Think and Tell

- A sports academy wants to analyze the performance of all soccer team players based on their age.
- How can a programmer store data that contains the age details of all the 11 athletes?

Ages of Players: Solution 1

Is this an ideal solution?

Can we declare 11 different variables with unique names to hold the age of players?

```
int ageOfPlayer1, ageOfPlayer2, ageOfPlayer3, … ageOfPlayer11;
```

Ages of Players: Solution 2

Is this an ideal solution?

Can you declare only one variable which can hold all 11 values?

Working With Arrays

Think and Tell

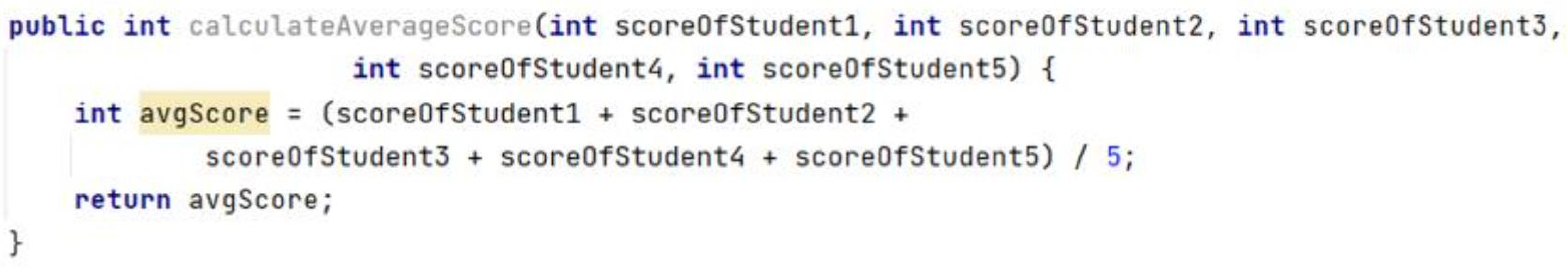
Calculate the Average Score

Individual scores obtained by the top five students of a class are listed below.

How can you create a function to calculate the average score for these students?

- scoreOfStudent1 = 87
- scoreOfStudent2 = 73
- scoreOfStudent3 = 98
- scoreOfStudent4 = 67
- scoreOfStudent5 = 89

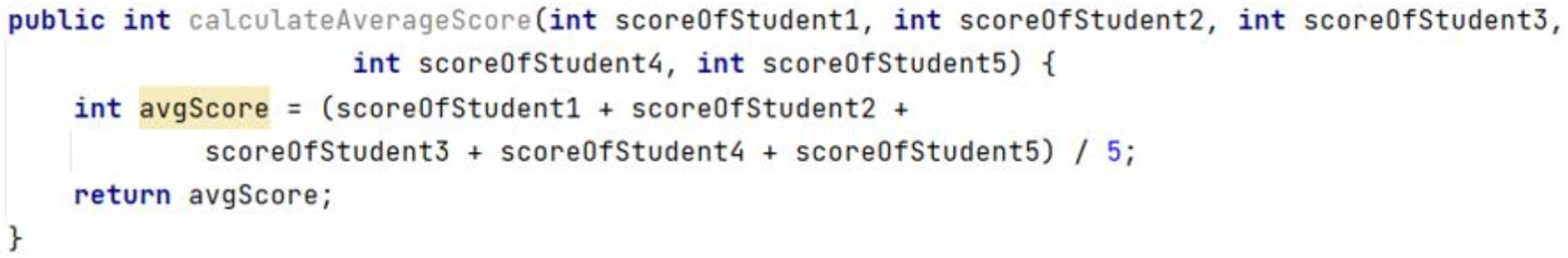
Calculating the Average Score: Solution



```java
public int calculateAverageScore(int scoreOfStudent1, int scoreOfStudent2, int scoreOfStudent3,
                           int scoreOfStudent4, int scoreOfStudent5) {
    int avgScore = (scoreOfStudent1 + scoreOfStudent2 +
            scoreOfStudent3 + scoreOfStudent4 + scoreOfStudent5) / 5;
    return avgScore;
}
```

Passing Values to a Method

Will this solution work, if we have to find the average score for the entire class of 30 students? Can we pass 30 different variables as parameters to a function?



```java
public int calculateAverageScore(int scoreOfStudent1, int scoreOfStudent2, int scoreOfStudent3,
                                 int scoreOfStudent4, int scoreOfStudent5) {
    int avgScore = (scoreOfStudent1 + scoreOfStudent2 +
            scoreOfStudent3 + scoreOfStudent4 + scoreOfStudent5) / 5;
    return avgScore;
}
```

Learning Objectives

- Describe Arrays and its types
- Declare an Array
- Initialize an Array
- Access Array elements using Loop
- Pass Array as a method parameter and return type
- Manipulate values of an Array

Array and Its Types

What Is an Array?

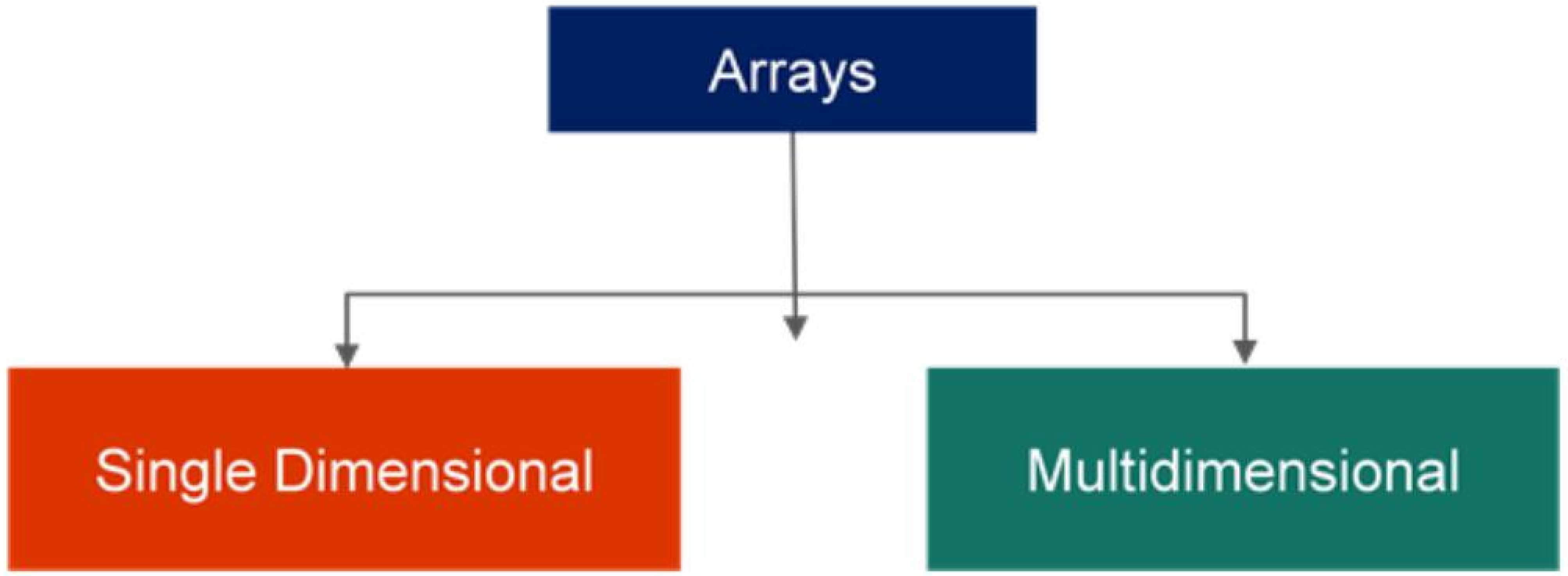
Array Name - ageOfPlayers

Value of array	30	32	33	28	29	30	41	33	32
Index	0	1	2	3	4	5	6	7	8

- An array is a collection of items of similar data type referred by a common name(ex – ageOfPlayers).
- It contains fixed number of items of the same data type.
- Individual data items in arrays are called elements.
- Each element of an array is indexed. Indexing begins at 0.
- An index is the position at which the element is stored in an array.
- Each element is accessed by its numerical index.
- Elements are stored contiguously in memory, i.e., in consecutive memory locations.

Types of Array

- There are two types of Array:
  - **Single** Dimension Array
  - **Multidimensional** Dimension Array



Single Dimension Array

- A collection of elements with a single index value.
- May have multiple columns but only one row.

	Index 0	Index 1	Index 2	Index 3	Index 4
athleteAge	16	17	16	20	21
	athleteAge[0]	athleteAge[1]	athleteAge[2]	athleteAge[3]	athleteAge[4]

Multidimensional Array

- It is a collection of elements with a multi-index value.
- It can have multiple rows and columns.
- It consists of arrays of arrays.

Country Name	Gold	Silver	Bronze
USA	46	28	29
CHINA	38	28	22
INDIA	29	17	19

Declare an Array

Array Declaration

- Arrays are declared with appropriate datatypes and size.
- Below are two valid array declaration syntax:

```
datatype variableName [];
datatype [] variableName;
```

- The datatype can be of any type like int, float, String etc.
- The name of the array variable is `variableName`.
- [ ] - Square brackets is the notion for declaring an array.

Declaring an Array of Multiple Datatypes

- Declaring an int array

```
int [] array;
```

- Declaring a Float array

```
float [] array;
```

- Declaring a String array

```
String[] array;
```

Initialize an Array

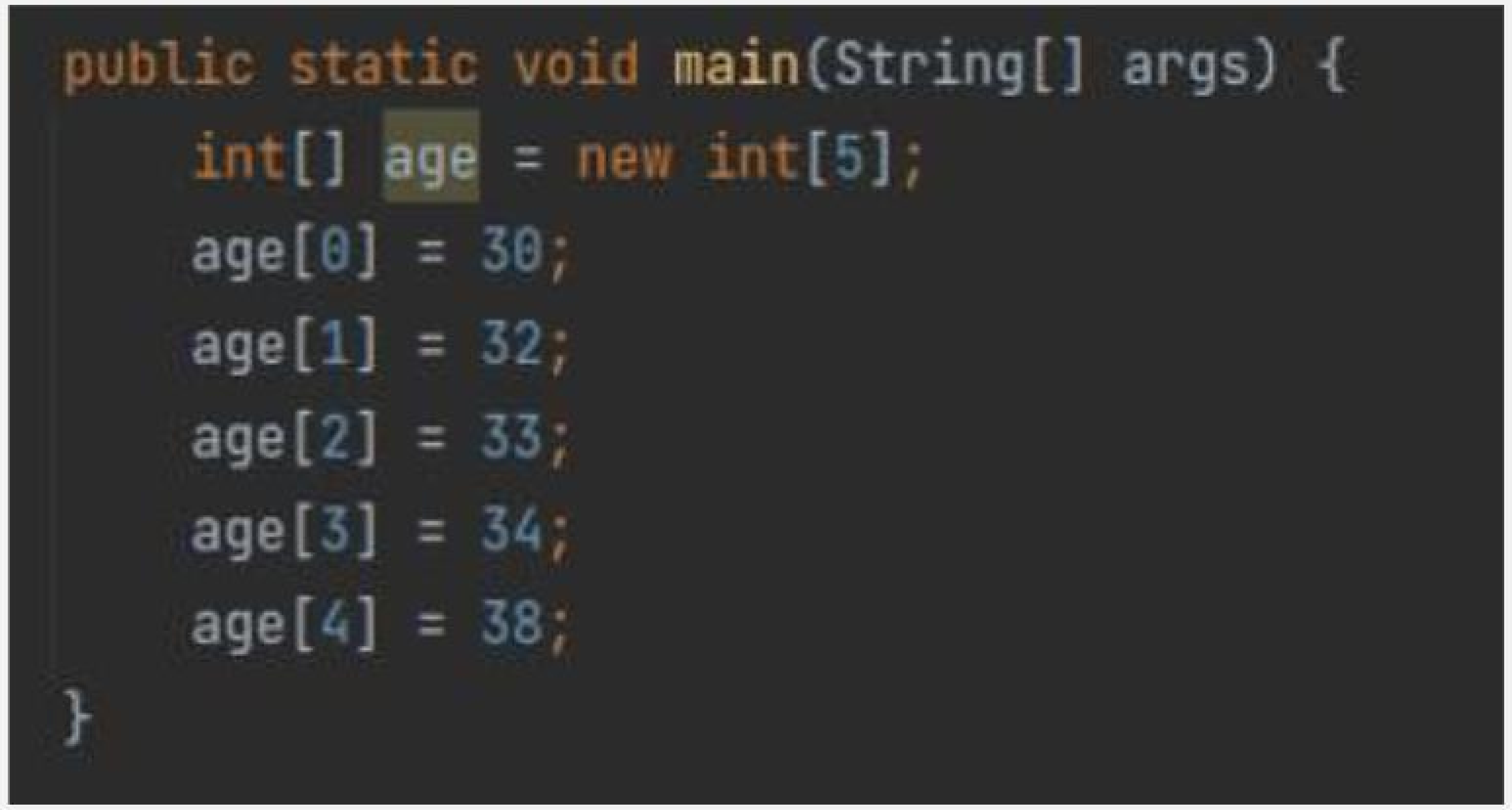
Array Declaration With Size

- `int age []  = new age[10];`
- The number 10 indicates the number of elements in the array or the size of the array.
- Once an array is created, its size is fixed. It cannot be changed.
  - `int size = array.length`
  - Here, the size of the array is 10 which can be determined using the `length` property.
  - Here, the age is called the **reference** variable that will hold ten integer values.

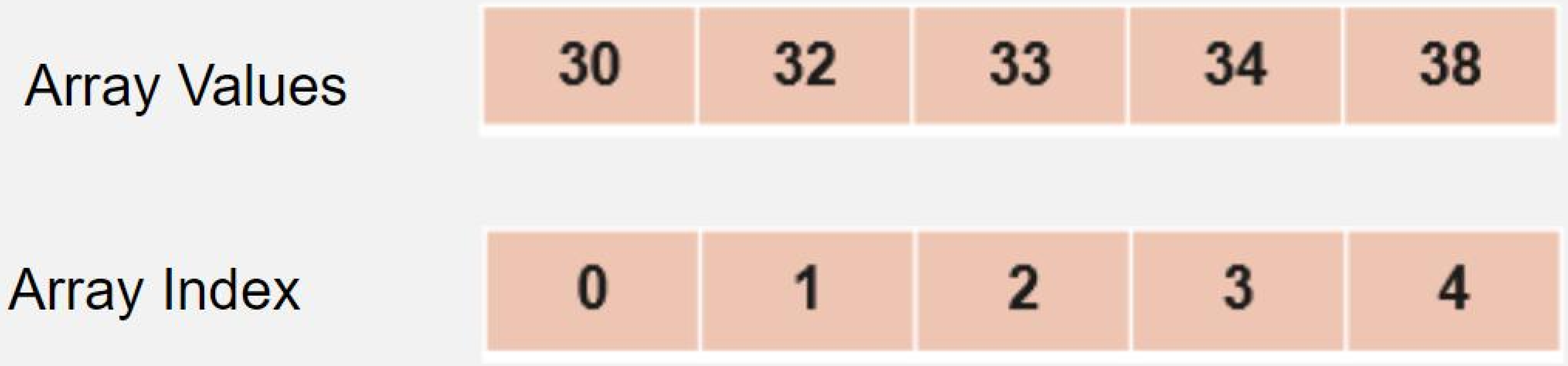
Note: We will discuss more about `new` keyword in next course.

Initializing Values to a Declared Array of Fixed Size

- Declaring and initializing an int array of size 5.
- Index is the position where array elements are stored.
- Arrays start with 0 indexing, so the first element is added in 0th position.
- Elements are added from 0th index till fourth index for an array of size 5.



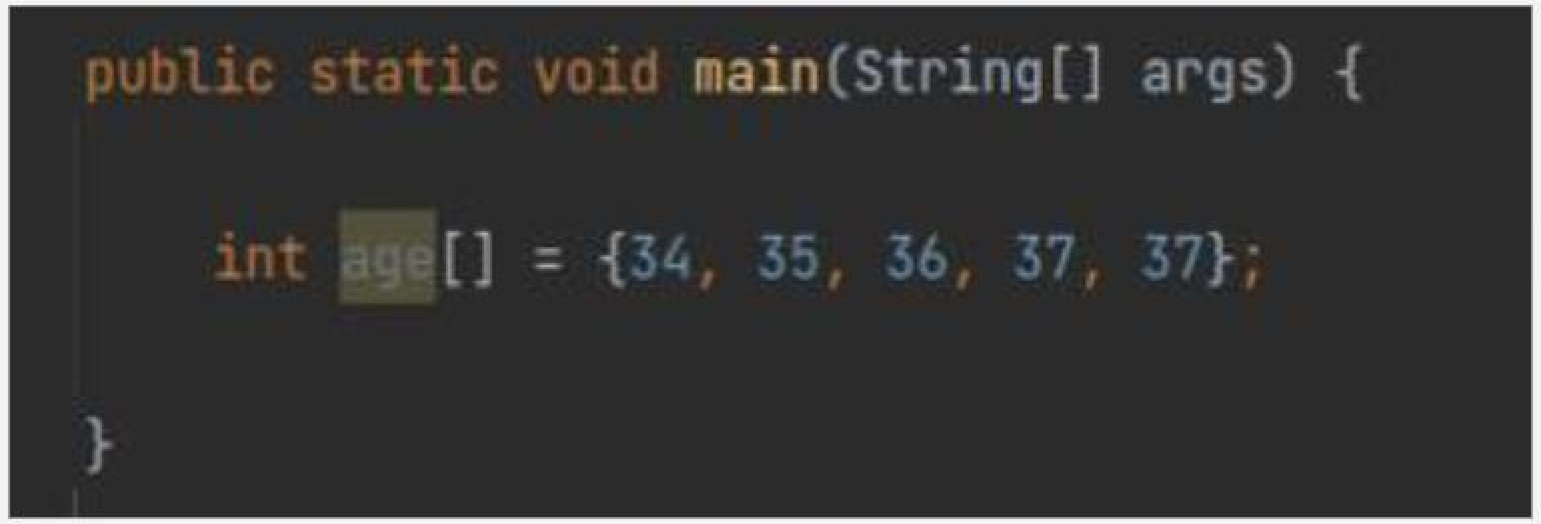
```
public static void main(String[] args) {
    int[] age = new int[5];
    age[0] = 30;
    age[1] = 32;
    age[2] = 33;
    age[3] = 34;
    age[4] = 38;
}
```



Array Values	30	32	33	34	38
Array Index	0	1	2	3	4

Declare and Initialize an Array

- The declaration, initialization and assigning of values to an array can happen in one step.
- When the size of the array and variables of the array are already known, array literals can be used.
- Array literals are just like primitive type, where an array is declared, initialized without the new keyword.
- The number of elements determine the length of the created array.
- The age array is of size 5.



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

    int age[] = {34, 35, 36, 37, 37};

}
```

Array Values

34	35	36	37	37

Index

0	1	2	3	3

Quick Check

How do you determine the number of elements in an array?

```
int empId = new int [4];
```

1. empId.size()
2. empId.length
3. empId.length()
4. empId.[Size]

Quick Check: Solution

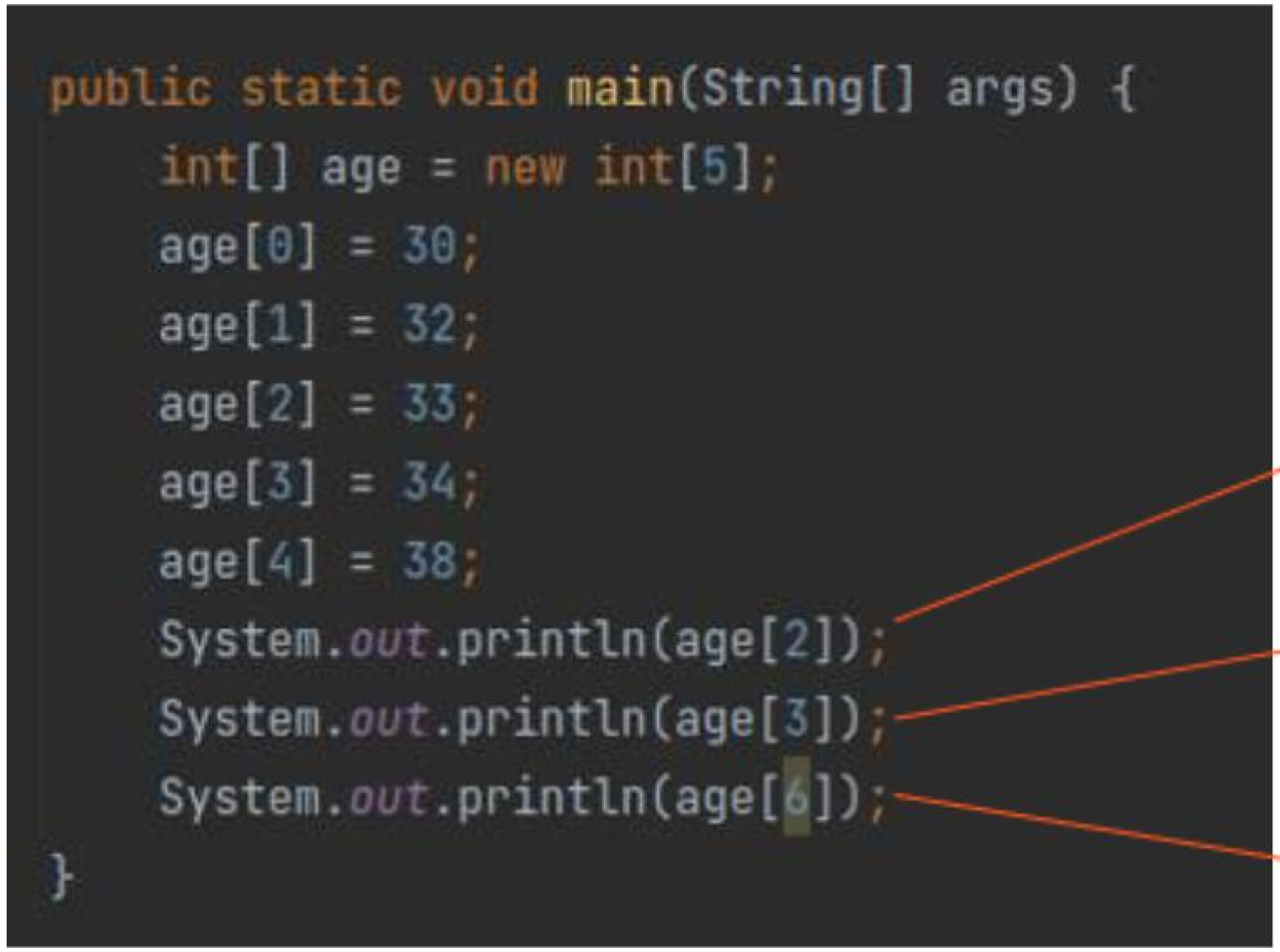
How do you determine the number of elements in an array?

```
int empId = new int [4];
```

1. empId.size()
2. **empId.length**
3. empId.length()
4. empId.[Size]

Accessing the Array Using Index

- Each element in the array is accessed via its index. The index begins with 0 and ends at (total array size)-1.
- The index value is represented within the [] brackets.



```
public static void main(String[] args) {
    int[] age = new int[5];
    age[0] = 30;
    age[1] = 32;
    age[2] = 33;
    age[3] = 34;
    age[4] = 38;
    System.out.println(age[2]);
    System.out.println(age[3]);
    System.out.println(age[6]);
}
```

It will print the value at index position [2] : 33

It will print the value at index position [3] : 34

The last line will give run time error as the size of the array is only 5 and we are trying to access the sixth index position.

Soccer Team

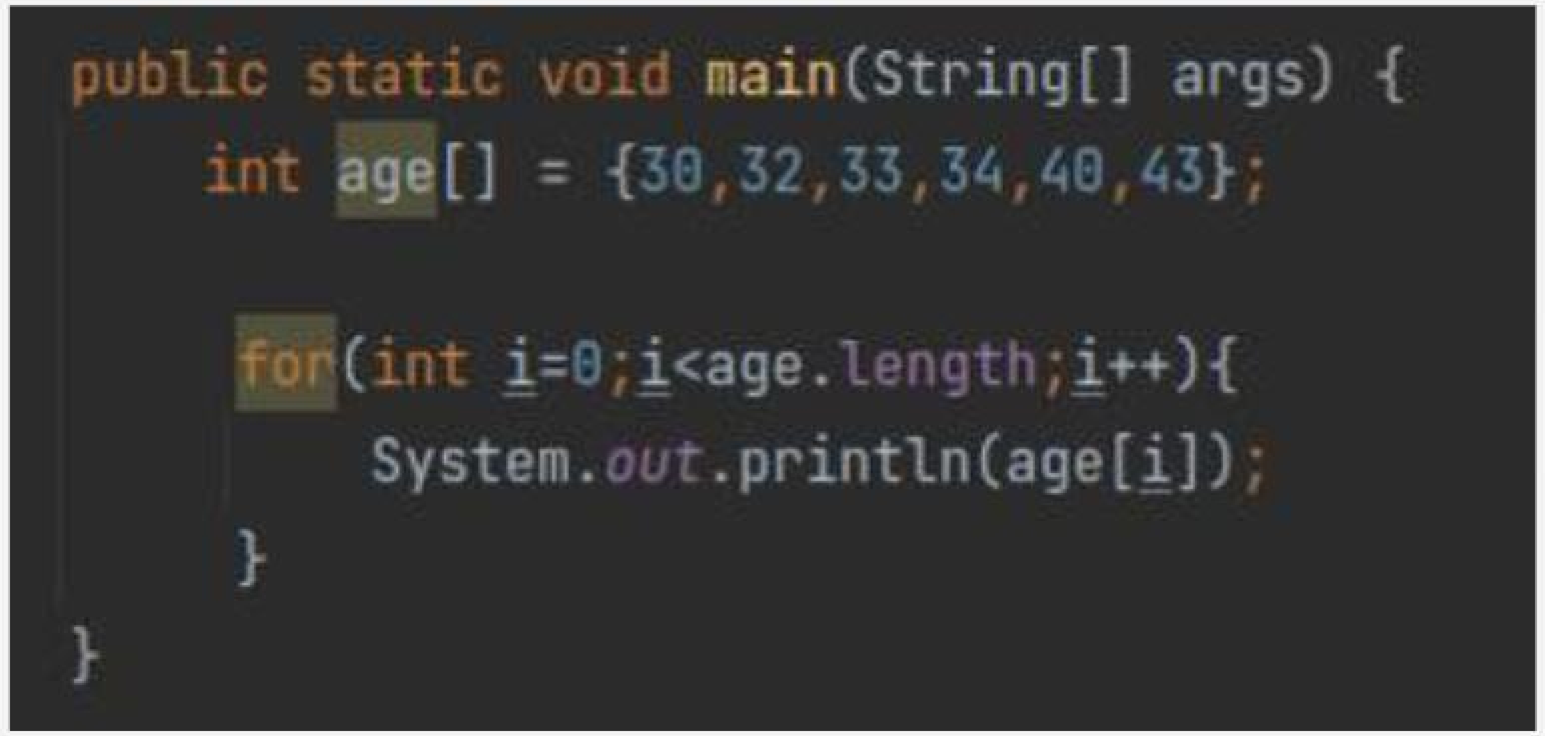
Write a program to store the age of all the players of a soccer team and represent them in an array using single-dimensional arrays.

Task 1: Create array.
Task 2: Assign values to the array.
Task 3: Print all the values of array using index position.

Click here for the solution.
The workbench must be used for the demonstration.

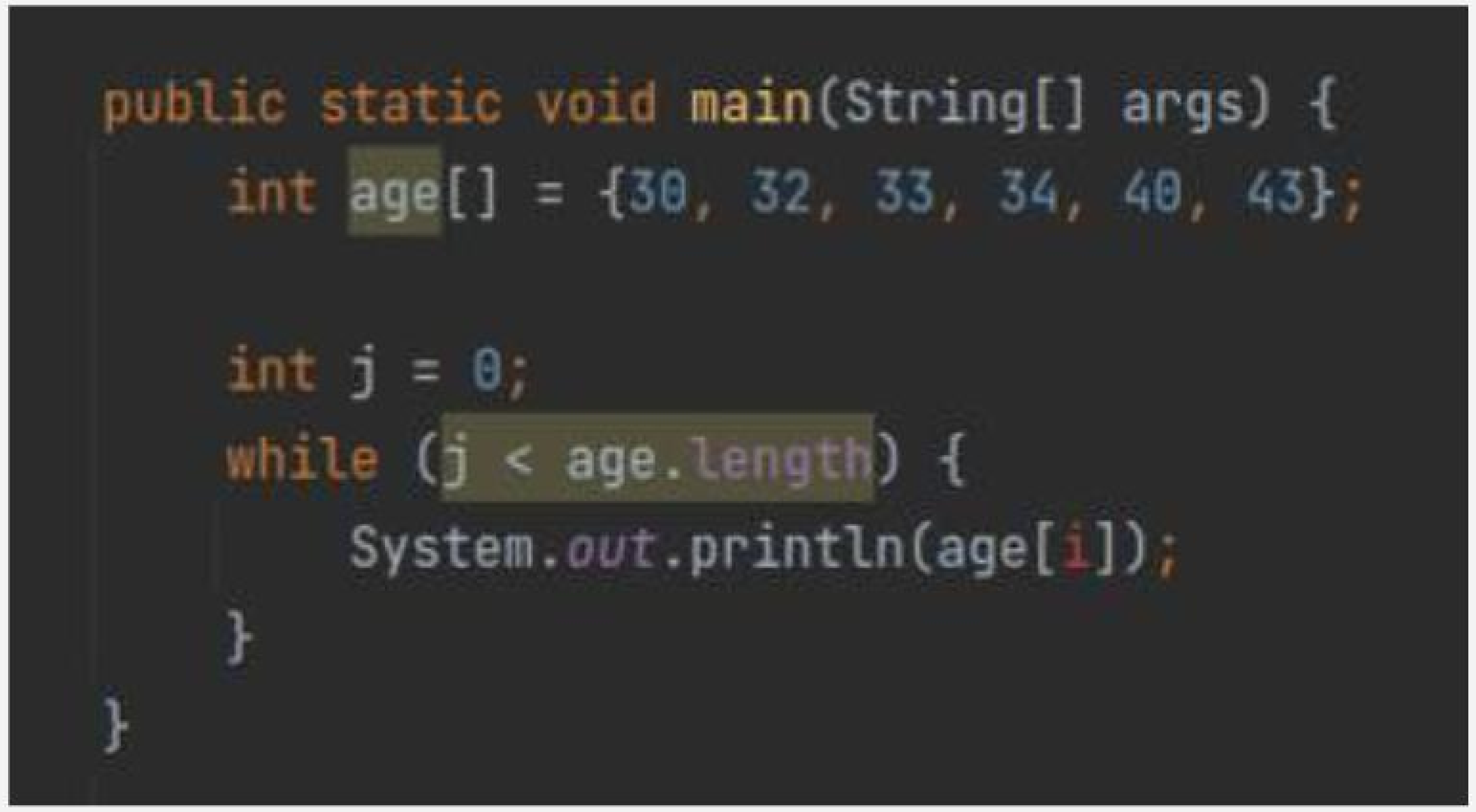
DEMO

Access Array Elements Using Loop



```
public static void main(String[] args) {
    int age[] = {30,32,33,34,40,43};

     for(int i=0;i<age.length;i++){
         System.out.println(age[i]);
     }
}
```



```
public static void main(String[] args) {
    int age[] = {30, 32, 33, 34, 40, 43};

    int j = 0;
    while (j < age.length) {
        System.out.println(age[i]);
    }
}
```

Traversing an Array Using Loops

- Enumerating every element of an array line by line is cumbersome.
- Loops are used to access array elements by their index position.
- The for loop is used to iterate through an array and print the elements of the array.
- Initializing `i = 0` as indexing start with 0.
- Checking boolean condition `i<age.length`
- Incrementing `i` to get the next index position.
- Length is a property of array which gives the size of array.
- The while loop can also be used to iterate through the array.

Even Numbers

Write a program that creates an array of numbers.
Iterate through the array and find all the even numbers and sum them.
Print the sum value.

Click here for the solution.
The workbench must be used for the demonstration.

DEMO

Quick Check

Which of the following statements are true about an array?

1. An array can contain duplicate elements.
2. Each element of an array is indexed. Indexing begins at 0.
3. Length property is used to know the size of an array.
4. An array can expand automatically when its size is full.

Quick Check

Which of the following statements are true about an array?

1. An array can contain duplicate elements.

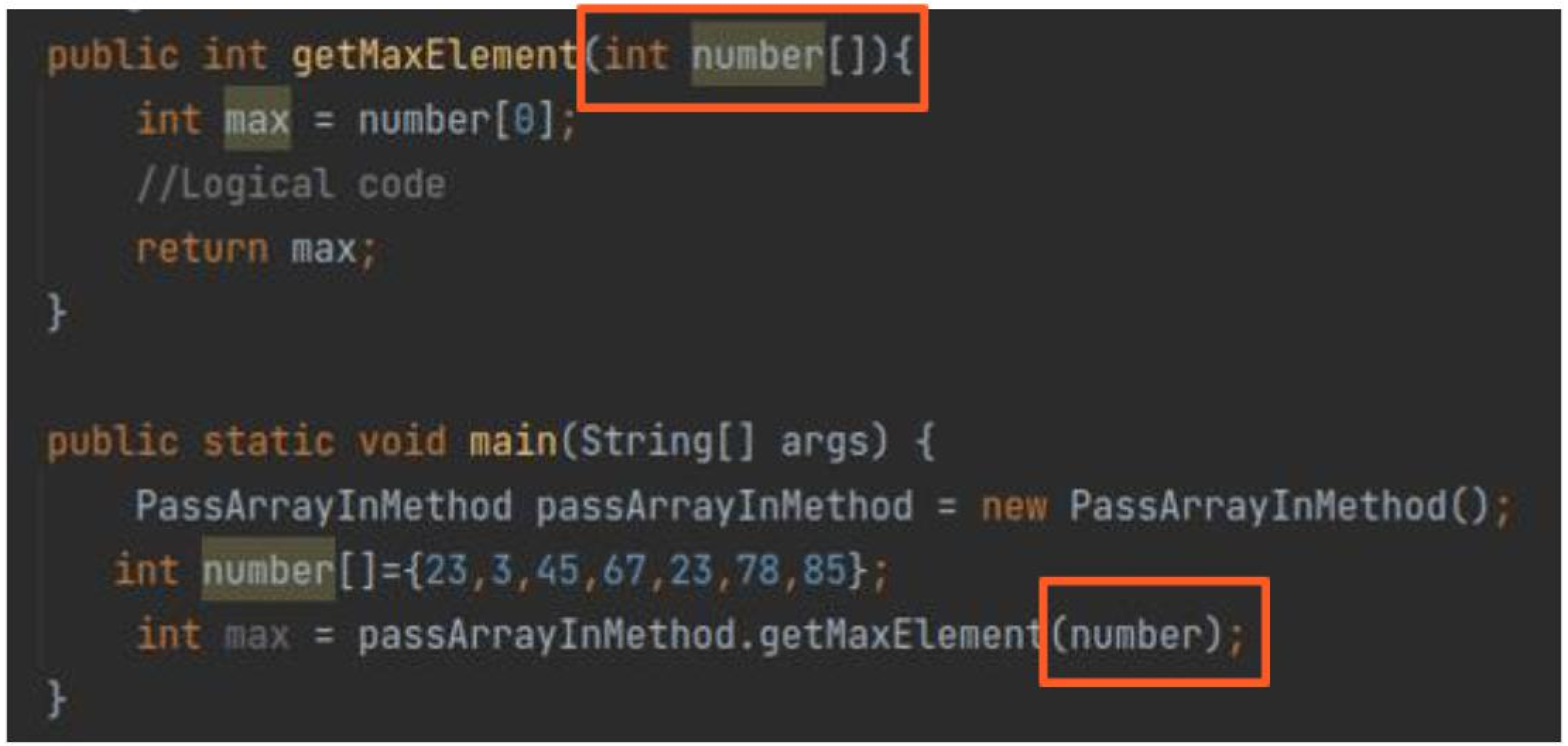
2. Each element of an array is indexed. Indexing begins at 0.

3. Length property is used to know the size of an array.

4. An array can expand automatically when its size is full.

Array as a Method Parameter and Return Type

Array as a Parameter

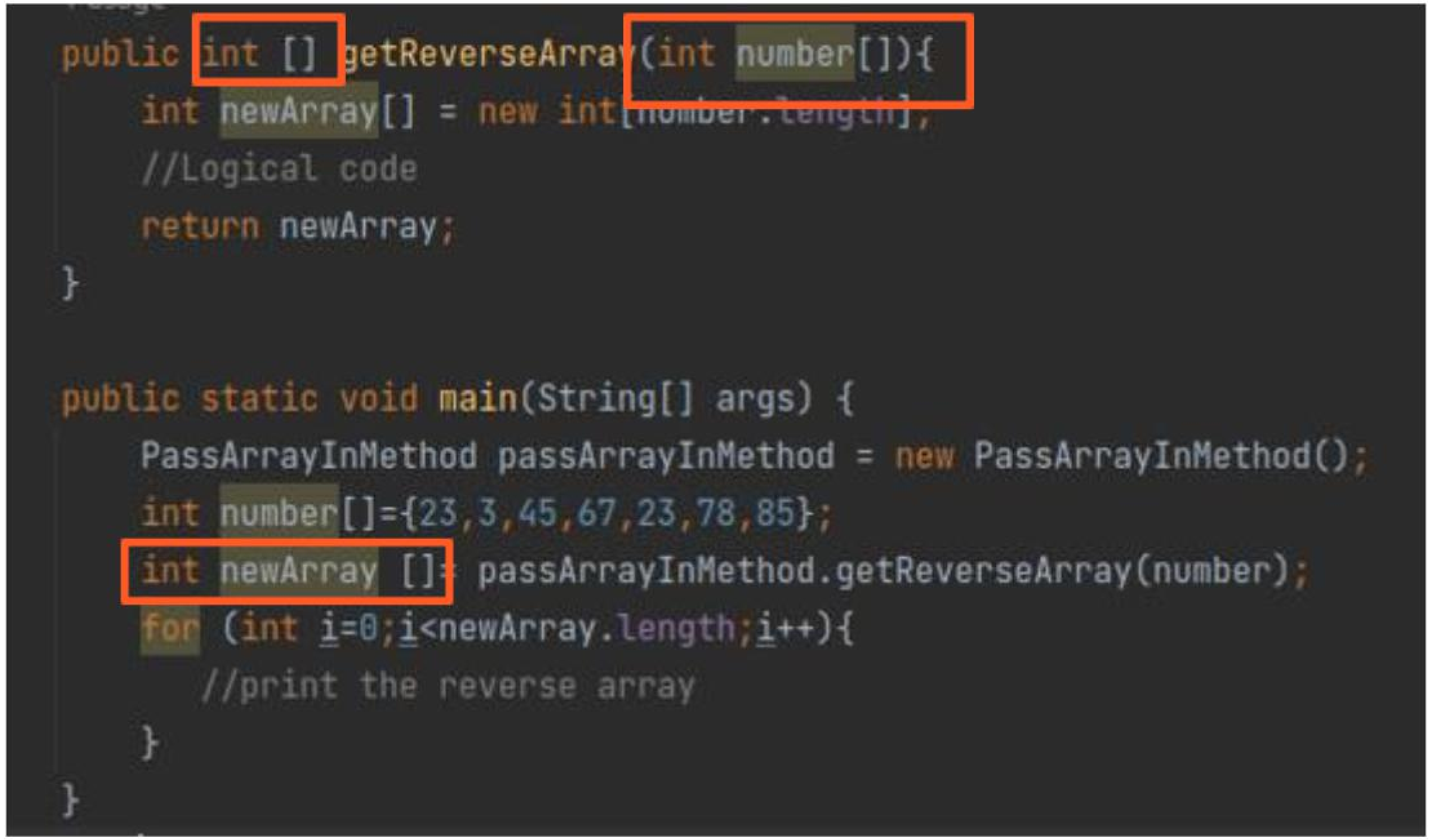


```
public int getMaxElement(int number[]){
    int max = number[0];
    //Logical code
    return max;
}

public static void main(String[] args) {
    PassArrayInMethod passArrayInMethod = new PassArrayInMethod();
   int number[]={23,3,45,67,23,78,85};
    int max = passArrayInMethod.getMaxElement(number);
}
```

- An array can be passed to a method as a parameter just like normal variables.
- Inside the main method number array is created.
- `getMaxElement` method has a parameter of type array – `int number[]`
- When calling the method `getMaxElement` in the main method we pass only the reference variable `number` without the `[]` brackets.

Array as a Return Type and Also as Parmeter



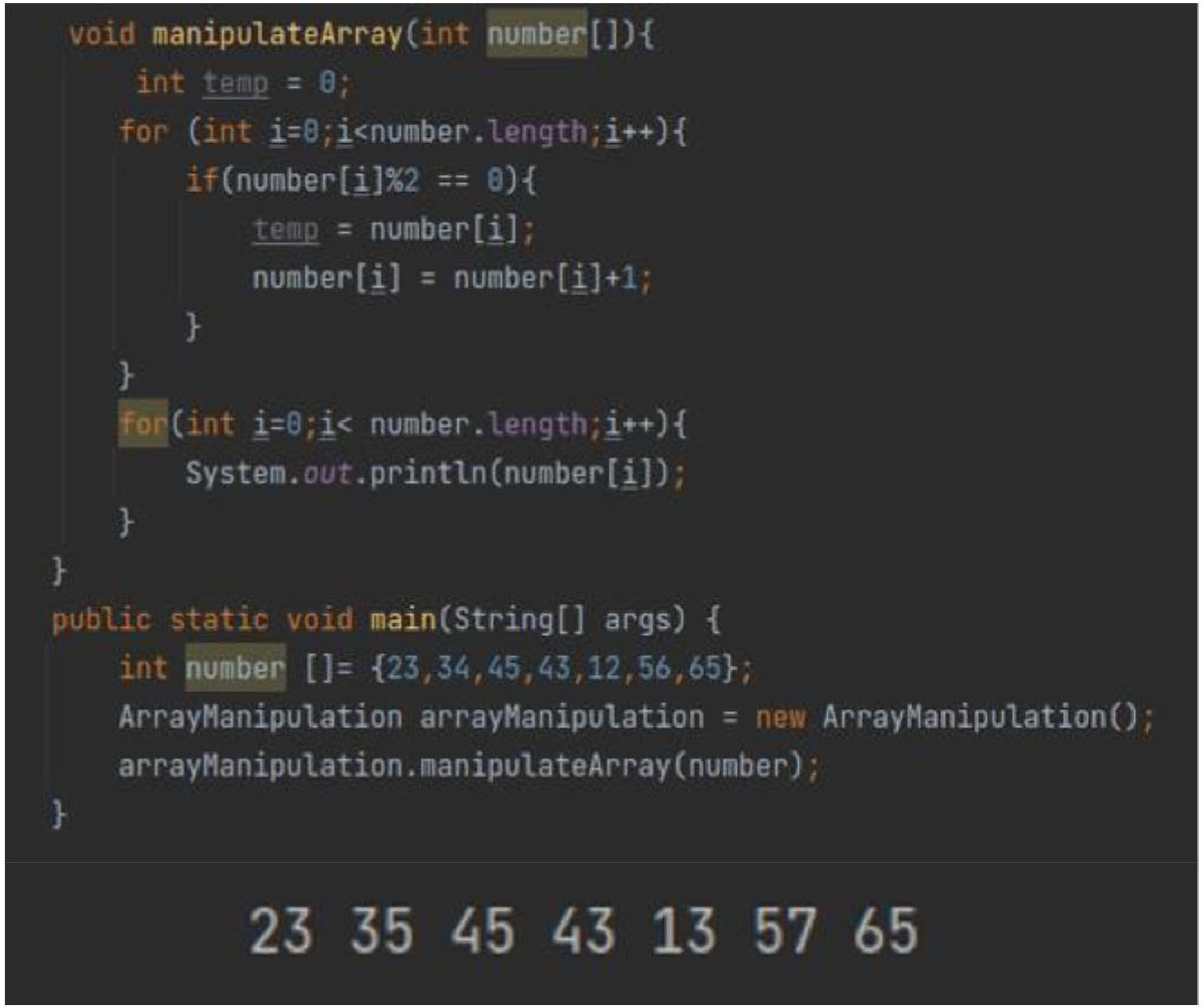
```
public int [] getReverseArray(int number[]){
    int newArray[] = new int[number.length];
    //Logical code
    return newArray;
}

public static void main(String[] args) {
    PassArrayInMethod passArrayInMethod = new PassArrayInMethod();
    int number[]={23,3,45,67,23,78,85};
    int newArray []= passArrayInMethod.getReverseArray(number);
    for (int i=0;i<newArray.length;i++){
        //print the reverse array
    }
}
```

- Array can be a return type from a method.
- A variable of type array needs to be created to hold the value that is getting returned from the getReverseArray() method.
- GetReverseArray method returns as array.
- When specifying the return type in the method the [] brackets must be specified along with the data type.

Array Manipulation

Array Manipulation



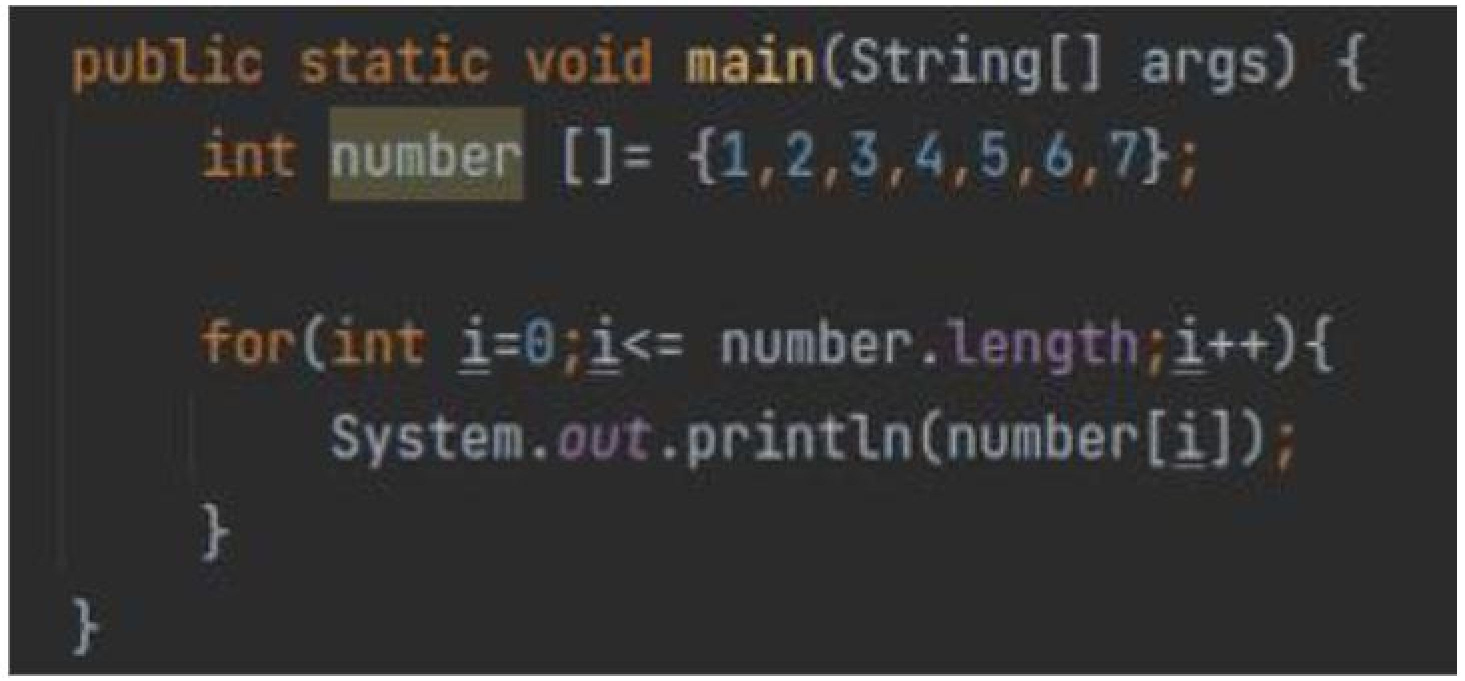
```
void manipulateArray(int number[]){
    int temp = 0;
   for (int i=0;i<number.length;i++){
       if(number[i]%2 == 0){
           temp = number[i];
           number[i] = number[i]+1;
       }
   }
   for(int i=0;i< number.length;i++){
       System.out.println(number[i]);
   }
}
public static void main(String[] args) {
    int number []= {23,34,45,43,12,56,65};
    ArrayManipulation arrayManipulation = new ArrayManipulation();
    arrayManipulation.manipulateArray(number);
}
```

```
23 35 45 43 13 57 65
```

- The method `manipulateArray` takes an array as parameter.
- This array gets manipulated by replacing all even digits with the odd digits.
- Each element is checked if even or odd.
- If the element is even, then that element is replaced with the next odd number by adding 1 to it.

Quick Check

Predict the output of the following code:



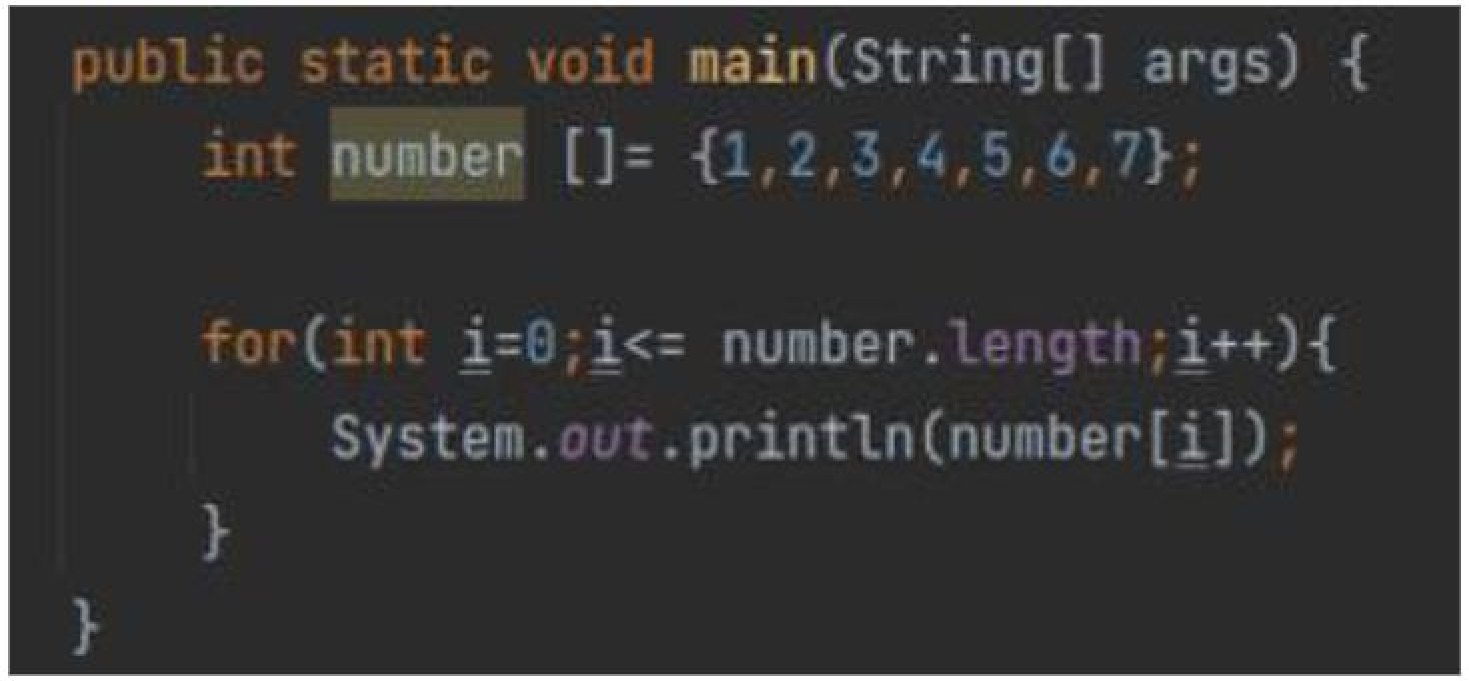
```
public static void main(String[] args) {
    int number []= {1,2,3,4,5,6,7};

    for(int i=0;i<= number.length;i++){
        System.out.println(number[i]);
    }
}
```

1. 1,2,3,4,5,6,7
2. 1,2,3,4,5,6,7, ArrayIndexOutOfBoundException
3. Syntax Error
4. ArrayIndexBoundOfException

Quick Check: Solution

Predict the output of the following code:



```
public static void main(String[] args) {
    int number []= {1,2,3,4,5,6,7};

    for(int i=0;i<= number.length;i++){
        System.out.println(number[i]);
    }
}
```

1. 1,2,3,4,5,6,7
2. **1,2,3,4,5,6,7, ArrayIndexOutOfBoundException**
3. Syntax Error
4. ArrayIndexBoundOfException

Even to Odd

Write a program that converts all even numbers present in the array to the odd numbers and stores it in another array.

Click on the given link for the solution.
The workbench must be used for the demonstration. Execute the test cases provided in the test folder.

DEMO