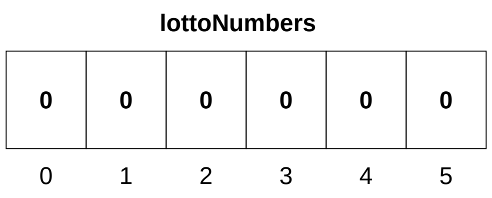
Array Operations

Introduction to Programming

Array basics

- Arrays allow the programmer to work with a group of values of the same type.

```
int[] lottoNumbers = new int [6];
```

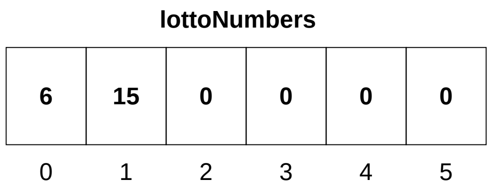


***REMEMBER Array is initialised when created

Array basics

- Individual array elements can be accessed using the **subscript** or **index**.

```java
int[] lottoNumbers = new int [6];
lottoNumbers[0] = 6;
lottoNumbers[1] = 15;
```

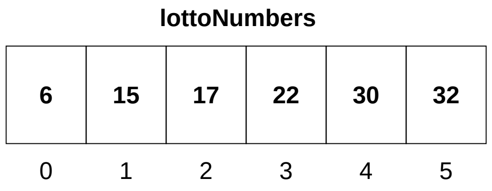


Array basics

- A `for` loop can be used to access all array elements in turn.

```java
int[] lottoNumbers = {6, 15, 17, 22, 30, 32};

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



Sample program # 1

Write a program using an array that will read in the prices of 10 different items. The program should calculate and display the total price.

Pseudocode:

```
For each item in the array
    READ in the price
    ADD to totalprice
DISPLAY totalprice
```

```java
import java.util.Scanner;
public class PriceArray{

 public static void main(String[] args)
 {
   Scanner keyboardIn = new Scanner(System.in);

   // declare an array to hold ten prices
   double[] prices = new double[10];
   double totalPrice = 0;

   for(int i=0; i < prices.length; i++)
   {
      // read in the Price
      System.out.print("Enter price no. " + (i+1));
      prices[i] = keyboardIn.nextDouble();

      // add to the total price
      totalPrice = totalPrice + prices[i];
   }
   // display total price
   System.out.println("Total price: " + totalPrice);
 }
} // end class
```

Sample program # 2

Write a program that uses an array to store the ages of 4 students. The user should be prompted to enter the ages of all 4 students. The program should then display the current age of each student and the age of each student in 5 years time.

Pseudocode:

```
For each student
    READ in the age

For each item in the array
    Display current age
    Display age in 5 years
```

```java
import java.util.Scanner;
public class StudentAges
{
   public static void main(String [] args)
   {
     // declare array to hold ages
     int[] ages = new int[4];
     Scanner keyboardIn = new Scanner(System.in);

     // read in ages to the array
     for(int i = 0; i < ages.length; i++)
     {
        System.out.print("Enter age of student " + (i+1));
        ages[i] = keyboardIn.nextInt();
     }

     // output ages and their equivalent in five years
     System.out.println("Current Age\t\tAge in 5 years");
     for(int i = 0; i < ages.length; i++)
     {
        System.out.println(ages[i] + "\t\t\t" + (ages[i] + 5));
     }
   }
}
```

Searching an Array

- There are times when a programmer will wish to search an array for a particular value.
- A ***linear*** or ***sequential*** search involves going through the array and checking each element in turn.
- This is not the most efficient way of searching an array, but it is easy to understand and implement.

Searching an Array

Problem:

Find the position of a particular value in an array

Pseudocode:

```
FOR each element in the array
    IF element matches the searchValue
        STORE the position

DISPLAY the position
```

Search an array of values

```
Scanner keyboardIn = new Scanner(System.in);

int[] data = {15, 150, 28, 30, 31, 7};
int searchValue;
int position = -1;  //assume not found

System.out.print("Enter the value to search for: ");
searchValue = keyboardIn.nextInt();
```

continued……

……continued

```
// search the array for the search value
   for(int i = 0; i < data.length; i++)
   {
      if(data[i] == searchValue)   //if match found
      {
        position = i;   //store position
      }
  }
   if(position != -1) //if value of position has changed
   {
     System.out.print("Value found at position: " + position);
   }
   else
   {
     System.out.print("Value NOT found ");
   }
```

Search an array…alternative

```
    // search an array of int values
Scanner keyboardIn = new Scanner(System.in);

int[] data = {15, 150, 28, 30, 31, 7};
int searchValue;
int position = 0;
boolean found = false;  //note use of boolean flag

System.out.print("Enter the value to search for: ");
searchValue = keyboardIn.nextInt();
```

continued……

```
// search the array for the search value
   for(int i = 0; i < data.length; i++)
   {
      if(data[i] == searchValue)  //if match found
      {
        position = i; //store position
        found = true; //remember that it is found
      }
  }
   if(found) //if found is true
   {
     System.out.print("Value found at position: " + position);
   }
   else
   {
     System.out.print("Value NOT found ");
   }
```

Finding the Maximum value

- This involves stepping through the array and testing to see if each value is larger than the current largest value.
- A variable is used to keep track of the largest value
- This value is tested against each element in turn. If the element is larger, it is copied into the variable.

Finding the Maximum value

Problem:
Find the maximum value in an array of integers

Pseudocode:

```
ASSIGN largestYet the value in first element
For each element in the array
    IF element is greater than the largestYet
        ASSIGN value to largestYet

DISPLAY largestYet
```

```
public class FindLargest
{
   public static void main(String[] args)
   {
      int[] data = {15, 150, 28, 30, 31, 7};
       //int[] data = new int[] {15, 150, 28, 30, 31, 7};
      int largest = data[0];

      // search the array for the highest value
      for(int i = 0; i < data.length; i++)
      {
         if(data[i] > largest)
         {
            largest = data[i];
         }
      }

      System.out.print("The largest value is: " +largest);
   }
}
```

Finding the Minimum value

- The technique for finding the smallest value is the same as that used to find the largest.
- Again, a variable is used to hold what is currently the smallest value.
- This value is tested against each element in turn. If the element is smaller, it is copied into the variable.

Counting occurrences/matches

- There are times when you will want to find the number of occurrences of a certain value in an array.
- To count occurrences in an array, check all elements and count the matches until you reach the end of the array
- Use a loop to go through the array, incrementing a counter each time you find a match
  - Can also be used to count occurrences of relative values, e.g. number of students who pass or fail, number of overdrawn bank balances …

```java
import java.util.Scanner;
public class CountingMatches{
   public static void main(String[] args)
   {
       Scanner keyboardIn = new Scanner(System.in);
       int[] data = {12, 13, 12, 26, 12, 2, 34, 12, 0, 34, 13};
       int target, count = 0;
       //get target value
       System.out.print("Enter Search value: " );
       target = keyboardIn.nextInt();

       // loop through the array, counting matches
       for(int i = 0; i < data.length; i++)
       {
          if(target == data[i])
             count++;
       }
       System.out.print(target +" occurs " +count +" times" );

   }
}
```

Parallel Arrays

- Parallel Arrays are two or more arrays in which values with same subscripts/indexes relate to each other
- In this example temp[ ], and rainfall[ ] hold the temperatures and rainfall for the 7 days of the week. temp[0] holds the temperature for day 1 and rainfall[0] holds the rainfall for day 1

temp

rainfall

0	1	2	3	4	5	6

Parallel Arrays

- Display the result of multiplying corresponding elements together
- Could store ***stockLevels*** in one array and ***price*** in another
- Find total cost of each stock item by multiplying corresponding elements

stockLevels

3	10	4	5	100	2	0

price

3.33	1.20	120.00	10	5.5	3	.90
0	1	2	3	4	5	6

Sample program using parallel arrays

```
import java.util.*;

public class Q2Multiply
{
      public static void main(String[] args)
      {
         Scanner keyboardIn = new Scanner(System.in);
      //create 2 arrays
         int[] array1 = new int[5];
         int[] array2 = new int[5];
      //Read values into array1
         System.out.println("Enter 5 integer values:  ");
         for(int i = 0; i<array1.length; i++)
         {
            System.out.print("Number " +(i+1) +": ");
            array1[i] = keyboardIn.nextInt();
         }
```

Continued….

…continued

```
       //Read values into array2
         System.out.println("Enter 5 integer values:  ");
         for(int i = 0; i<array2.length; i++)
         {
            System.out.print("Number " +(i+1) +": ");
            array2[i] = keyboardIn.nextInt();
         }

 //display the result of multiplying corresponding
    //elements together
         for(int i = 0; i<array1.length; i++)
         {
            System.out.print(array1[i]*array2[i] +" ");
         }
      }//end main method
   }//end class
```