

## 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)

The screenshot shows an IDE interface with a code editor and a terminal window.

**Code Editor Content:**

```
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))  
}
```

**Terminal Output:**

```
> External Libraries  
≡ Scratches and Consoles  
Run Element x  
> ↑ "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.encoding=UTF-8  
↓ Displaying All Elements in the Periodic Table Enum:  
Element: Hydrogen  
Atomic Number: 1  
Atomic Weight: 1.008  
-----  
> Element: Helium
```

```
Element: Helium
Atomic Number: 2
Atomic Weight: 4.0026
-----
Element: Neon
Atomic Number: 10
-----
Atomic Number: 10
Atomic Weight: 20.18
-----
Lookup by Label (HE): null
Lookup by Atomic Number (2): HE
Lookup by Atomic Weight (20.180): NE
-----
Lookup by Label (HE): null
Lookup by Atomic Number (2): HE
Lookup by Atomic Weight (20.180): NE
-----
Process finished with exit code 0
```

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] + " ");
```

The screenshot shows a Java code editor with a dark theme. The code is part of a class named Task020. It initializes a character array with the letters 'M', 'e', 'e', 'r', and 'a'. It then prints the entire array as a string, finds its length, prints the number of letters, and finally uses a for loop to print each letter with a space between them. The code editor also shows a file structure on the left with 'src' containing 'Main', 'Task020', '.gitignore', and 'Task020.iml'. The 'Run' tab at the bottom shows the output of the program: 'My name is: Meera', 'There are 5 letters in my name', 'Letters in my name: M e e r a', and 'Process finished with exit code 0'.

```
> 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++) {  
             [  
             ]  
             System.out.print(Name[i] + " ");  
             [  
             ]  
         }  
         [  
         ]  
         System.out.println(); // For a clean newline at the end  
         [  
         ]  
     }  
     [  
     ]  
 }  
 [  
 ]
```

src  
  └ Main  
    └ Task020  
      └ Task020  
      └ .gitignore  
      └ Task020.iml  
  └ External Libraries  
  └ Scratches and Consoles

Run Task020

```
↑ My name is: Meera  
↓ There are 5 letters in my name  
≡ Letters in my name: M e e r a  
≡ Process finished with exit code 0
```

Task 29 home task  
**shallow copy vs deep copy**

The screenshot shows the IntelliJ IDEA interface with a Java code editor and a terminal window.

**Java Code Editor:**

```
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++) {
```

**File Structure:**

- out
- src
  - Main
  - Test
- .gitignore
- Array.java
- External Libraries
- Scratches and Consoles

**Terminal Output:**

```
*C:\Program Files\Java\jdk-17\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2025.1\lib\idea_rt.jar=52212"
false
1 2 3 Original Array
1 4 3
Process finished with exit code 0
```

### 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 {
}

```

The screenshot shows the IntelliJ IDEA interface. The left sidebar displays the project structure for 'Task031 [My\_Calculation]'. The main editor window shows two files: 'Main.java' and 'My\_Calculation.java'. 'My\_Calculation.java' is open and contains the following code:

```
15 public class My_Calculation extends Calculation {
16     public void multiplication(int x, int y) { // usage
17         z = x * y;
18         System.out.println("The product of the given numbers: " + z);
19     }
20
21     public static void main(String[] args) {
22         int a = 20, b = 10;
23         My_Calculation demo = new My_Calculation();
24         demo.addition(a, b);
25         demo.Subtraction(a, b);
26         demo.multiplication(a, b);
27     }
28 }
29
30
31 }
```

The 'Run' tab at the bottom shows the output of the program:

```
"C:\Program Files\Java\jdk-17\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2025.1\lib\idea_rt.jar=53134" -Dfile.encoding=UTF-8
The sum of the given numbers: 30
The difference between the given numbers: 10
The product of the given numbers: 200
Process finished with exit code 0
```

### 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

```
public class TypeOfParameters {
    // Method 1: add two integers
    void add(int x, int y) { System.out.println("x = " + x + ", y = " + y); }

    // Method 2: add three integers (overloaded)
    void add(int x, int y, int z) { System.out.println("x = " + x + ", y = " + y + ", z = " + z); }

    // main method
    public static void main(String[] args) {
        TypeOfParameters obj = new TypeOfParameters();

        obj.add(x: 10, y: 20, z: 30); // calls add(int, int, int)
        obj.add(x: 50, y: 100); // calls add(int, int)
    }
}
```

Run TypeOfParameters

```
*C:\Program Files\Java\jdk-17\bin\java.exe* "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2025.1\lib\idea_rt.jar=60216" -Dfile.encoding=UTF-8
x = 10, y = 20, z = 30
x = 50, y = 100

Process finished with exit code 0
```

Activate Windows

## 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

```
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) {  
  
        Main.java SequenceOfParameters.java  
1  public class SequenceOfParameters {  
14  
15  
16  public static void main(String[] args) {  
17      SequenceOfParameters obj = new SequenceOfParameters();  
18  
19      obj.add('d', 'a');  
20      obj.add(100, 100);  
21  }  
22 }  
23  
  
Run SequenceOfParameters x  
C:\Program Files\Java\jdk-17\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2025.1\lib\idea_rt.jar=56131" -Dfile.encoding=UTF-8  
Char method called  
x = d, y = a  
Int method called  
x = 100, y = 100
```

### 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)  
}
```

The screenshot shows the IntelliJ IDEA interface with the following details:

- Code Editor:** Displays the `Task036` class with two `add` methods and a `main` method.
- Project Tool Window:** Shows the project structure with `.idea`, `out`, `src` (containing `Main` and `Task036`), `.gitignore`, and `taskss.iml`.
- Run Tool Window:** Shows the output of the program:

```
"C:\Program Files\Java\jdk-17\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2025.1\lib\idea_rt.jar  
Method add(float x, int y) called  
x = 10.5, y = 60  
Method add(int x, float y) called  
x = 100, y = 80.8  
> Process finished with exit code 0
```

```
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();  
        obj.add( x: 10.50f, y: 60);    // Calls add(float, int)  
        obj.add( x: 100, y: 80.80f);   // Calls add(int, float)  
    }  
}
```

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(){  
    }  
}
```

```
bin.java      Employee.java      Hr.java
public class Employee { 1 usage 1 inheritor
    private int pwd; 3 usages
    protected int salary; 3 usages
    public int empId; 3 usages

    public Employee() { 1 usage
        pwd = 0;
        salary = 0;
        empId = 0;
    }
    ...
    // Setter method for pwd
    public void setPwd(int pwd) { 1 usage
        this.pwd = pwd;
    }
}
```

```
> .idea
> out
< src
    < Employee
    < Hr
    < Main
    < .gitignore
    < Task0036.iml
> External Libraries
< Scratches and Consoles
11
12
13    // Setter method for pwd
14    public void setPwd(int pwd) { 1 usage
15        this.pwd = pwd;
16    }
17
18    // Getter method for pwd
19    public int getPwd() { 1 usage
20        return pwd;
21    }
22
23

Run  Hr x
G  |  :
C:\Program Files\Java\jdk-17\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2025.1\lib\idea_rt.jar=56506" -Dfile.encoding=UTF-8
Password (via getter): 1254
Salary (protected): 50000
Employee ID (public): 10001
Process finished with exit code 0
Activate Windows
```

### 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
```

```
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
```

```
43 |   public String getAddress() { no usages
44 |     return address;
45 |   }
46 |
47 |   public void setAddress(String newAddress) { no usages
48 |     address = newAddress;
49 |   }
50 |
51 |   public int getNumber() { no usages
52 |     return number;
53 |   }
54 | }
```

Build Build Output ×

Task38: build failed At 05-06-2025 17:20 with 2 errors, 128 ms

AbstractClassDemo.java src 2 errors

class AbstractDemo is public, should be declared in a file named AbstractDemo.java

Employee is abstract; cannot be instantiated

## Task 039

Rewrite the above code to give the output without errors

```
// 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
>
>         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 getAddress() { no usages
        return address;
    }

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

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

The screenshot shows the IntelliJ IDEA interface with the following details:

- Code Editor:** Displays the `Employee` class with its methods `getAddress`, `setAddress`, and `getNumber`. It also shows the `Manager` class extending `Employee` and its constructor.
- File Tree:** Shows the project structure with files like `AbstractDemo.java`, `Main`, `.gitignore`, `Task38AbstractDemo.iml`, `External Libraries`, and `Scratches and Consoles`.
- Run History:** Shows the output of the `AbstractDemo` run, including the construction of an `Employee` and a `Manager`, and a mail check operation.

## 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();
    }
}
```

```
// Abstract base class
@4 abstract class Gadgets { 5 usages
    @4 abstract void turnOn(); 4 usages 4 implementations
    @4 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.");
    }
}

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

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

```
// 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
```

```
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) {
```

```
// Main public class named GadgetTest
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();
        remote.turnOff();
    }
}
```

```
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();
}
```

The screenshot shows the IntelliJ IDEA interface. On the left is the project structure tree with nodes like .idea, out, src, Main, task0040.java, .gitignore, task0040.iml, External Libraries, and Scratches and Consoles. The src folder is expanded, and task0040.java is selected. The main method code is displayed in the editor:

```
60  public class task0040 {  
61      public static void main(String[] args) {  
62          remote = new ACRemote();  
63          remote.turnOn();  
64          remote.turnOff();  
65  
66          remote = new FanRemote();  
67          remote.turnOn();  
68          remote.turnOff();  
69  
70          remote = new CoolerRemote();  
71          remote.turnOn();  
72          remote.turnOff();  
73      }  
74  }
```

Below the editor is the run configuration bar with "Run task0040". The run output window shows the following text:  
"C:\Program Files\Java\jdk-17\bin\java.exe" -javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2025.1\lib\idea\_rt.jar=5300,C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2025.1\bin  
↑ TV is turned ON.  
↓ TV is turned OFF.  
→ AC is turned ON.  
← AC is turned OFF.  
> Fan is turned ON.  
> Fan is turned OFF.

## 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] + " ");  
    }  
}
```

The screenshot shows the IntelliJ IDEA interface. The left sidebar displays a project structure for 'Task021' with files like Main.java, New.java, and Task021.iml. The main editor window shows two tabs: 'Main.java' and 'New.java'. The 'New.java' tab is active, displaying the following code:

```
1  class New {
2      Runnable class
3  }
4  public static void main(String[] args) {
5      // initializing array
6      int[] arr = { 1, 2, 3, 4, 5 };
7
8      // size of array
9      int n = arr.length;
10
11     // traversing array
12     for (int i = 0; i < n; i++) {
13         System.out.print(arr[i] + " ");
14     }
15
16
17
18 }
```

The terminal window at the bottom shows the execution results:

```
"C:\Program Files\Java\jdk-17\bin\java.exe" -javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2025.1\lib\idea_rt.jar=5334,C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2025.1\bin
1 2 3 4 5
Process finished with exit code 0
```

## 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;
```

The screenshot shows the IntelliJ IDEA interface. In the top navigation bar, 'Main' is selected. Below it, the project structure shows 'task022' as the active file. The code editor contains the following Java code:

```
14 arr[3] = 40;
15 arr[4] = 50;
16
17 // accessing the elements of the specified array
18 for (int i = 0; i < arr.length; i++)
19     System.out.println("Element at index " + i + " : " + arr[i]);
20 }
21
22 |
```

In the bottom terminal window, the output of the run command is displayed:

```
*C:\Program Files\Java\jdk-17\bin\java.exe* --javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2025.1\lib\idea_rt.jar=54597* -Dfile.encoding=UTF-8
Element at index 0 : 10
Element at index 1 : 20
Element at index 2 : 30
Element at index 3 : 40
Element at index 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+ " }");
}

}
```

```
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");
```

```
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 + " }");
        }
    }
}
```

The screenshot shows the IntelliJ IDEA interface with the following details:

- Project Structure:** On the left, the project tree shows files: Mainss.java, task022, .gitignore, day8array.iml, External Libraries, and Scratches and Consoles.
- Code Editor:** The main window displays Java code in Mainss.java. The code prints elements from an array named students. The code is as follows:

```
20
21     for (int i = 0; i < students.length; i++) {
22         System.out.println("Element at " + i + " : { "
23             + students[i].roll_no + " " + students[i].name +
24         }
25     }
26 }
27 
```

- Terminal:** Below the editor, the terminal window shows the execution of the code and its output:

```
"C:\Program Files\Java\jdk-17\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2025.1\lib\idea_rt.jar"
Hello and welcome!i = 1
i = 2
i = 3
i = 4
i = 5
```

## 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]);
    }
}
```

```

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);
    }
}

```

The screenshot shows the IntelliJ IDEA interface. The code editor displays the Java code for Task0025. The file structure on the left shows the project tree with 'Main' and 'Task0025' selected. The run console at the bottom shows the output of the program: "Rows : 3" and "Columns : 3".

```

Main
Task0025
.gitignore
Task0025.iml
External Libraries
Scratches and Consoles

Run Task0025 ×
C: [Run] [Stop] [Output]
↑
↓
Rows : 3
Columns : 3
Process finished with exit code 0

```

## 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 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] + " ");
            }
        }
    }
}
```

The screenshot shows the IntelliJ IDEA interface. On the left, the project structure is visible with a 'src' folder containing 'Main', 'Task026', '.gitignore', and 'Task026.iml'. Below that is 'External Libraries' and 'Scratches and Consoles'. The main area is a code editor with the following Java code:

```
11
12
13     // Traversing and printing the 2D array
14     for (int i = 0; i < 3; i++) {           // Outer loop for rows
15         for (int j = 0; j < 3; j++) {       // Inner loop for columns
16             System.out.print(arr[i][j] + " ");
17         }
18         System.out.println();               // Move to the next line after each row
19     }
20
21 }
```

Below the code editor is a terminal window titled 'Run Task026'. It shows the command being run: "C:\Program Files\Java\jdk-17\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2025.1\lib\idea\_rt.jar" -Dfile.encoding=UTF-8 Task026. The output in the terminal is:

```
2 7 9
3 6 1
7 4 2
```

## 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);
    }

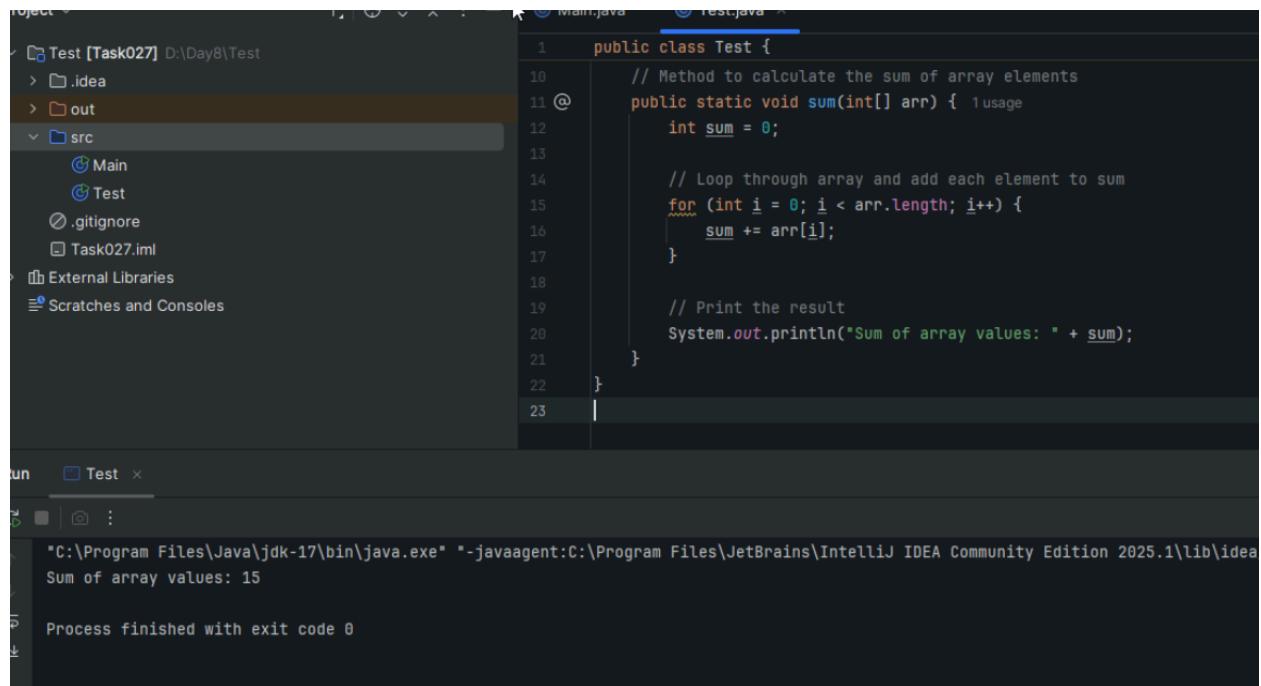
    public static void sum(int[] arr)
    {
        // getting sum of array values
        int sum = 0;

        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
```



```
public class Test {
    // Method to calculate the sum of array elements
    public static void sum(int[] arr) {
        int sum = 0;

        // Loop through array and add each element to sum
        for (int i = 0; i < arr.length; i++) {
            sum += arr[i];
        }

        // Print the result
        System.out.println("Sum of array values: " + sum);
    }
}
```

## 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
```

The screenshot shows the IntelliJ IDEA interface. On the left, the Project tool window displays a project named 'Task028' with a 'src' directory containing 'Main.java' and 'Tests.java'. The 'out' directory is also visible. On the right, the Editor tab bar has 'Main.java' and 'Tests.java' tabs, with 'Tests.java' currently selected. The code in 'Tests.java' is:

```

4 class Tests {
5     public static void main(String[] args) {
6         // Call method m1() and store the returned array
7         int[] arr = m1();
8
9         // Print each element of the array
10        for (int i = 0; i < arr.length; i++) {
11            System.out.print(arr[i] + " ");
12        }
13    }
14
15    // Method that returns an array of integers
16    public static int[] m1() { 1 usage
17        // Return an array directly
18        return new int[] { 1, 2, 3 };
19    }
}

```

Below the editor, the Run tool window shows the output of the 'Tests' run configuration:

```
"C:\Program Files\Java\jdk-17\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA Community Edition 2025.1\lib\idea_rt.jar=565,127.0.0.1:47285" -Dfile.encoding=UTF-8 Main
1 2 3
Process finished with exit code 0
```

## 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

```

The screenshot shows an IDE interface with the following details:

- Project:** Task030 (selected in the Project tree)
- Task030.java:** Contains the following code:
 

```

3 public class Task030 {
4     public static void main(String[] args) {
5         // Declare and initialize a 2D array
6         int[][] intArray = { { 1, 2, 3 }, { 4, 5 } };
7
8         // Clone the 2D array (shallow copy)
9         int[][] cloneArray = intArray.clone();
10
11        // Check if the original and cloned arrays are the same object
12        System.out.println(intArray == cloneArray); // Expected: false
13
14        // Check if the sub-arrays are the same objects (shared in shallow copy)
15        System.out.println(intArray[0] == cloneArray[0]); // Expected: true
16        System.out.println(intArray[1] == cloneArray[1]); // Expected: true
17    }
18 }
```
- Run:** Task030 (selected in the Run dropdown)
- Output:** Shows the console 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);
    }
}

```

The screenshot shows a code editor interface with a dark theme. A tooltip or code completion dropdown is open over the variable 'item' in the line 'String item = "Tomatoes";'. The tooltip displays the variable's type ('String') and its value ('Tomatoes'). The background code includes a 'Customer' class definition and a 'Mart' class definition.

```

class Customer { no usages OFF
    int cost = 40; no usages
    String item = "Tomatoes"; no usages
}

public void purchaseList() { System.out.println("Customer purchased: " + item + " for Rs. ")
}

public class Mart extends Customer {

    void billing() { no usages
        item = "Onions";
        cost = 30;
        System.out.println("Billing item: " + item);
    }
}

```

```
Main.java      Task030.java      Mart.java      Mart.java X  
8         System.out.println("Billing item: " + item);  
9         System.out.println("Billing cost: " + cost);  
0     }  
1  
2  
3     public static void main(String[] args) {  
4         [  
5             Mart m = new Mart();  
6  
7             m.item = "Potatoes";  
8             m.cost = 50;  
9  
10            System.out.println("%%%%%%%%%%%%%");  
11            System.out.println("Item: " + m.item);  
12        }  
13    }
```

```
ect ▾  
Task030 D:\Day8\Task030  
  □ .idea  
  □ out  
  □ src  
    ○ .gitignore  
    □ Task030.iml  
  □ External Libraries  
  □ Scratches and Consoles  
  
Main.java      Task030.java      Mart.java      Mart.java X  
32            System.out.println("%%%%%%%%%%%%%");  
33            System.out.println("Item: " + m.item);  
34            System.out.println("Cost: " + m.cost);  
35            System.out.println("%%%%%%%%%%%%%");  
36  
37            m.billing();  
38        }  
39    }  
40}  
41  
  
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
```