Guide to simple and practical exercises developed in java

JAVA

Exercises

intNova by Darwin Gómez

FOREWORD

This guide presents a compendium with the most outstanding exercises carried out during my educational stage, simple and practical exercises that point to the staging of a single program, both structural and object oriented.

This is aimed at students in the first semester of programming, which is when teachers teach structured programming and then it is difficult to assimilate the concept of object-oriented programming paradigm. The objective of this guide is not to teach to object-oriented programming but rather I offer you a help of how a structured program would lead to object-oriented programming with simple and practical exercises.

INDEX

Greeting program
Estructured ••••••••••••••••••• 3
POO ••••••••••••••••••• 4
Equal Value Check program
Estructured ••••••••••••••••••• 5
POO ••••••••••••••••••• 6
Week Day program
Estructured ••••••••••••••••••• 7
POO ••••••••••••••••••• 8
Factoring program
Estructured ••••••••••••••••••• 9
POO ••••••••••••••••••• 10
Pow Two program
Estructured ••••••••••••••••••• 11
POO ••••••••••••••••••• 12
Multiplication Table program
Estructured ••••••••••••••••••• 13
POO MVC
-class View ••••••••••••••••••• 14
-class Controller ••••••••••••••••••• 15
-class Model ••••••••••••••••••• 16
-class Main ••••••••••••••••••• 17

Program that allows the user to enter their name, and then shows a greeting with their name in a structured way.

```java
import java.util.Scanner;

public class Greeting {

    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);

        String name = "";
        System.out.print("Enter your name_> ");
        name = sc.nextLine();

        System.out.println("Hello, " + name + ", how are you doing?");

    }

}
```

Now we see that the previous program is very direct, but what if we divide the tasks a little so that our program goes faster and is more organized when debugging.

```java
import java.util.Scanner;

public class Greeting {
    private String name;
    private Scanner sc;

    public void putName() {
        sc = new Scanner(System.in);
        System.out.print("Enter your name_> ");
        name = sc.nextLine();
    }
    public String getName() {
        return name;
    }
    public void showGreeting(String name) {
       System.out.println("Hello, " + name + ", how are you doing?");
    }

    public static void main(String[] args) {
        Greeting gr = new Greeting();
        gr.putName();
        gr.showGreeting(gr.getName());
    }

}
```

Program that allows the user to know if two values entered from the keyboard are the same or different.

```java
import java.util.Scanner;
public class EqualValueCheck {
    public static void main(String[] args) {

        Scanner sc = new Scanner(System.in);
        int firstVal, lastVal;
        System.out.print("Insert first value_> ");
        firstVal = sc.nextInt();
        System.out.print("Insert last value_> ");
        lastVal = sc.nextInt();

        if(firstVal == lastVal) {
             System.out.println(firstVal + " is the same as " + lastVal);
         } else {
             System.out.println(firstVal + " is not the same as " + lastVal);
         }
    }
}
```

It is more comfortable if we assign specific tasks to perform in smaller pieces of code, it will be a more maintainable and scalable program.

```java
import java.util.Scanner;
public class EqualValueCheck {
    private int[] values;
    private Scanner sc;
    public void putValues() {
        sc = new Scanner(System.in);
        values = new int[2];
        for (int = 0; i < values.length; i++) {
            System.out.print("Insert value " + (i + 1) + "_> ");
            values[i] = sc.nextInt();
        }
    }
    public int[] getValues() {
        return values;
    }
    public void showEquality(int[] values) {
        for (int j = 0; j < values.length; j++) {
            if (j > 0) {
                System.out.print((values[0] != values[1])
                 ? (values[j - 1] + " is not the same as " + values[j])
                 : (values[j - 1] + " is the same as " + values[j]));
            }
        }
    }
    public static void main(String[] args) {
        EcualValueCheck evc = new EcualValueCheck();
        evc.putValues();
        evc.showEquality(evc.getValues());
    }
}
```

Program that allows the user to enter a numerical value and shows the corresponding day of the week using the control structure switch case

```java
import java.util.Scanner;
public class WeekDay {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int val;
        System.out.print("Insert a value_> ");
        val = sc.nextInt();
        switch(val) {
            case 1:
                    System.out.print("Monday");
                    break;
            case 2:
                    System.out.print("Tuesday");
                    break;
            case 3:
                    System.out.print("Wednesday");
                    break;
            case 4:
                    System.out.print("Thursday");
                    break;
            case 5:
                    System.out.print("Friday");
                    break;
            case 6:
                    System.out.print("Saturday");
                    break;
            case 7:
                    System.out.print("Sunday");
                    break;
            default:
                    System.out.print("wrong day");
        }
    }
}
```

Now we adapt the previous program using arrangements to make it more dynamic.

```java
import java.util.Scanner;
public class WeekDay {
  private Scanner sc;
  private String[] days;
  private int value;
  public void putDays() {
     days = new String[] {"Monday", "Tuesday",
                         "Wednesday", "Thurday",
                         "friday", "Saturday",
                         "Sunday", "Day wrong"};
     sc = new Scanner(System.in);
     System.out.print("Enter a value now_> ");
     value = sc.nextInt();
  }
  public String[] getDays() {
     return days;
  }
  public int getValue() {
     return value;
  }
  public void showWeekDay(String[] days, int value) {
     String day = "";
     if (value > 8) {
        System.out.print("Blatant error");
     } else {
        for (int i = 0; i < days.length; i++) {
           day = days[value - 1];
        }
        System.out.print(day);
     }
  }
  public static void main(String[] args) {
      WeekDay hw = new WeekDay();
      wd.putDays();
      wd.showWeekDay(wd.getDays(), wd.getValue());
  }
}
```

Program that allows the entry of a value and shows its factorial.

```java
import java.util.Scanner;
public class Factoring {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int val = 1;
        int factor = 1;
        System.out.print("Enter a value_> ");
        val = sc.nextInt();
        System.out.print(val + "! = ");
        while (val > 0) {
            System.out.print(val);
            System.out.print(val > 1 ? " x " : " = ");
            factor = factor * val;
            val--;
        }
            System.out.print(factor);
    }
}
```

Now let's see the same program in object-oriented programming.

```
import java.util.Scanner;
public class Factoring {
    private int val;
    private Scanner sc;
    public void putValue() {
        sc = new Scanner(System.in);
        System.out.print("Enter a value_> ");
        val = sc.nextInt();
    }
    public int getValue() {
        return val;
    }
    public void showFactorial(int value) {
        int factor = 1;
        System.out.print(val + "! = ");
        while (value > 0) {
            System.out.print(value);
            System.out.print(value > 1 ? " x " : " = ");
            factor = factor * value;
            value--;
        }
            System.out.print(factor);
    }
    public static void main(String[] args) {
        Factorize fa = new Factorized();
        fa.putValue();
        fa.showFactorial(fa.getValue);
    }
}
```

Program that allows the entry of a value and calculate its power of two.

```java
import java.util.Scanner;
public class PowTwo {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        int val = 1;
        int double = 2;
        System.out.print("Enter a value_> ");
        val = sc.nextInt();
        if (val > 0) {
            System.out.print(val + " /\ " + double + " = " + (val * val));
        } else {
            System.out.print("Value wrong");
        }
    }
}
```

Now let's see the same program in object-oriented programming.

```java
import java.util.Scanner;
public class PowTwo {
    private int val;
    private Scanner sc;
    public void putValue() {
        sc = new Scanner(System.in);
        System.out.print("Enter a value_> ");
        val = sc.nextInt();
    }
    public int getValue() {
        return val;
    }
    public void showPowTwo(int value) {
        int double = 2;

        if (value > 0) {
            System.out.print(val + " /\ " + double + " = " + (val * val));

        } else {
            System.out.print("Value wrong");
        }
    }
    public static void main(String[] args) {
        PowTwo pt = new PowTwo();
        pt.putValue();
        pt.showFactorial(pt.getValue);
    }
}

```

Now we are going to create a program using the Control.io library developed by intNova and later we make an implementation of the viewer contolador model with the same program.

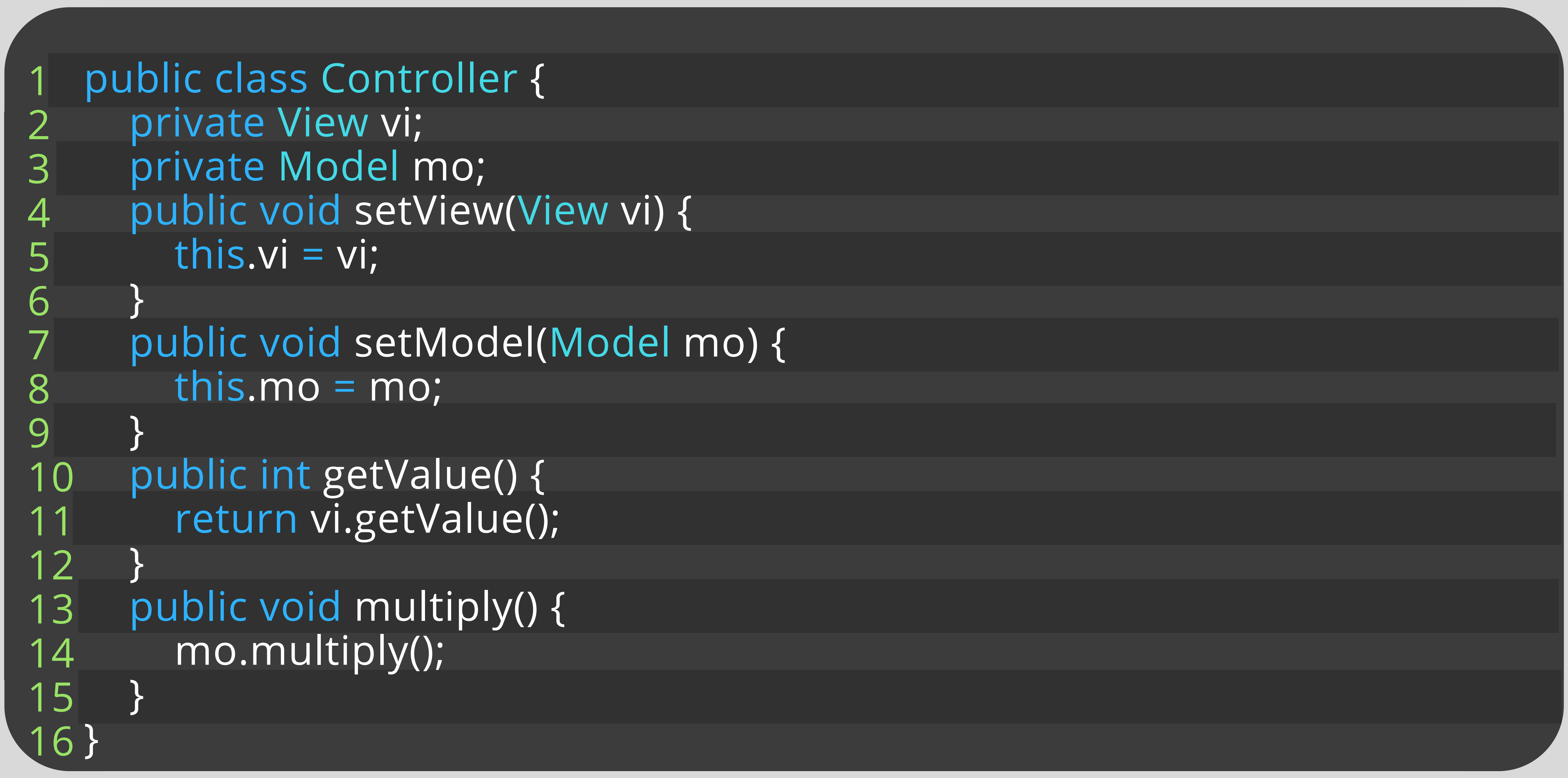
```java
import com.intNova.io.Control;

public class MultiplicationTable {
    public static void main(String[] args) {
        Control.out.write("Enter a value_> ");
        int val = Control.in.readInt();
        int count = 1;
        int mult = 1;
        do {
            mult = val * count;
            Control.out.writeln(val + " x " + count + " = " + mult);
            count++;
        } while (count <= 10);
    }
}
```

Implementation of the program in the model view controller.

-Class View

```java
import com.intNova.io.Control;
public class View {
    private Controller co;
    private int value;
    public void setController(Controller co) {
        this.co = co;
    }
    public void showIn(String msg) {
        Control.out.write(msg);
        value = Control.in.readInt();
    }
    public int getValue() {
        return value;
    }
    public void showOut() {
        co.multiply();
    }
}
```

-Class Controller



```java
public class Controller {
    private View vi;
    private Model mo;
    public void setView(View vi) {
        this.vi = vi;
    }
    public void setModel(Model mo) {
        this.mo = mo;
    }
    public int getValue() {
        return vi.getValue();
    }
    public void multiply() {
        mo.multiply();
    }
}
```

-Class Model

```java
import com.intNova.io.Control;
public class Model {
    private Controller co;
    public void setController(Controller co) {
        this.co = co;
    }
    public int getValue() {
        return co.getValue();
    }
    public void multiply() {
        int val = getValue();
        int count = 1;
        int mult = 1;
        do {
            mult = val * count;
            Control.out.writeln(val + " x " + count + " = " + mult);
            count++;
        } while (count <= 10);
    }
}
```

To finish we create the main class of java to pass the instances by reference and the value travels correctly until its final destination.

-Class Main

```java
public class Main {
    public static void main(String[] args) {
        View vi = new View();
        Controller co = new Controller();
        Model mo = new Model();
        vi.showln("Enter a valuex_> ");
        co.setView(vi);
        mo.setController(co);
        co.setModel(mo);
        vi.setController(co);
        vi.showOut();
    }
}
```

EPILOG

This small guide is dedicated to all those people who, even knowing how to program, find it difficult to enter the world of object-oriented programming.

The exercises described here were assigned by the teachers during my learning process. Any similarity with other programs should be treated as a coincidence, the programming is broad and there are many ways to make a program, but programs like those I propose are difficult to obtain in books or on the Internet since most authors are governed by certain parameters of teaching based on their knowledge and practice, in fact, that was one of the reasons for creating this guide to share my way of thinking and provide knowledge support

Write once, run anywhere

intNova by Darwin Gomez