("Fan is turned ON.");
```

```java
// Concrete subclass for FanRemote
class FanRemote extends Gadgets {  1 usage
    @Override  4 usages
    void turnOn() {
        System.out.println("Fan is turned ON.");
    }

    @Override  4 usages
    void turnOff() {
        System.out.println("Fan is turned OFF.");
    }
}

// Concrete subclass for CoolerRemote
class CoolerRemote extends Gadgets {  1 usage
```

```java
class CoolerRemote extends Gadgets {  1 usage
    @Override  4 usages
    void turnOn() {
        System.out.println("Cooler is turned ON.");
    }

    @Override  4 usages
    void turnOff() {
        System.out.println("Cooler is turned OFF.");
    }
}

// Main public class named task0040
public class task0040 {
    public static void main(String[] args) {
```

```java
public class task0040 {
    public static void main(String[] args) {
        Gadgets remote;

        remote = new TVRemote();
        remote.turnOn();
        remote.turnOff();

        remote = new ACRemote();
        remote.turnOn();
        remote.turnOff();

        remote = new FanRemote();
        remote.turnOn();
```

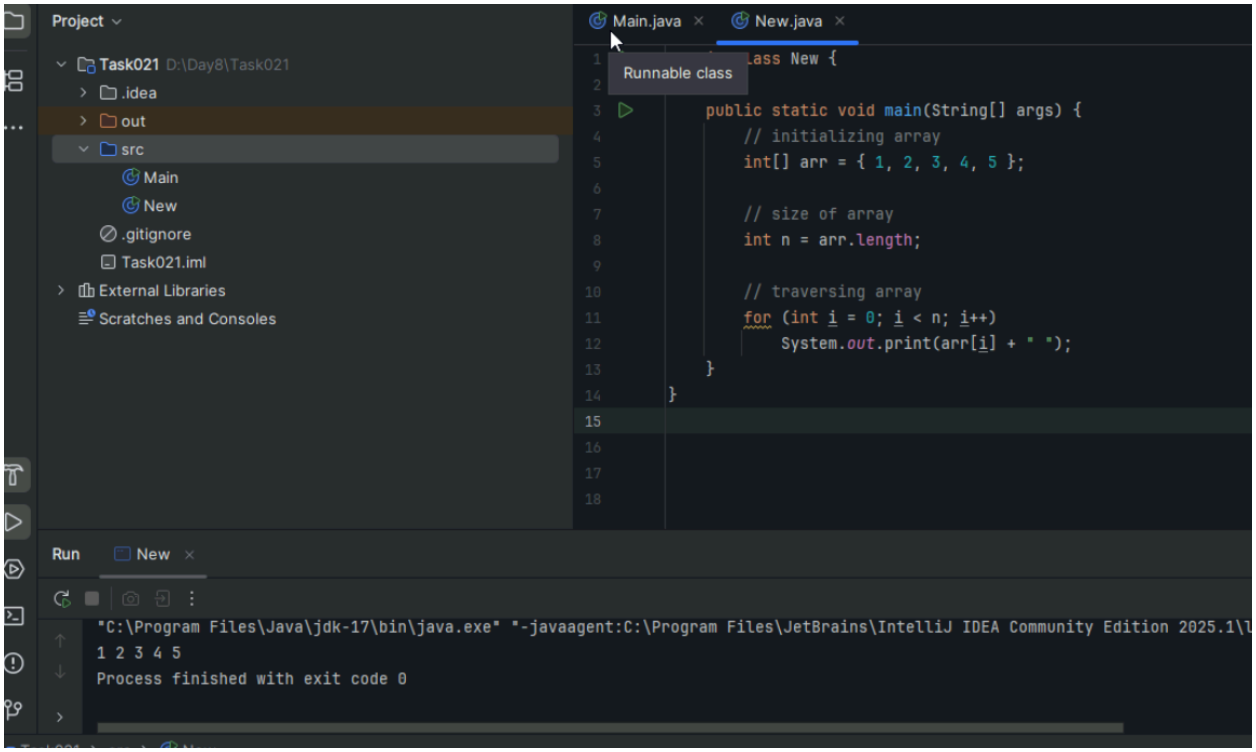
```java
    public static void main(String[] args) {
        remote.turnOn();
        remote.turnOff();

        remote = new ACRemote();
        remote.turnOn();
        remote.turnOff();

        remote = new FanRemote();
        remote.turnOn();
        remote.turnOff();

        remote = new CoolerRemote();
        remote.turnOn();
        remote.turnOff();
```



Task 21: Home Task

Example: This example demonstrates how to initialize an array and traverse it using a for loop to print each element.

```
public class Main {
    public static void main(String[] args)
    {

        // initializing array
        int[] arr = { 1, 2, 3, 4, 5 };

        // size of array
        int n = arr.length;

        // traversing array
        for (int i = 0; i < n; i++)
            System.out.print(arr[i] + " ");
    }
```



Task 022 - home task

Implementation:

```
// Java program to illustrate creating an array
// of integers,  puts some values in the array,
// and prints each value to standard output.

class GFG {
    public static void main(String[] args)
    {
        // declares an Array of integers.
        int[] arr;

        // allocating memory for 5 integers.
        arr = new int[5];
```

```
        // initialize the elements of the array
        // first to last(fifth) element
       arr[0] = 10;
       arr[1] = 20;
        arr[2] = 30;
        arr[3] = 40;
        arr[4] = 50;

        // accessing the elements of the specified array
        for (int i = 0; i < arr.length; i++)
            System.out.println("Element at index "
                               + i + " : " + arr[i]);
    }
}
```

```
class task022 {
    public static void main(String[] args) {
        // declares an Array of integers.
        int[] arr;

        // allocating memory for 5 integers.
        arr = new int[5];

        // initialize the elements of the array
        // first to last(fifth) element
        arr[0] = 10;
        arr[1] = 20;
        arr[2] = 30;
        arr[3] = 40;
        arr[4] = 50;
```

Task 023 - home task

Example: Here we are taking a student class and creating an array of Student with five Student objects stored in the array. The Student objects have to be instantiated using the constructor of the Student class, and their references should be assigned to the array elements.

```
// Java program to illustrate creating
//  an array of objects

class Student {
   public int roll_no;
   public String name;
    Student(int roll_no, String name){
       this.roll_no = roll_no;
       this.name = name;
   }
}

public class Main {
   public static void main(String[] args){

       // declares an Array of Student
```

```
        Student[] arr;

        // allocating memory for 5 objects of type Student.
        arr = new Student[5];

        // initialize the elements of the array
        arr[0] = new Student(1, "aman");
        arr[1] = new Student(2, "vaibhav");
        arr[2] = new Student(3, "shikar");
        arr[3] = new Student(4, "dharmesh");
        arr[4] = new Student(5, "mohit");

        // accessing the elements of the specified array
        for (int i = 0; i < arr.length; i++)
            System.out.println("Element at " + i + " : { "
                               + arr[i].roll_no + " "
                               + arr[i].name+" }");
    }
}
```

```java
class Student {  7 usages
    public int roll_no;  2 usages
    public String name;  2 usages

    public Student(int roll_no, String name) {  5 usages
        this.roll_no = roll_no;
        this.name = name;
    }
}

public class Mainss {
    public static void main(String[] args) {
        Student[] students = new Student[5];

        students[0] = new Student( roll_no: 1,  name: "Aman");
        students[1] = new Student( roll_no: 2,  name: "Vaibhav");
```

```java
public class Mainss {
    public static void main(String[] args) {
        Student[] students = new Student[5];

        students[0] = new Student( roll_no: 1,  name: "Aman");
        students[1] = new Student( roll_no: 2,  name: "Vaibhav");
        students[2] = new Student( roll_no: 3,  name: "Shikar");
        students[3] = new Student( roll_no: 4,  name: "Dharmesh");
        students[4] = new Student( roll_no: 5,  name: "Mohit");

        for (int i = 0; i < students.length; i++) {
            System.out.println("Element at " + i + " : { "
                    + students[i].roll_no + " " + students[i].name + " }");
        }
    }
```

`Task 024 Home task`

Example: An array of objects is also created like

```
// Java program to illustrate creating
//  an array of objects
 class Student{
   public String name;
    Student(String name){
       this.name = name;
   }

  @Override
   public String toString(){
       return name;
   }
}

public class Main{
   public static void main (String[] args){

       // declares an Array and initializing the
      // elements of the array
       Student[] myStudents = new Student[]{
         new Student("Dharma"),new Student("sanvi"),
         new Student("Rupa"),new Student("Ajay")
       };
```

```
         // accessing the elements of the specified array
        for(Student m:myStudents){
            System.out.println(m);
        }
    }
}
```

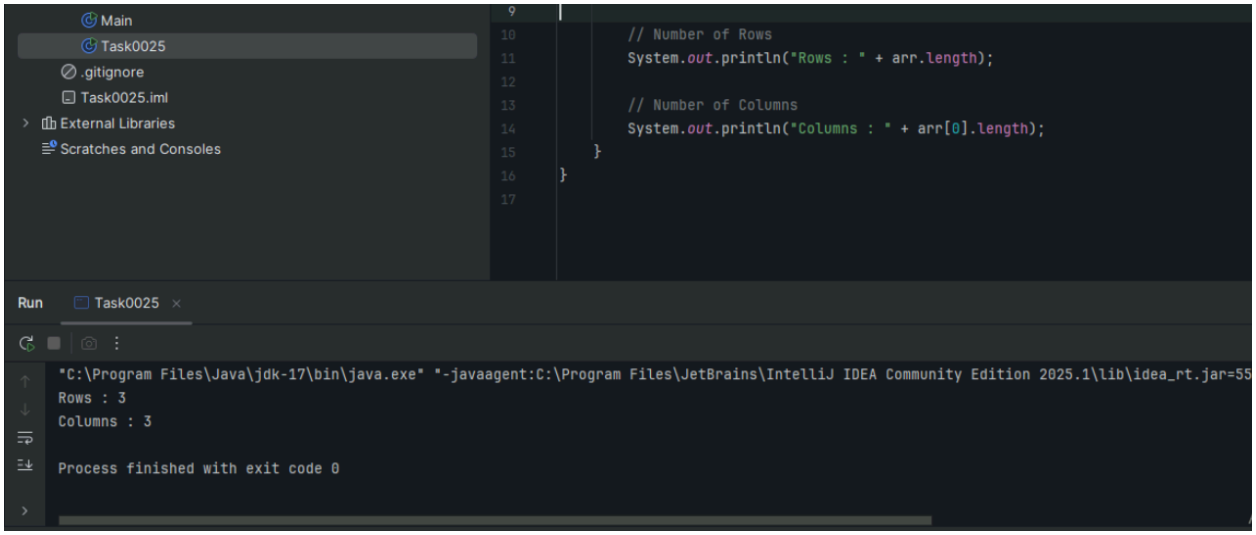
Output

```
Dharma
sanvi
Rupa
Ajay
```

Task 025 - home Task

```
// Code for showing error "ArrayIndexOutOfBoundsException"

public class GFG {
    public static void main(String[] args)
    {
        int[] arr = new int[4];
        arr[0] = 10;
        arr[1] = 20;
        arr[2] = 30;
        arr[3] = 40;

        System.out.println(
            "Trying to access element outside the size of array");
        System.out.println(arr[5]);
    }
}
```

```java
class Task0025{
    public static void main(String[] args){

        // Two Dimensional Array
        // Declared and Initialized
        int[][] arr = new int[3][3];


        // Number of Rows
        System.out.println("Rows : " + arr.length);

        // Number of Columns
        System.out.println("Columns : " + arr[0].length);
    }
}
```



`Task 026 - Home Task`

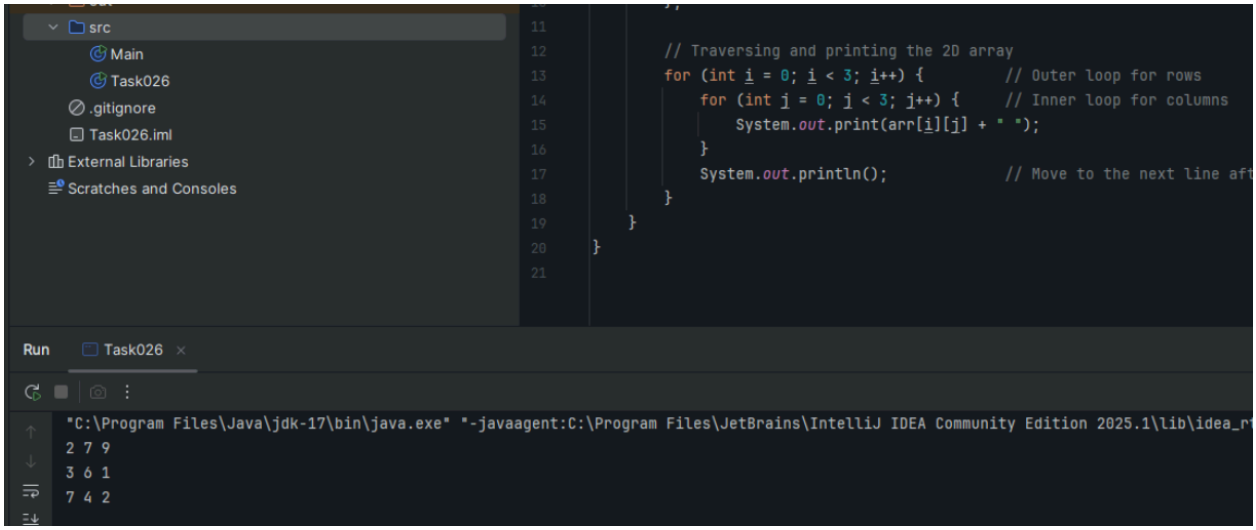
Example: Now, after declaring and initializing the array we will check how to Traverse the Multidimensional Array using for loop.

```java
// Java Program to Multidimensional Array

// Driver Class
public class multiDimensional {
     // main function
```

```
    public static void main(String args[])
    {
        // declaring and initializing 2D array
        int arr[][] = { { 2, 7, 9 }, { 3, 6, 1 }, { 7, 4, 2 } };

        // printing 2D array
        for (int i = 0; i < 3; i++) {
            for (int j = 0; j < 3; j++)
                System.out.print(arr[i][j] + " ");

            System.out.println();
        }
    }
}
```

Output

```
public class Task026 {
    public static void main(String[] args) {
        // Declaring and initializing a 2D array (3x3)
        int[][] arr = {
                { 2, 7, 9 },
                { 3, 6, 1 },
                { 7, 4, 2 }
        };

        // Traversing and printing the 2D array
        for (int i = 0; i < 3; i++) {          // Outer loop for rows
            for (int j = 0; j < 3; j++) {      // Inner loop for columns
                System.out.print(arr[i][j] + " ");
            }
```

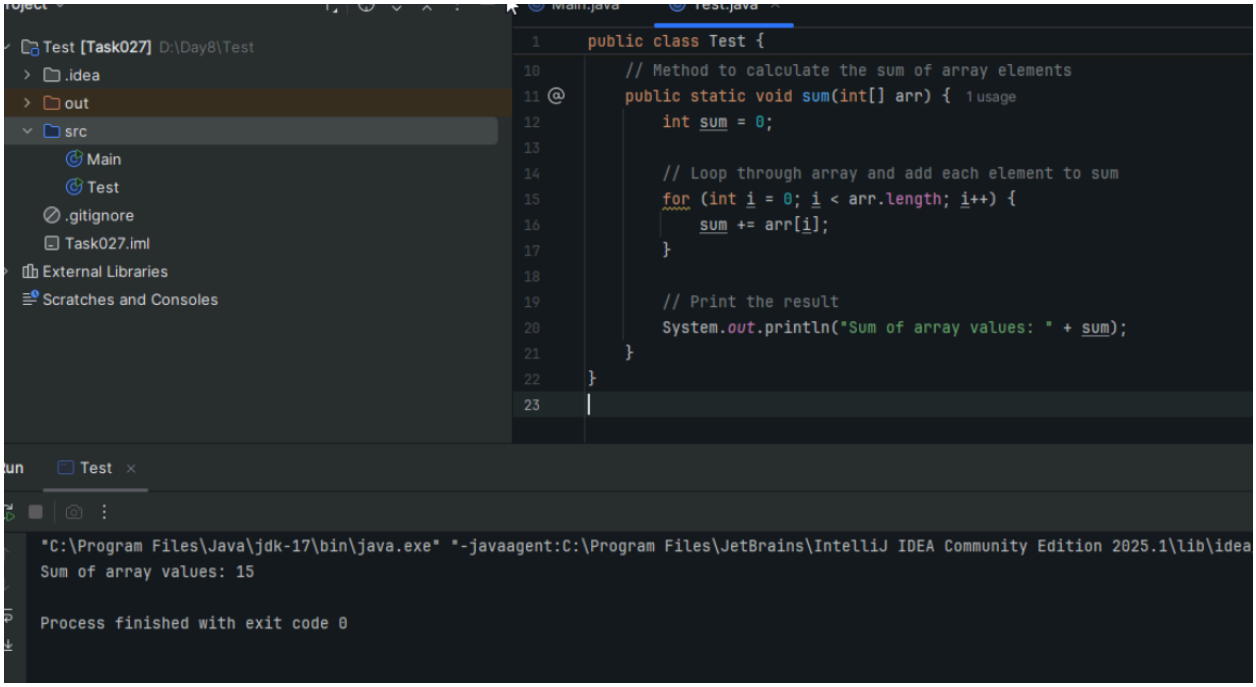


Task 27 - Home task

```
// Java program to demonstrate
// passing of array to method

public class Test {
    // Driver method
    public static void main(String args[])
    {
        int arr[] = { 3, 1, 2, 5, 4 };

        // passing array to method m1
        sum(arr);
    }

    public static void sum(int[] arr)
    {
        // getting sum of array values
        int sum = 0;

        for (int i = 0; i < arr.length; i++)
            sum += arr[i];

        System.out.println("sum of array values : " + sum);
    }
}
```

Output

```
sum of array values : 15
```



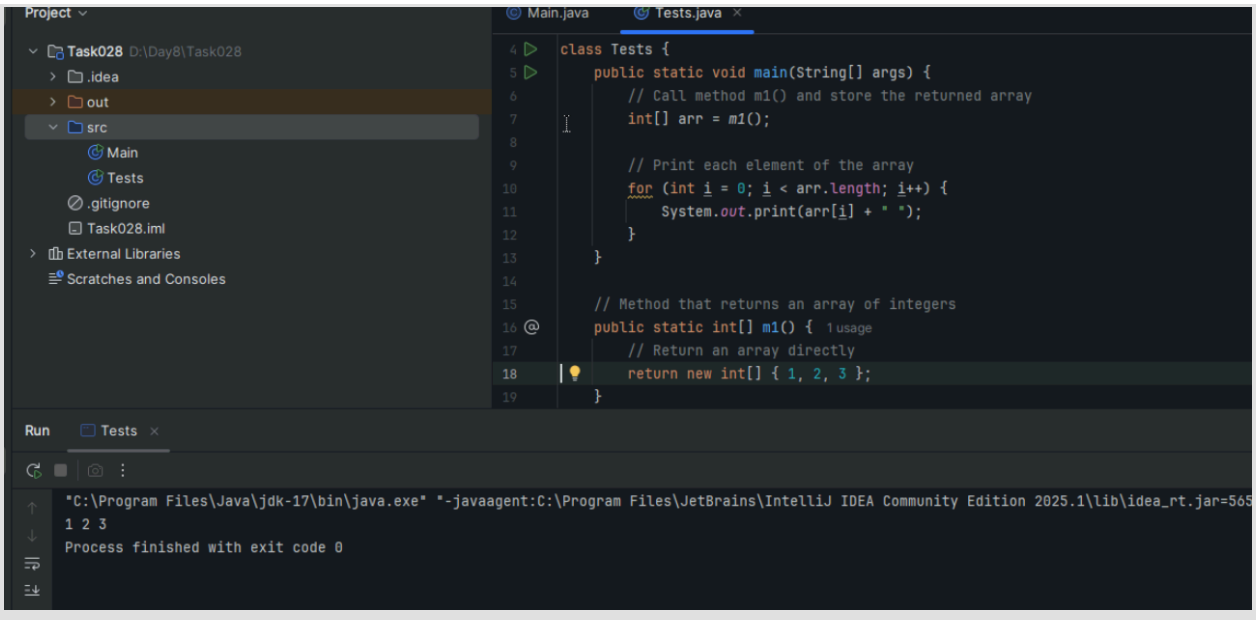
Task 28 - Home Task

```
// Java program to demonstrate
// return of array from method

class Test {
    // Driver method
    public static void main(String args[])
    {
        int arr[] = m1();

        for (int i = 0; i < arr.length; i++)
            System.out.print(arr[i] + " ");
    }

    public static int[] m1()
    {
```

```
        // returning  array
        return new int[] { 1, 2, 3 };
    }
}
```

Output

```
1 2 3
```

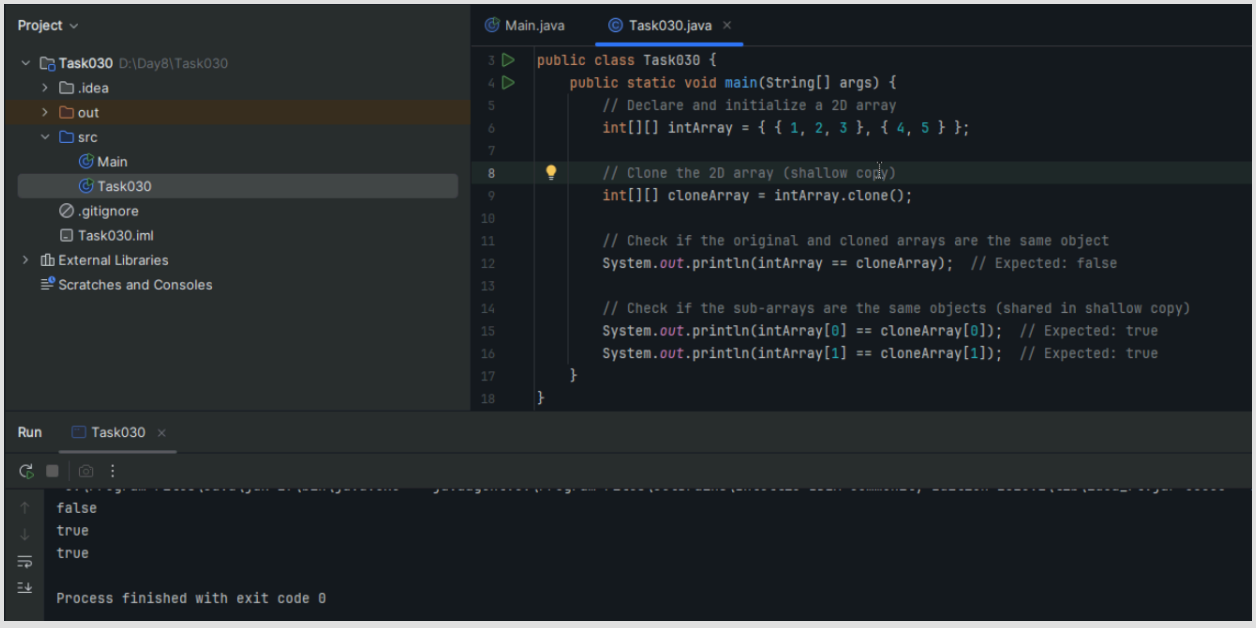



```
Task 030 Home Task
// Java program to demonstrate
// cloning of multi-dimensional arrays

class Test {
   public static void main(String args[])
   {
        int intArray[][] = { { 1, 2, 3 }, { 4, 5 } };

        int cloneArray[][] = intArray.clone();
```

```
        // will print false
        System.out.println(intArray == cloneArray);

        // will print true as shallow copy is created
        // i.e. sub-arrays are shared
        System.out.println(intArray[0] == cloneArray[0]);
        System.out.println(intArray[1] == cloneArray[1]);
    }
}
```

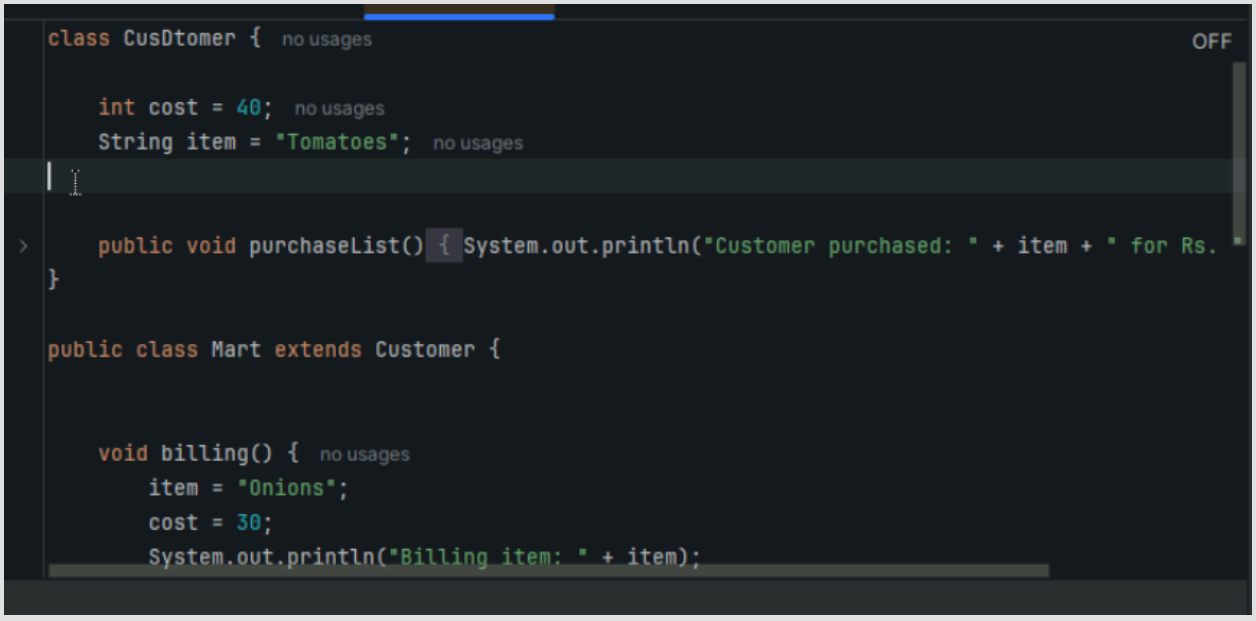
Output

```
false
true
true
```

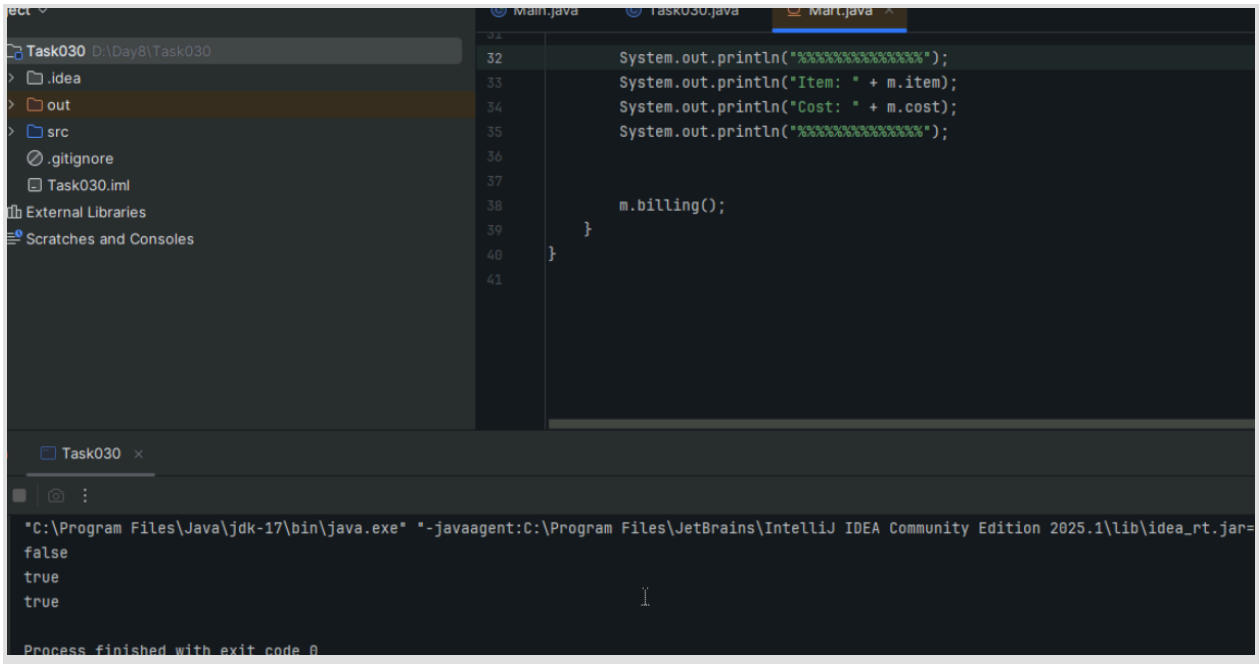



```
class Customer {

        Void purchage_list{
                Int cos = 40t;
                String items = “Tomatoes”;
        }
}
```

```
public class Mart extends Customer {
        Void billing(){
                String items = “onions”;
                Int cost = 30;
        }
        Psvm   (String[] args) {
                Super.items = “Potatoes”
                Super.cost = 50;

                Sout(items);
                sout(cost);
                sout”(%%%%%%%%%%%%%%”);
                Sout(super.items);
                sout(suer.cost);
        }
}
```




Main.java
Task030.java
Mart.java
System.out.println("Billing item: " + item);
System.out.println("Billing cost: " + cost);
}
public static void main(String[] args) {
Mart m = new Mart();
m.item = "Potatoes";
m.cost = 50;
System.out.println("%%%%%%%%%%%%%%%");
System.out.println("Item: " + m.item);


Task030 D:\Day8\Task030
.idea
out
src
.gitignore
Task030.iml
External Libraries
Scratches and Consoles
System.out.println("%%%%%%%%%%%%%%%");
System.out.println("Item: " + m.item);
System.out.println("Cost: " + m.cost);
System.out.println("%%%%%%%%%%%%%%%");
m.billing();
}
}
Task030
"C:\Program Files\Java\jdk-17\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2025.1\lib\idea_rt.jar=
false
true
true
Process finished with exit code 0