Lista 01 - POO

Nome: Mateus Henrique Vieira Figueiredo
Matrícula: 4707

1.

```
import java.util.Scanner;

class Exercicio01
{
   public static void main(String[] args)
   {
       Scanner scanner = new Scanner(System.in);

       try
       {
           System.out.println("Type the current year:");
           int currYear = scanner.nextInt();

           System.out.println("Type the year of your birth:");
           int birthYear = scanner.nextInt();

           if (birthYear > currYear)
           {
               System.out.println("Invalid value. Current year cannot
be greater than birth year.");
           }
           else
           {
               System.out.println("Your current age is: " + (currYear
- birthYear) + " years old.");
           }
       }
       catch (NumberFormatException e)
       {
           System.out.println("Invalid value, not a number.");
       }
   }

}
```

2.

```java
import java.util.Scanner;

class Exercicio02
{
   public static void main(String[] args)
   {
       Scanner scanner = new Scanner(System.in);

       try
       {
           System.out.println("Type the numerator: ");
           int numerator = scanner.nextInt();

           System.out.println("Type the denominator: ");
           int denominator = scanner.nextInt();

           if (denominator == 0)
           {
               System.out.println("Warning: Cannot divide by 0.");
               return;
           }

           System.out.println("Result = " + (numerator / denominator));
       }
       catch (NumberFormatException e)
       {
           System.out.println("Invalid input, it must be a number.");
       }
   }

}
```

3.

```java
import java.util.Scanner;

class Exercicio03
{
   public static void main(String[] args)
   {
       Scanner scanner = new Scanner(System.in);

       try
       {
           System.out.print("Type your salary: R$");
           float salary = scanner.nextFloat();

           System.out.print("Type the loan desired value: R$");
           float desiredLoan = scanner.nextFloat();

           if (desiredLoan > (salary * 0.3))
           {
               System.out.println("\nYou ARE NOT able to get the loan.");
           }
           else
           {
               System.out.println("\nYou ARE able to get the loan.");
           }
       }
       catch (NumberFormatException e)
       {
           System.out.println("\nInvalid input, it must be a number.");
       }
   }

}
```

4.

```
import java.util.Map;
import java.util.Scanner;
import java.util.TreeMap;

class Exercicio04
{
   public static void main(String[] args)
   {
        Scanner scanner = new Scanner(System.in);

        Map<String, Integer> months = new
TreeMap<>(String.CASE_INSENSITIVE_ORDER);

        months.put("Janeiro", 1);
        months.put("Fevereiro", 2);
        months.put("Março", 3);
        months.put("Abril", 4);
        months.put("Maio", 5);
        months.put("Junho", 6);
        months.put("Julho", 7);
        months.put("Agosto", 8);
        months.put("Setembro", 9);
        months.put("Outubro", 10);
        months.put("Novembro", 11);
        months.put("Dezembro", 12);

        System.out.println("Type the month: ");

        String userInput = scanner.nextLine();

        if (months.containsKey(userInput))
        {
            System.out.println("\"" + userInput + "\": " +
months.get(userInput));
        }
        else
        {
            System.out.println("Month not found.");
        }
   }

}
```

5.

```
import java.util.Scanner;

class Exercicio05
{
   public static void main(String[] args)
   {
       Scanner scanner = new Scanner(System.in);

       Month[] months = {
               new Month("Janeiro", 31),
               new Month("Fevereiro", 28),
               new Month("Março", 31),
               new Month("Abril", 30),
               new Month("Maio", 31),
               new Month("Junho", 30),
               new Month("Julho", 31),
               new Month("Agosto", 31),
               new Month("Setembro", 30),
               new Month("Outubro", 31),
               new Month("Novembro", 30),
               new Month("Dezembro", 31),
       };

       try
       {
           System.out.println("Day: ");
           int day = scanner.nextInt();

           System.out.println("Month: ");
           int month = scanner.nextInt();

           if (month > 0 && month < 13)
           {
               if (day <= months[month - 1].days)
               {
                   System.out.println("The date is valid, month: " +
months[month - 1].name + ".");
               }
               else
               {
                   System.out.println("Error: Invalid day");
               }
           }
           else
           {
               System.out.println("Error: Invalid month");
           }
       }
       catch (NumberFormatException e)
       {
```

```
            System.out.println("Invalid value.");
        }

    }

    private static class Month
    {
        String name;
        int days;

        public Month(String name, int days)
        {
            this.name = name;
            this.days = days;
        }

    }

}
```

6.

```
import java.util.Scanner;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

class Exercicio06
{
    public static void main(String[] args)
    {
        Scanner scanner = new Scanner(System.in);
        Pattern p = Pattern.compile("(\\d{2}):(\\d{2}):(\\d{2})");

        System.out.println("Type the initial time in the format
(HH:MM:SS)");
        String unmodifiedInitialTime = scanner.nextLine();

        Matcher m = p.matcher(unmodifiedInitialTime);

        int[] initialTime = new int[3];

        if (!m.find())
        {
            System.out.println("Invalid format");
            return;
        }

        initialTime[0] = Integer.parseInt(m.group(1));
        initialTime[1] = Integer.parseInt(m.group(2));
```

```
        initialTime[2] = Integer.parseInt(m.group(3));

        System.out.println("Type the final time in the format (HH:MM:SS)");
        String unmodifiedFinalTime = scanner.nextLine();

        m = p.matcher(unmodifiedFinalTime);

        int[] finalTime = new int[3];

        if (!m.find())
        {
            System.out.println("Invalid format");
            return;
        }

        finalTime[0] = Integer.parseInt(m.group(1));
        finalTime[1] = Integer.parseInt(m.group(2));
        finalTime[2] = Integer.parseInt(m.group(3));

        System.out.println("Difference in seconds = " +
                (getTimeInSeconds(finalTime) -
getTimeInSeconds(initialTime)) +
                " seconds.");
    }

    private static int getTimeInSeconds(int[] timestamp)
    {
        return timestamp[0] * 3600 + timestamp[1] * 60 + timestamp[2];
    }
}
```

7.

```java
import java.util.Arrays;
import java.util.Scanner;

class Exercicio07
{
   public static void main(String[] args)
   {
       Scanner scanner = new Scanner(System.in);

       try
       {
           System.out.println("Type a number between 0 and 1000");
           int qtyOfNumbers = scanner.nextInt();

           if (qtyOfNumbers < 0 || qtyOfNumbers > 1000)
           {
               System.out.println("Invalid number.");
               return;
           }

           int[] numbers = new int[qtyOfNumbers];

           for (int i = 0; i < qtyOfNumbers; i++)
           {
               try
               {
                   numbers[i] = scanner.nextInt();
               }
               catch (NumberFormatException e)
               {
                   System.out.println("Not a number.");
               }
           }

           if (numbers.length == 0)
           {
               return;
           }

           long evenNumTotal = Arrays.stream(numbers).filter(value ->
(value % 2) == 0).count();

           System.out.println("Smallest value: " +
Arrays.stream(numbers).min().getAsInt() +
                   "\nArithmetic mean: " + (Arrays.stream(numbers).sum() /
qtyOfNumbers) +
                   "\nBiggest value: " +
Arrays.stream(numbers).max().getAsInt() +
                   "\nQuantity of even numbers: " + evenNumTotal +
                   "\nQuantity of odd numbers: " + (qtyOfNumbers -
evenNumTotal));
```

```
        }
        catch (NumberFormatException e)
        {
            System.out.println("Invalid value.");
        }
    }

}
```

8.

```
import java.util.ArrayList;
import java.util.Scanner;

class Exercicio08
{
    public static void main(String[] args)
    {
        Scanner scanner = new Scanner(System.in);

        ArrayList<Integer> numbers = new ArrayList<>();

        while (true)
        {
            System.out.println("Type a number or 'fim' word");

            String userInput = scanner.nextLine();

            if (userInput.equalsIgnoreCase("fim"))
            {
                break;
            }

            try
            {
                numbers.add(Integer.parseInt(userInput));
            }
            catch (NumberFormatException e)
            {
                System.out.println("Invalid value.");
            }
        }

        System.out.println("\nNumbers sorted");
        numbers.stream().sorted().forEach(System.out::println);
    }

}
```

9.

```java
import java.util.Scanner;

class Exercicio09
{
   public static void main(String[] args)
   {
       Scanner scanner = new Scanner(System.in);

       System.out.println("Type the command: ");
       String[] command = scanner.nextLine().split(" ");

       try
       {
           switch (command[0].toUpperCase())
           {
               case "MULTIPLICA" ->
                       System.out.println("RESPOSTA: " +
(Float.parseFloat(command[1]) * Float.parseFloat(command[3])));
               case "DIVIDE" ->
               {
                   float denominator = Float.parseFloat(command[3]);

                   if (denominator == 0)
                   {
                       System.out.println("Cannot divide by 0.");
                       break;
                   }

                   System.out.println("RESPOSTA: " +
(Float.parseFloat(command[1]) / denominator));
               }
               case "SOMA" ->
                       System.out.println("RESPOSTA: " +
(Float.parseFloat(command[1]) + Float.parseFloat(command[3])));
               case "SUBTRAI" ->
                       System.out.println("RESPOSTA: " +
(Float.parseFloat(command[1]) - Float.parseFloat(command[3])));
               default -> System.out.println("Invalid operation.");
           }
       }
       catch (NumberFormatException e)
       {
           System.out.println("Invalid formatting.");
       }
   }

}
```

10.

```java
import java.util.Random;
import java.util.Scanner;

class Exercicio10
{
   public static void main(String[] args)
   {
        Scanner scanner = new Scanner(System.in);

        int randomNumber = new Random().nextInt(100) + 1;
        int numberOfTries = 0;

        while (true)
        {
            System.out.println("\nGuess the number: ");

            try
            {
                int userNumber = scanner.nextInt();

                if (userNumber < randomNumber)
                {
                    System.out.println("Try a bigger number.");
                }
                else if (userNumber > randomNumber)
                {
                    System.out.println("Try a smaller number.");
                }
                else
                {
                    System.out.println("Nice one, congratulations!! Number
of tries: " + numberOfTries);
                    break;
                }

                numberOfTries++;
            }
            catch (NumberFormatException e)
            {
                System.out.println("Input must be a number.");
            }
        }
   }

}
```

11.

```java
import java.io.IOException;
import java.nio.file.FileSystems;
import java.nio.file.Files;
import java.nio.file.Path;
import java.util.ArrayList;
import java.util.List;
import java.util.regex.Matcher;
import java.util.regex.Pattern;

class Exercicio11
{
   public static void main(String[] args)
   {
       // Format: Name|Genre|Weight|Age|Height
       //      (String) (Char) (Float) (Int) (Float)
       Pattern p = Pattern.compile("(.*)\\|(.*)\\|(.*)\\|(.*)\\|(.*)");

       ArrayList<Person> people = new ArrayList<>();

       try
       {
           Path dataPath =
FileSystems.getDefault().getPath("src/data.txt");

           List<String> lines = Files.readAllLines(dataPath);

           for (String line : lines)
           {
               Matcher m = p.matcher(line);

               if (m.find())
               {
                   people.add(new Person(m.group(1),
                           m.group(2).charAt(0),
                           Float.parseFloat(m.group(3)),
                           Integer.parseInt(m.group(4)),
                           Float.parseFloat(m.group(5))));
               }

           }
       }
       catch (IOException e)
       {
           e.printStackTrace();
       }

       if (people.isEmpty())
       {
           return;
       }
```

```
        System.out.println("Number of patients = " + people.size() +
                "\nMale age mean = " +
                (people.stream()
                        .filter(person -> person.genre == 'M')
                        .map(person -> person.age)
                        .reduce(0, Integer::sum) / people.size()) + " y/o" +
                "\nNumber of women with height between 1.60 and 1.70 and
weight above 70kg = " +
                people.stream()
                        .filter(person -> person.height >= 1.60 &&
person.height < 1.70 && person.weight > 70 && person.genre == 'F')
                        .count() +
                "\nNumber of people between 18 and 25 y/o = " +
                people.stream()
                        .filter(person -> person.age > 18 && person.age <
25)
                        .count() +
                "\nOldest patient's name = " +
                people.stream()
                        .reduce((p1, p2) -> p1.age > p2.age ? p1 : p2)
                        .get().name +
                "\nShortest woman's name = " +
                (people.stream()
                        .filter(person -> person.genre == 'F')
                        .reduce((p1, p2) -> p1.height < p2.height ? p1 : p2)
                        .map(person -> person.name)
                        .orElse(null))
        );
    }

    private static class Person
    {
        String name;
        char genre;
        float weight;
        int age;
        float height;

        public Person(String name, char genre, float weight, int age, float
height)
        {
            this.name = name;
            this.genre = genre;
            this.weight = weight;
            this.age = age;
            this.height = height;
        }

    }

}
```