Programming CS Assignment

2022

Victor Lima Gulart da Silva

Student ID: 2140923

Contents

Introduction

Each task has screenshots of the source code and the output of the running application. Task number 5 is the only exception because it further contains a UML diagram of the Account class.

I kept the notes on the document to a minimum to avoid repeating myself too many times. So instead, I included enough comments on the code to explain the bits and pieces of the application.

There are concise explanations about the code at the beginning of the classes or main methods. These give an overall understanding of the program. All the functions of the classes have information about what they are doing so that their functionality is crystal clear. Also, I added extra notes on some lines to explain in better detail what is happening.

Task 1 - American Flag

For this task I created two possible outcomes:
- One is a flag drawn on the screen with the help of a JavaFx on a window.
- The other, the flag is printed on the terminal.

Terminal Version

1) Program Source

```java
public static void simple_flag(){
    /*
    * Prints the American Flag on the terminal
    * There are 3 types of base strings that are going
    * to be printed on the screen -> these are part of the flag.
    * I also use a for loop to print each line and which depends
    * on whether the line is odd or even and if the loop count is
    * greater than 8
    * */
    String line5stars  = "* * * * * * ==============================";
    String line4stars  = " * * * * *  ==============================";
    String simpleLine = "==========================================";

    for ( int i=0; i<15; i++){
        if (i>8){
            System.out.println(simpleLine);
        }else if (i%2==0){
            System.out.println(line5stars);
        }else{
            System.out.println(line4stars);
        }
    }

}
```

2) Running Application

```
App
 * * * * *  ==============================
* * * * * * ==============================
 * * * * *  ==============================
* * * * * * ==============================
 * * * * *  ==============================
* * * * * * ==============================
 * * * * *  ==============================
* * * * * * ==============================
==============================================
==============================================
==============================================
==============================================
==============================================
==============================================
 --DRAWING THE FLAG--
```

Windowed Version

For this version I used the JavaFx Application Software to create the window and also draw the elements on it.

1) Program Source Code

```java
public class AmericanFlag extends Application {

    /**
     * The American flag GUI version & Terminal version
     *
     *  @author victor
     *  @student id 21409203
     *
     *  To make the window I used JavaFX framework
     *  1) I extend my class to use the Application class
     *  2) overrode the start and stop method of the Application class
     *  3) I had to create a few other methods for creating the flag.
     *      get_flag() --> main function that wraps all the code that
     *                     draws the flag
     *      get_rectangle() --> draws rectangles with a color. it is used to
     *                     draw the red and white straps of the flag, also the
     *                     blue rectangle behind the white stars.
     *      get_stars() --> main function that wraps all the code that
     *                     draws the stars
     *      get_star() --> auxiliary method that is called from inside the
     *                     get_stars() method. It helps extract the complexity
     *                     of the code. This method only draws a single star
     *                     and returns it.
     */

    // Colors to draw the flag
    final Color RED = Color.rgb( i: 247, i1: 17, i2: 17 );
    final Color BLUE = Color.rgb( i: 40, i1: 21, i2: 140 );
    final Color WHITE = Color.rgb( i: 255, i1: 255, i2: 255 );

    // GUI Width and Height
    final int WIDTH = 720 + 40; // 40 of padding
    final int HEIGHT = 286 + 40;
```

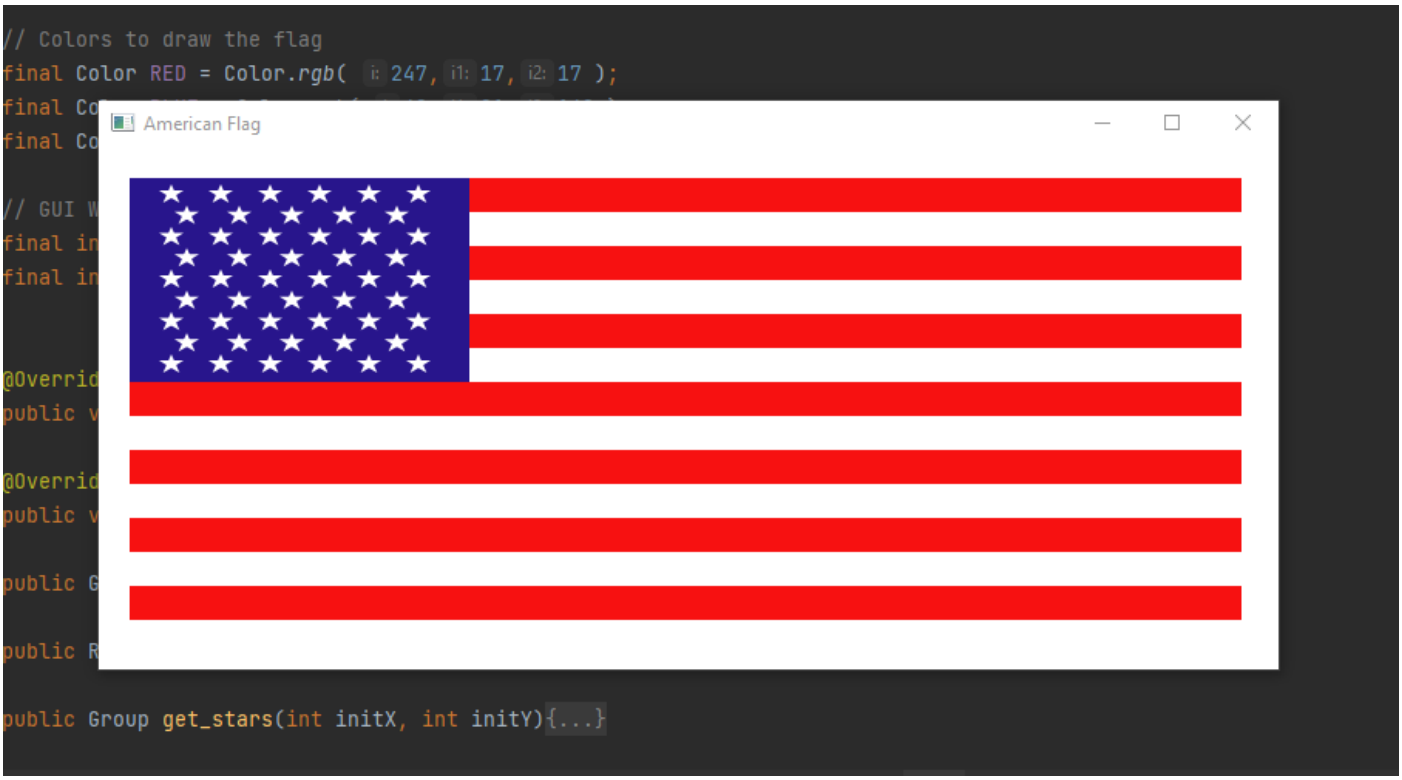
```java
    @Override
    public void start( Stage stage ){
        /*
        * Creates simple window
        * Other methods that create parts of the flag are called from here.
        * */

        // Groups of elements
        Group root = new Group(); // main group which has all the other elements
        Group flag = get_flag( initX: 20, initY: 20 ); // the flag group
        root.getChildren().add( flag ); // adding the flag to the main group

        // Root Scene
        Scene rootScene = new Scene(root,  WIDTH, HEIGHT ); // Main

        // Window Title
        stage.setTitle( "American Flag" );

        // Necessary methods to run GUI
        stage.setScene( rootScene );
        stage.show();
    }

    @Override
    public void stop( ){
        // It is overwritten to end the program properly.
        Platform.exit();
        System.exit( status: 0 );
    }

    public Group get_flag( int initX, int initY ){
        /*
        * Returns the drawing of the flag as a group
        * It receives the initial x and y, the point to start drawing
        */

        Group flag = new Group();
        Group stars;
        Rectangle blue_rect;

        int x = initX;
        int y = initY;

        for ( int i=0; i<13; i++){
            Rectangle rect; // for the background

            // Loop to draw the rectangles red, white and blue
            if (i%2==0){
                if (i<6){
                    // draw a red rectangle from where the blue rectangle ends
                    rect = get_rectangle( x: 220, y, w: WIDTH-20-220, h: 22, bgColor: "red" );
                }else{
                    // draw a full width red rectangle
                    rect = get_rectangle( initX, y, w: WIDTH-40, h: 22, bgColor: "red" );
                }
            }else{
                if (i<6){
                    // draw a white rectangle from where the blue rectangle ends
                    rect = get_rectangle( x: 220, y, w: WIDTH-20-220, h: 22, bgColor: "white" );
                }else{
                    // draw a full width white rectangle
                    rect = get_rectangle( initX, y, w: WIDTH-40, h: 22, bgColor: "white" );
                }
            }
            flag.getChildren().add(rect);
            y += 22;
        }
```

```java
        // Creating the blue rectangle where the stars will be
        blue_rect = get_rectangle(initX, initY, w: 20*11, h: 22* 6, bgColor: "blue");
        flag.getChildren().add(blue_rect);

        // Creating the stars of the flag
        stars = get_stars(initX, initY);
        flag.getChildren().add(stars);

        return flag;
    }

    public Rectangle get_rectangle( int x, int y, int w, int h, String bgColor){
        /*
        * Draws the rectangles for filling the background
        * This method receives the x,y as starting points, the width (w) and height (h),
        * and the color for the rectangle
        * The rectangles are the blue, red and white parts of the flag
        */

        // Draw a rectangle with the correct size and starting x and y.
        Rectangle rect = new Rectangle();
        rect.setX( x );
        rect.setY( y );
        rect.setWidth( w );
        rect.setHeight( h );

        // Giving a color to the rectangle
        if ( "red".equals( bgColor ) ){
            rect.setFill( RED );
        }else if ( "blue".equals( bgColor ) ){
            rect.setFill( BLUE );
        }else if ( "white".equals( bgColor ) ){
            rect.setFill( WHITE );
        }

        return rect; // Returns the finished rectangle
    }

    public Group get_stars(int initX, int initY){
        /*
        *
        * Creates the stars of the blue rectangle
        * Each star is made of 1 arrow and 1 triangle.
        * x and y are the starting points of the blue rectangle
        *
        * It returns all the stars in a group
        * */

        Group stars = new Group();  // Main group for the stars

        double squareX = 20, squareY = 22;
        int counter = 0;
        double X, Y, star_w, star_h;

        X = (double) initX / 2 ;
        Y = (double) initY + 2; // shift 2px down, offset

        // star width and height
        // 4 is the padding
        // 2 is the extra padding for the rows
        star_w = 20 - 4;
        star_h = 18 - 4 - 2;
```

```java
    while (counter<9){
        // this loop runs 9 times because there are 9 rows with 6 and 5 stars each
        List<Group> stars_arr = new ArrayList<Group>();
        double tempX = X / 2 ;

        if ( counter % 2==0 ){
            // creating a 6 stars row
           for (int i=0; i<6; i++) {
               // Create a new star at a new location
               tempX += squareX + 12;
               Group star = get_star(tempX,Y, star_w, star_h);
               stars_arr.add(star);
           }
        }else{
            // creating a 5 stars row
            for (int i=0; i<5; i++){
                // Create a new star at a new location
                tempX += squareX + 14;
                Group star = get_star( X: tempX + (star_w/2),Y, star_w, star_h);
                stars_arr.add(star);
            }
        }

        Y += squareY / 1.6; // increase Y position to draw the next row below the newly created one

        // Add the row of stars to the stars group
        stars.getChildren().addAll(stars_arr);

        // Increment the counter
        ++counter;
    }

    return stars;
}
```

```java
public Group get_star( double X, double Y, double star_w, double star_h ){
    /*
    * Draws a star with the given coordinates, width and height.
    * The star is composed of an arrow and a triangle overlapped
    * */

    Group star = new Group();
    Polygon arrow = new Polygon();
    Polygon triangle = new Polygon();
    Rotate rotation;
    Double[] arr_nodes = new Double[]{};
    Double[] tri_nodes = new Double[]{};

    // Arrow points
    arr_nodes = new Double[]{
            // x , y
            X + 2 + (star_w / 6), Y + 2 + star_h,
            X + 2 + (star_w / 2), Y + 2,
            X + 2 + (star_w * 5/6 ) , Y + 2 + star_h,
            X + 2 + (star_w / 2), Y + 2 + (star_h*4.5/6),
            X + 2 + (star_w / 6), Y + 2 + star_h,
    };

    arrow.setFill(WHITE); // gives a background color to the arrow
    arrow.getPoints().addAll(arr_nodes);

    // Triangle points
    tri_nodes = new Double[]{
            X + 2, Y + 2 + (star_h / 3),
            X + 2 + star_w, Y + 2 + (star_h / 3),
            X + 2 + (star_w/2), Y + 2 + (star_h*4.5/6)
    };
    triangle.getPoints().addAll(tri_nodes);
    triangle.setFill(WHITE); // gives a background color to the triangle

    star.getChildren().addAll(arrow, triangle); // add both shapes to the final group
    return  star;
}
```

2) Running Application



Task 2 - BMI Calculator

I used the JavaFx software platform on this project too.

Program Source Code

```java
public class Main extends Application {

    /**
     * @author: Victor Lima Gulart da Silva
     * @studentID: 21409203
     *
     * The main class
     * BMI formula → weight(kg) / ( height(m)^1 )
     *
     * start() → method from JavaFx Application it must be overridden.
     *          it is the first method to run, inside it there are other methods
     *          calls to create the window and the content inside it.
     * stop() → also from JavaFx Application, it is overridden just to
     *          guarantee proper exit of the application.
     * getUI() → this method is responsible for the creation of the text fields,
     *          labels and the calculate button. and returns everything inside a
     *          FlowPane, which is used for layout.
     * calc_btn() → this actually creates the button and returns it to the getUI().
     *          the button is created separate because it needs extra functionality.
     *          it needs an event handler, a function to handle the click of the user.
     *          so it's separated for better organization.
     * calculate_bmi() → the method get the text fields where the user inserted some
     *          values (weight and height), it gets the values, calculates the bmi and
     *          returns it. (this method is called from the event handler of the button)
     * get_bmi_assess() → this method simply, determines what the bmi value means,
     *          and returns the verdict as a string to later be display to the user.
     *          this method is called from the button event handler, which receives this
     *          string and displays to the user on the user interface (window).
     *
     * */

    float WIDTH = 250;
    float HEIGHT = 300;
    float bmi = 0f;
    TextField weight_input = new TextField();
    TextField height_input = new TextField();

    // Need to be in the class for later access
    GridPane out_box = new GridPane();

    public static void main(String[] args) {
        /**
         * Main function
         * it just launches the JavaFx Application
         *
         * Author: Victor Lima Gulart da Silva
         * Student ID: 21409203
         *
         */
        launch(args);
    }
}
```

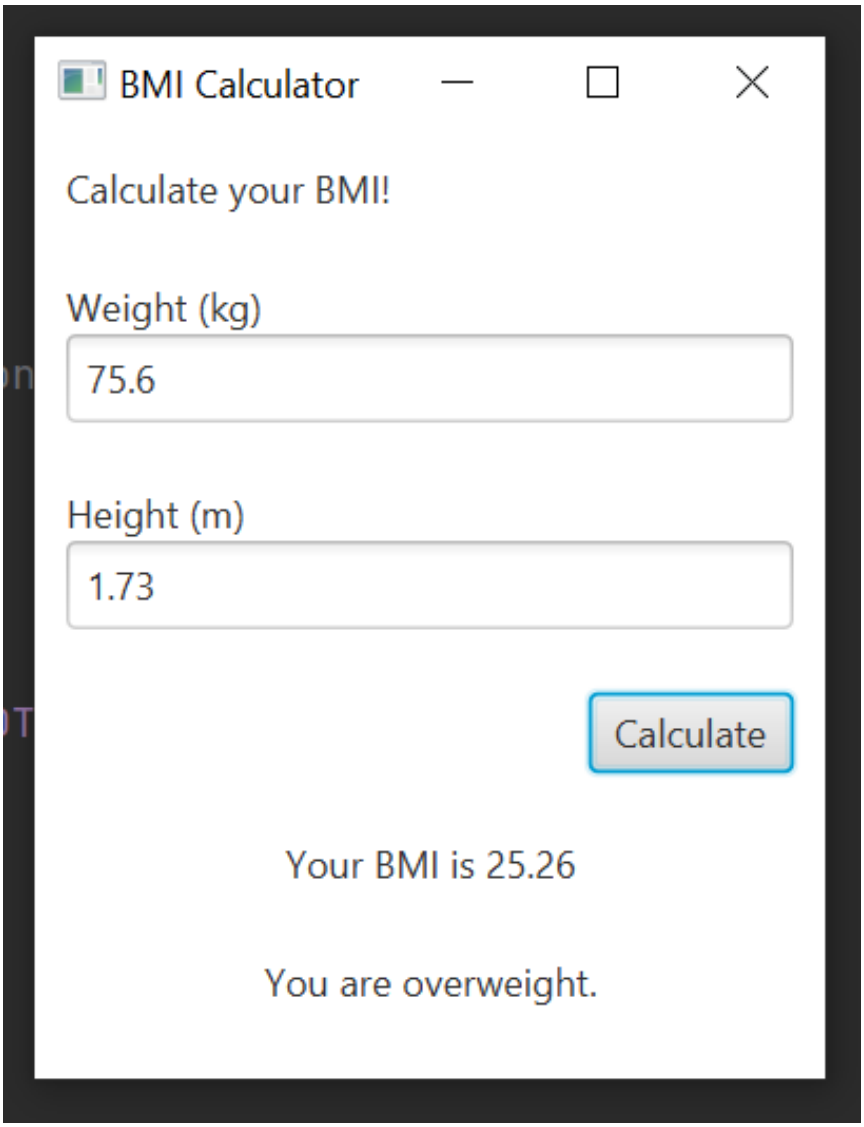
```java
    public Pane getUI(){
        /**
         * This method creates the user interface and its elements,
         * text fields, labels, and the button that
         * the user will see and interact with.
         * */

        FlowPane ui = new FlowPane(); // for the layout - positioning of the elements
        ui.setPrefSize(this.WIDTH, this.HEIGHT);
        ui.setAlignment(Pos.TOP_CENTER);

        // Main container
        VBox vertical = new VBox(); // for the layout
        vertical.setPrefWidth(250d);

        //Vertical Containers for weight and height
        // so that the label and text fields are display in a row
        VBox weight_layout = new VBox();
        VBox height_layout = new VBox();

        // Adding padding to weight and height Vertical Boxes
        weight_layout.setPadding(new Insets( v: 10, v1: 10, v2: 10, v3: 10));
        height_layout.setPadding(new Insets( v: 10, v1: 10, v2: 10, v3: 10));

        // Label - Calculate your BMI!
        Label info = new Label( s: "Calculate your BMI!");
        info.setPadding(new Insets( v: 10, v1: 10, v2: 10, v3: 10));

        // Weight input and label
        Label weight_label = new Label( s: "Weight (kg) ");
        weight_input.setPadding(new Insets( v: 5, v1: 5, v2: 5, v3: 5));

        // Height input and label
        Label height_label = new Label( s: "Height (m)");
        height_input.setPadding(new Insets( v: 5, v1: 5, v2: 5, v3: 5));

        // Appending input and labels
        weight_layout.getChildren().addAll(weight_label, weight_input);
        height_layout.getChildren().addAll(height_label,height_input);

    @Override
    public void start(Stage stage) throws Exception {
        /**
         * This method comes from JavaFX and must be overridden
         * It creates the main window and set a title and main scene.
         * It calls other functions that create the user interface
         * and its functionality.
         * */

        Group root = new Group(); // main group to hold all the other groups or elements
        Pane ui = this.getUI(); // this is the pane UI to hold text fields, labels and buttons
        root.getChildren().addAll(ui); // appending the pane to the root group

        Scene rootScene = new Scene(root, this.WIDTH, this.HEIGHT); // Add the root group to the main scene

        stage.setTitle("BMI Calculator"); // App window title
        stage.setScene(rootScene); // Main scene
        stage.show(); // To finally show the app
    }

    @Override
    public void stop() throws Exception {
        // This method comes from JavaFX.
        super.stop();
    }
```

```java
        // Label to show some text
        Label bmi_text = new Label();
        bmi_text.setAlignment(Pos.CENTER);
        bmi_text.setPrefWidth(250d);
        bmi_text.setPadding(new Insets( v: 10, v1: 10, v2: 10, v3: 10));

        // The BMI calculation will be shown through this label
        Label result = new Label( s: "" );
        result.setAlignment(Pos.CENTER);
        result.setPrefWidth(250d); // preferable width
        result.setPadding(new Insets( v: 10, v1: 10, v2: 10, v3: 10)); // adding padding for better positioning

        // Adding the labels to a GridPane for better positioning
        this.out_box.setConstraints(bmi_text,  i: 0, i1: 0);
        this.out_box.setConstraints(result,  i: 0, i1: 1);
        this.out_box.getChildren().addAll(bmi_text, result);

        // Button to calculate the result
        HBox btn_box = calc_btn(bmi_text, result);

        // Adding the nodes to the vertical
        vertical.getChildren().addAll(info, weight_layout, height_layout, btn_box, out_box);

        // Adding main outer box to main pane
        ui.getChildren().add(vertical);

        return ui;
    }
    public HBox calc_btn(Label bmi, Label result) {
        /**
         * Creates a button and returns it to be added to the UI
         * It also creates the event handler to deal with the click of the user
         * */

        HBox btn_box = new HBox();
        Button calc = new Button( s: "Calculate");

        // Positioning children nodes
        btn_box.setAlignment(Pos.CENTER_RIGHT);
        btn_box.setPadding(new Insets( v: 10, v1: 10, v2: 10, v3: 10));
        btn_box.getChildren().addAll(calc);

        // This part handles with the click
        calc.addEventHandler(MouseEvent.MOUSE_CLICKED, event -> {
            if (event.getButton().equals(MouseButton.PRIMARY)){
                // Calculate the value
                this.bmi = this.calculate_bmi(this.weight_input, this.height_input);

                // Show the value on the app (labels that are already there but had nothing on them)
                bmi.setText("Your BMI is " + BigDecimal.valueOf( this.bmi ).setScale( newScale: 2, RoundingMode.HALF_UP) );
                result.setText(get_bmi_assess());
            }
        });

        return btn_box;
    }
    public Float calculate_bmi(TextField w, TextField h){
        /**
         * Receives both text fields that have the
         * weight and height values.
         * Then it calculates the user's BMI and returns it
         * */
        float result = 0f;
        float weight = Float.parseFloat(w.getText());
        float height = Float.parseFloat(h.getText());
        result = weight / (height*height);
        return result;
    }
```

```java
public String get_bmi_assess(){
    /**
     * Analyses the bmi that was already calculated and
     * returns the verdict as a string.
     * */

    if (this.bmi < 18.5){
        return "You are underweight.";
    }else if (this.bmi < 24.9){
        return  "You look healthy. Keep up the good work!";
    }else if (this.bmi < 29.9){
        return "You are overweight.";
    }else{
        return "You are obese.";
    }
}
```

Running Application

The user can insert the weight in kilograms and the height in meters. To calculate the BMI, the user must press the button “Calculate”, which will compute the value and display the final assessment of the BMI.



Task 3 - Online Store

This program runs on the terminal.
→ It has a simple menu that shows to the user 3 options - products, checkout and exit.
→ It runs until the user chooses either chekcout or exit.

The user can see the catalogue of products, choose a product to be added to the cart and then checkout. Meaning it will show the total amount of the purchase.

Program Source Code

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

    /**
     *
     * @author: Victor Lima Gulart da Silva
     * @studentID: 21409203
     *
     * The loop starts by printing the menu of options to the user, which are
     * products, checkout or exit
     *
     * When the user chooses "products", the list of products is shown along
     * with their prices. The user can choose a product and a quantity.
     *
     * Later the user may go to checkout.
     * Or exit the program
     * */

    Scanner input = new Scanner(System.in); // to record the user input
    int idx; // it is used to select a product on the arrays
    int choice; // this is the user's choice for selecting the menu options
    int qty; // the quantity of a product to be added to the cart
    double amount; // the final amount for the checkout

    double[] cart_qty = {0,0,0,0}; // products quantities
    int[] cart_prods = {-1, -1, -1, -1}; // products idx
    int product;

    // Arrays - Catalogue of products and prices
    String[] products = {"Santa Chocolate", "Milk Chocolate", "Dark Chocolate", "White Chocolate"};
    double[] prices = {1.2, 0.99, 2.39, 1.99};


    print( text: "=========================", newLine: true);
    print( text: "== Welcome to GoShop ==", newLine: true);
    print( text: "=========================", newLine: true);

public static void menu(){
    /**
    * It prints the current options for the user.
    * This is the main menu of the application.
    * */

    print( text: "### MENU ###", newLine: true);
    print( text: "1- Products", newLine: true);
    print( text: "2- Checkout", newLine: true);
    print( text: "-1 for Exiting", newLine: true);
    print( text: "############", newLine: true);
}
```

```java
        // Program loop
        while (true) {
            System.out.println("");
            menu(); // print the main menu
            System.out.print(">> ");
            choice = input.nextInt();

            switch (choice){
                case 1:
                    // get a product
                    product = printCatalogue(products, prices); // print catalogue

                    // go to next iteration, user did not choose a product
                    if (product == -1){
                        continue;
                    }

                    // get quantity
                    print( text: "How many would you like? ",  newLine: true);
                    print( text: ">> ",  newLine: false);
                    qty = input.nextInt();

                    // go to next iteration, user did not add qty
                    if (qty==0){
                       continue;
                    }

                    // add product to cart
                    idx = inCart(cart_prods, product);

                    // Add the item to the cart if it is not there yet
                    if (idx == -1){
                        cart_prods = addItemToCart(cart_prods, product);
                        print( text: "Item added to the cart.",  newLine: true);
                    }

                    // get the index location of the product in the array
                    idx = inCart(cart_prods, product);
                    cart_qty = addQtyToCart(idx, cart_qty, qty);

                    break;
                case 2:
                    // Calculates the total amount of the items in the cart
                    amount = checkout(prices, cart_qty);
                    print( text: "",  newLine: true);
                    print( text: "",  newLine: true);
                    double scale = Math.pow(10, 2); // this is for rounding purposes
                    print( text: "The total is: £" + Math.round(amount * scale)/scale,  newLine: true);
                    print( text: "Thanks for coming.",  newLine: true);

                    return; // Exits the program

                case -1:
                    // Exist the program
                    print( text: "",  newLine: true);
                    print( text: "",  newLine: true);
                    print( text: "Thank you for coming. See you next time.",  newLine: true);
                    return;
                default:
                    print( text: "Choose something.",  newLine: true);
                    break;
            }

            print( text: "",  newLine: true);
            print( text: "",  newLine: true);
        }
    }
}
```

This print() method was created just to shorten the System.out.println() and System.out.print() in just one function.

```java
public static void print(String text, boolean newLine){
    /**
    * It is just a wrapper method to print a message that is passed as
    * argument on the console with System.out.println() or System.out.print()
    *
    * Made it just to shorten the whole process of printing. xD
    */

    if (newLine == true){
        System.out.println(text);
    }else{
        System.out.print(text);
    }
}

public static int printCatalogue(String[] products, double[] prices){
    /**
    * It prints the catalogue and waits for the user to choose a product
    * If the products exists it returns the product idx location on the products array
    * If it does not find the product it returns an empty string
    * */

    Scanner input = new Scanner(System.in);
    int choice;

    print( text: "What would you like? ",  newLine: true);
    // printing each product with the price
    for (int idx=0; idx<products.length; idx++) {
        print( text: idx + 1 + "- " + products[idx] + "→ £" + prices[idx],  newLine: true);
    }
    print( text: ">> ",  newLine: false);
    choice = input.nextInt(); // waits for the user choice

    if (choice > products.length || choice < 1){
        print( text: "",  newLine: true);
        print( text: "I'm sorry, we couldn't find that product.",  newLine: true);

        return -1; // just an empty string, the program will check and start the program loop again
    }

    return choice-1; // returns the product name
}
```

```java
public static int inCart(int[] cart_prods, int product){
    /**
     * Searches if the product (which is an idx that works as an ID for the product)
     * is already in the cart. if it is, it returns the index location of the users
     * cart is returned, otherwise it returns -1.
     */

    for (int i=0; i<cart_prods.length; i++){
        if (cart_prods[i] == product){
            return i;
        }
    }
    return -1;
}
```

```java
public static int[] addItemToCart(int[] cart_prods, int product){
    /**
     * Adds the product to the user's cart and returns the changed cart
     * Receives the user's cart that holds the idx values of products that
     * the user wants, also the idx of the new product to add to the cart
     * */
    for (int i=0; i<cart_prods.length; i++) {
        if (cart_prods[i] == -1){
            cart_prods[i] = product ;
            return cart_prods;
        }
    }
    return cart_prods;
}

public static double[] addQtyToCart(int idx, double[] cart_qty, int qty){
    /**
    * This method increases the quantity of a certain product in the cart
    * */
    cart_qty[idx] += qty;
    return cart_qty;
}

public static double checkout(double[] prices, double[] cart){
    /**
     * It receives the user's cart and the prices
    * It calculates the total amount of the items in the cart
    * */
    double total = 0;

    for (int i=0; i<cart.length; i++){
        total += cart[i] * prices[i];
    }
    return total; // returns the total
}
```

Running Application

The App Menu

```
=========================
= Welcome to GoShop =
=========================

### MENU ###
1- Products
2- Checkout
-1 for Exiting
############
>>
```

Selecting option 1.
It shows the catalogue.

```
>> 1
What would you like?
1- Santa Chocolate→ £1.2
2- Milk Chocolate→ £0.99
3- Dark Chocolate→ £2.39
4- White Chocolate→ £1.99
>> |
```

Selecting the 1st item and the quantity.

```
What would you like?
1- Santa Chocolate→ £1.2
2- Milk Chocolate→ £0.99
3- Dark Chocolate→ £2.39
4- White Chocolate→ £1.99
>> 1
How many would you like?
>> 2
Item added to the cart.
```

Selecting a 2nd item and the quantity.

```
What would you like?
1- Santa Chocolate→ £1.2
2- Milk Chocolate→ £0.99
3- Dark Chocolate→ £2.39
4- White Chocolate→ £1.99
>> 2
How many would you like?
>> 1
Item added to the cart.
```

At checkout it shows the total amount of the purchase and exits the program.

```
### MENU ###
1- Products
2- Checkout
-1 for Exiting
############
>> 2


The total is: £3.39
Thanks for coming.
```

Selecting the exit option.

```
═══════════════════════
═ Welcome to GoShop ═
═══════════════════════

### MENU ###
1- Products
2- Checkout
-1 for Exiting
############
>> -1


Thank you for coming.
See you next time.
```

Task 4 - Election

This program runs on the terminal.
When the election app is run, it prints a message at the beginning telling that the voting system has started. It explains to the user that it has to enter a candidate name to vote. And how to exit the

Program Source Code

```java
public static void getWinner(ArrayList<String> names, ArrayList<Integer> votes){
    /**
    * It finds the winner of the election and prints it out.
    * */
    String name = "";
    int max = 0;

    for (int i=0; i<names.size(); i++){
        if ( votes.get(i)>votes.get(max)){
            max = i;
        }
    }
    System.out.println("==========================================");
    System.out.println("The winner is " + names.get(max));
    System.out.println("==========================================");
}
```

```java
public static void printoutVotes(ArrayList<String> names, ArrayList<Integer> votes){
    /**
     * Receives the list of candidates and the list with the sum of votes for each
     * candidate.
     * It prints all the users and the number of votes that each one had.
     */
    System.out.println("\n\nTotal votes for each candidate:");
    for (int i=0; i< names.size(); i++) {
        System.out.print(names.get(i) + " had ");
        if (votes.get(i)==1) {
            System.out.println(votes.get(i) + " vote.");
        }else{
            System.out.println(votes.get(i) + " votes.");
        }
    }
}
```

```java
    public static void main(String[] args) {
        /**
         *
         * @author: Victor Lima Gulart da Silva
         * @studentID: 21409203
         *
         * The program is a short infinite while loop.
         * That asks for the name of candidate and adds 1 to the count
         * for the chosen person, until the program is exited.
         * At the end of the program it checks the votes for the winner.
         * It prints a list of the votes for each candidate and reveals
         * the winner.
        * */

        ArrayList<String> names = new ArrayList<String>(); // holds the all the candidates' names
        ArrayList<Integer> count = new ArrayList<Integer>(); // holds the votes counting
        Scanner input = new Scanner(System.in);
        String vote = "";

        System.out.println("==========================================");
        System.out.println("==         Voting system initiated.         ==");
        System.out.println("==========================================");
        System.out.println("");

        while(true){
            System.out.println("");
            System.out.println("");
            System.out.println("To vote enter the candidate name.");
            System.out.println("To exit → type Q or -1 or press Enter ");
            System.out.print("Choose your candidate: ");
            vote = input.nextLine();

            // check if the candidate is in the array names
            if (vote.toLowerCase().equals("q") || vote.equals("-1") || vote.equals("") ){
                break;
            }
            else if(names.contains(vote)) {
                // if the candidate is already registered
                // increment the vote count by 1
                for (int i = 0; i < names.size(); i++) {
                    if (names.get(i).equals(vote)) {
                        count.set(i, count.get(i) + 1);
                    }
                }
            }else{
                // if the candidate is not registered
                // add the name to the names list and
                // initialize the count for him/her
                names.add(vote);
                count.add(1);
            }
        } // End of while loop

        // Exits the program if the arrayLists are empty
        if (names.size() == 0){
            System.out.println("There were no votes recorded.");
            return;
        }

        // Print out each candidate votes
        printoutVotes(names, count);

        // Get the winner
        getWinner(names, count);
    }
}
```

Running Application

```
==============================================
==         Voting system initiated.         ==
==============================================



To vote enter the candidate name.
To exit -> type Q or -1 or press Enter
Choose your candidate:


To vote enter the candidate name.
To exit -> type Q or -1 or press Enter
Choose your candidate: John


To vote enter the candidate name.
To exit -> type Q or -1 or press Enter
Choose your candidate: Doe


To vote enter the candidate name.
To exit -> type Q or -1 or press Enter
Choose your candidate:


To vote enter the candidate name.
To exit -> type Q or -1 or press Enter
Choose your candidate: q


Total votes for each candidate:
John had 3 votes.
Doe had 2 votes.
================================
The winner is John
================================

Process finished with exit code 0
```

Task 5 - Bank Account

This program runs on the terminal.

I created the Account class with the specified functionality. It contains all the methods and variables stated.

Program Source Code

```java
public class Main {

    public static void main(String[] args) {
        /**
         * @author: Victor Lima Gulart da Silva
         * @studentID: 21409203
         * */

        // Setting the Interest Rate
        Account.setAnnualInterestRate(0.045);

        // Creating an Account Instance
        Account myAccount = new Account( id: 1122,  balance: 20000);

        // Adding and removing money from the account
        myAccount.withdraw( amount: 2500);
        myAccount.deposit( amount: 3000);

        // Printing to the terminal
        double scale = Math.pow(10,2);
        System.out.println("The current balance is " + myAccount.getBalance());
        System.out.println("The annual interest rate is → " + Account.getMonthlyInterestRate() * 100 + "%");
        System.out.println("The monthly interest rate is → " + Account.getMonthlyInterestRate() * 100 + "%");
        System.out.println("The monthly interest → " + myAccount.getMonthlyInterest());
        System.out.println("The account was created on " + myAccount.getDateCreated());
    }
}

class Account {
    private int id;
    private double balance;
    private static double annualInterestRate = 0; // current interest rate
    private static double monthlyInterestRate = 0; // current interest rate
    private Date dateCreated;

    Account(){
        // Creates a default account if no argument is passed
        this.id = 0;
        this.balance = 0;
        this.setDateCreated();
    }

    Account(int id, double balance){
        // Creates an account
        this.id = id;
        this.balance = balance;
        this.setDateCreated();
    }

    // Date Setter and Getter
    public void setDateCreated(){
        this.dateCreated = new Date();
    }

    public Date getDateCreated(){
        return this.dateCreated;
    }
```

```java
    // Set the Annual Interest Rate
    public static void setAnnualInterestRate(double interestRate){
        Account.setMonthlyInterestRate(interestRate);
        annualInterestRate = interestRate;
    }

    // Get the Annual Interest Rate
    public static Double getAnnualInterestRate(){
        return annualInterestRate;
    }

    public static void setMonthlyInterestRate(double interestRate){
        // Converts the annual interest rate to monthly and saves it
        // CONVERT LOGIC
        Account.monthlyInterestRate = interestRate / 12;
    }

    // Get the monthly Interest Rate
    public static double getMonthlyInterestRate(){
        return  Account.monthlyInterestRate;
    }

    // Get monthly interest
    public String getMonthlyInterest(){
        double interest = this.balance * Account.monthlyInterestRate;
        double scale = Math.pow(10,2);
        String result;

        result = "£"+ Math.round(interest*scale)/scale;

        return result;
    }

    // ID setter and getter
    public void setId(int id){
        this.id = id;
    }

    public int getId(){
        return this.id;
    }

    // Balance setter and getter
    public void setBalance(double balance){
        this.balance = balance;
    }

    public String getBalance(){
        double scale = Math.pow(10,2);
        return "£" + Math.round(this.balance*scale)/scale;
    }

    public void withdraw(double amount){
        // Withdraws money from the account
        double result = this.balance - amount;
        this.setBalance(result); // update the balance in the account
    }

    public void deposit(double amount){
        // Deposits an amount into the account
        double temp = this.balance + amount;
        double result = temp;
        this.setBalance(result);
    }

}
```

Running Application

```
The current balance is £20500.0
The annual interest rate is → 4.5%
The monthly interest rate is → 0.375%
The monthly interest → £76.88
The account was created on Fri Jan 21 07:38:23 GMT 2022

Process finished with exit code 0
```

UML Diagram