

**Learning Outcome:** 4

**Assessment Standard:** Classes & Objects, Creating an array of objects.

### INTRODUCTION

#### What is an array?

An array is the storage of elements of the same data type. We get an array of primitive data types, eg array of int, array of char, array of double, array of float, array of boolean.

We also get an array of non primitive types eg an array of String.

Study the following example.

This is an example of an array of int called numArray.

0	1	2	3	4
27	18	21	16	88

**Notes:** The values 0, 1, 2, 3 and 4 refer to the subscripts of the array. The subscript can also be referred to as an index. The values stored in the array are 27, 18, 21, 16 and 88 respectively. The values stored can also be referred to as the contents of the array.

**NB:** Notice that only a single element can be stored in each memory location of the array above, that is each index can hold onto one value.

#### What if we need to store more than one value having different data types in each index of the array?

Example we need to store details of employees of a company where each employee must have details such as employee number, name, surname, age and salary. Notice that we have five fields that need to be assigned to each employee. The fields are not all of the same data type. We assign all fields to an object, and we create an array of objects where each object can contain one or more field/s of the object.

Can you think of other objects and its likely attributes? Example a car object can have the following fields: make, model, colour and price. A computer object can have the following fields: capacityOfRam, processingSpeed, hardDriveSpace and price.

#### If we need to store more than one attribute in each memory location of an array, we use an array of objects.

#### Question 1:

- 1.1 Create a class called Employees with the following field properties. String fields are code, surname and name. Numeric fields are age and salary.
- 1.2 Write a default and a parameterised constructor to instantiate the Employees objects.
- 1.3 Create relevant Accessor and mutator methods to retrieve and change the state of the field properties.
- 1.4 Write a toString method to return the fields as a String object.
- 1.5 Add the following methods.
  - 1.5.1 A void method called heading that contains all the fields in the heading in columns.
  - 1.5.2 A void method called display to display all fields in Columns.
  - 1.5.3 A typed method called getHighestSalary to return the highest salary.
  - 1.5.4 A typed method called getRecordOfLowestSalary to return the record of the employee with the lowest salary.
  - 1.5.5 A typed method called getEldestEmployee to return the surname and name of the eldest employee.
  - 1.5.6 A typed method called getAverageAge to return the average age of employees.
- 1.6 Create a driver / test class called UseEmployees to enter an unknown number of employee details. Use a sentinel value of "Exit" for code to terminate the input. Create an object to call the heading and display methods of the Employee class. Also display the highest salary, all details of the employee with the lowest salary, the full name of the eldest employee and the average age.

#### Answer:

```
public class Employees
{
    // declare fields
    private String code="";
    private String surname="";
    private String name="";
    private int age=0;
    private double salary=0;

    // default constructor
    public Employees()
    {
    }
}
```



```
// parameterised constructor
Employees(String c,String sn, String n, int ag, double sal)
{
    code = c;
    surname = sn;
    name = n;
    age = ag;
    salary = sal;
}

// accessor methods, you may use only the necessary accessors.
public String getCode()
{
    return code;
}

public String getSurname()
{
    return surname;
}

public String getName()
{
    return name;
}

public int getAge()
{
    return age;
}

public double getSalary()
{
    return salary;
}

// mutator methods, you may use only the relevant mutators.
public void setCode(String c)
{
    code = c;
}

public void setSurname(String s)
{
    surname = s;
}

public void setName(String n)
{
    name = n;
}

public void setAge(int a)
{
    age = a;
}

public void setSalary(double s)
{
    salary = s;
}

Public String toString()
{
    String data = code +getSpaces(code,10)+surname +
    getSpaces(surname,15) +name+getSpaces(name,15) +
    age+getSpaces(""+age,8)+salary;
    return data;
}

public void heading()
{
    System.out.println("Code"+getSpaces("Code",10)+"Surname"+
    getSpaces("Surname",15)+"Name"+getSpaces("Name",15)+"A
    ge"+getSpaces("Age",8)+"Salary");
}

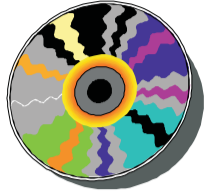
public void display(Employees[]details,int count)
{
    for (int i=0;i<count;i++)
        System.out.println(details[i]);
}

public double getHighestSalary(Employees[]details,int count)
{
    double high =details[0].getSalary();
    for (int i = 1; i< count; i++)
    {
        if (details[i].getSalary() > high)
        {
            high =details[i].getSalary();
        }
    }
    return high;
}

public String getEldestEmployee(Employees[]details,int count)
{
    int high =details[0].getAge();
    String eldest =details[0].getName() +"
    "+details[0].getSurname();
    for (int i =1; i< count; i++)
    {
        if (details[i].getAge() > high)
        {
            high =details[i].getAge();
            eldest =details[i].getName() +" "+details[i].getSurname();
        }
    }
    return eldest;
}
```



```
public double getAverageAge(Employees[]details,int
count)
{
    int sum = 0;
    double ave = 0;
    for (int i = 0; i < count; i++)
    {
        sum +=details[i].getAge();
    }
    ave = sum / count;
    return ave;
}
```



```
public Employees
getRecordOfLowestSalary(Employees[]details,int
count)
{
    Employees tempRec = new Employees();
    double low = details[0].getSalary();
    for (int i=1; i< count;i++)
    {
        if (details[i].getSalary() < low)
        {
            low = details[i].getSalary();
            tempRec = details[i];
        }
    }
    return tempRec;
}
```

```
Employees tempRec = new Employees();
double low = details[0].getSalary();
for (int i=1; i< count;i++)
```

```
if (details[i].getSalary() < low)
```

```
low = details[i].getSalary();
tempRec = details[i];
```

```
return tempRec;
```

```
// helper methods do not have to be public
String getSpaces(String s, int w)
```

```
String spaces="";
for (int i=0;i<w - s.length(); i++)
```

```
spaces = spaces + " ";
```

```
return spaces;
```



#### User class/ Driver class / Test Class

```
import java.io.*;
import java.util.*;
import javax.swing.*;
```

```
public class UseEmployees
```

```
{
    String code,surname,name;
    int n=0,age;
    double salary;
```

```
Employees [] details = new Employees[5];
```

```
Employees my = new Employees();
```

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

```
UseEmployees() {
    code = JOptionPane.showInputDialog("Please
enter code");
    while (!code.equalsIgnoreCase("Exit"))
```

```
{
    surname = JOptionPane.showInputDialog("Please
enter surname");
```

```
name = JOptionPane.showInputDialog("Please enter
name")
```

```
age = Integer.parseInt(JOptionPane.showInputDialog
("Please enter age"));
Salary=Double.parseDouble(JOptionPane.showInput
```

```
Dialog ("Please enter salary"));
details[n] = new Employees(code,surname,name,age,salary);
```

```
n++;
code = JOptionPane.showInputDialog("Please enter code") }
```

```
System.out.println("\t\t\tEmployees Table");
```

```
My.heading();
```

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

```
My.display(details,n);
```

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

```
System.out.println("The highest salary is "+
```

```
My.getHighestSalary(details,n));
```

```
System.out.println("The eldest person is "+
```

```
My.getEldestEmployee(details,n));
```

```
System.out.println("The average age is "+
```

```
my.getAverageAge(details,n));
```

```
System.out.println("The person with the lowest sal is");
```

```
System.out.println(my.getRecordOfLowestSalary(details,n));
```

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

```
{
    new UseEmployees();
}
```

