Exercises - Unit 5
Control structures: selection

Group I1E

Year 2017/2018

1. Supose the following part of code, where `x` is an `int` var and `c` is a `char` var, both of them correctly initialised:

```
if (x<0  && c=='x') System.out.println("Case 1");
else if  (x<0 && c!='x') System.out.println("Case 2");
else if  (x>=0 && c=='y') System.out.println("Case 3");
else if  (x>=0 && c!='y') System.out.println("Case 4");
```

   Rewrite the previous code using the following structure, in a way that positioning correctly the conditions and the `System.out.println()` instructions, for any input `x` and `c`, both code have the same behaviour:

```
if ( x < 0 )


else
```

2. Let `x` and `y` be two integer vars, and suppose the following conditional instructions:

   (a)
```
if (x==0)
      if (y==0) System.out.println("x and y are 0");
      else System.out.println("Only x is 0");
else
      if (y==0) System.out.println("Only y is 0");
      else System.out.println("x and y are not 0");
```

   (b)
```
if (x==0 && y==0)
       System.out.println("x and y are 0");
else if (x==0 && y!=0)
         System.out.println("Only x is 0");
else if (x!=0 && y==0)
         System.out.println("Only y is 0");
else System.out.println("x and y are not 0");
```

   Answer to:

   (a) How many comparisons between integers are done in a and b in all the four cases (x=y=0, x=0 and y≠0, y=0 and x≠0, x≠0 and y≠0)

   (b) According to the previous results, which of the two conditionals is more appropriate in order to minimise the number of conditions to be performed?

3. What is shown on the screen when executing the following code?

```
switch(2){
     case 1: System.out.println(1); break;
     case 2: System.out.println(2);
     case 3: System.out.println(3); break;
     default: System.out.println(4);
}
```

4. What is the output of the following code when `firstOp`, of type `int`, is 1? And when it is 2?

```
switch (firstOp + 1) {
      case 1: System.out.print("Salad ");
            break;
      case 2: System.out.print("Paella ");
            break;
      case 3: System.out.print("Fish ");
      case 4: System.out.print("Ice cream ");
            break;
      default: System.out.print("Enjoy your meal");
}
```

5. Write a Java program class with a `main` method that reads three integer values `a`, `b`, and `c` and implements different solutions for the following case analysis, using shortcut operators and conditional instructions:

   `a` $>$ `b` $\rightarrow$ `true`

   `a` $<$ `b` $\rightarrow$ `false`

   `a == b and a` $>$ `c` $\rightarrow$ `true`

   `a == b and a` $<$ `c` $\rightarrow$ `false`

   `a == b and a == c` $\rightarrow$ `false`

6. Write a Java method for the `Circle` class given in previous units that shows on the screen if the center is situated on:

   - The first quadrant ($x > 0$, $y > 0$)
   - The second quadrant ($x < 0$, $y > 0$)
   - The third quadrant ($x < 0$, $y < 0$)
   - The fourth quadrant ($x > 0$, $y > 0$)
   - The abscissa axis ($x \neq 0$, $y = 0$)
   - The ordinate axis ($x = 0$, $y \neq 0$)
   - The origin of coordinates ($x = 0$, $y = 0$)

7. Write a Java program class that allows to calculate the qualitative grade from a numerical grade in the range [0-10]. The correspondences are:

   - [9-10]: A
   - [7-9[: B
   - [5-7[: C
   - <5: F

   The program must ask for the numerical grade and, in case it is valid, show the qualitative grade. If the numerical grade is not correct, an error message must be printed out.

8. Write a Java program class that calculates the cost of a menu in a fast food restaurant. Basic cost of the menu is 5.99 euros. The following factors modify the value:

   - Menu size: normal (same value), large (+15%), extra-large (+25%), children (-10%)
   - Extra items: chips (+1 euro), garlic bread (+0.5 euros), extra sauce (+0.5 euros)
   - Dessert: +1.5 euros

   Write a Java program class that asks the costumer for the menu size, if s/he wants each of the extra items and the dessert, and calculates and prints out the final cost of the menu.

9. Write a Java datatype class `Date`, with three integer attributes that represent a correct day, month and year. Write a method that returns `true` if the date is a holyday and `false` in the other case. Only the following holydays must be considered: January 1st and 6th, May 1st, October 12th, November 1st, and December 25th. WARNING: it is mandatory to use a `switch` conditional instruction to solve this problem.

10. Write a Java method for the `Date` class that returns `true` if it is a leap year (366 days) and `false` in other case. A year is a leap year if it is divisible by 4 and not by 100, or it is divisible by 400.

11. Write a Java method for the `Date` class that returns the number of days of the month. You can use methods implemented in previous exercises.

12. Write a Java method for the `Date` class that checks whether the date is correct or not, taking into account leap years. You can use methods implemented in previous exercises.

13. Write a Java method for the `Date` class that given another `Date` object as parameter, checks whether the current date (referenced by the `this` reference) is previous to the other one (given as parameter), supposing they are correct.

14. Write a Java method for the `Date` class that returns to what zodiac sign (Aries, Taurus, Gemini, ...) corresponds the date.

15. Write a Java method for the `Date` class that returns the ordinal for the current day (1st, 2nd, 3rd, 4th, ..., 20th, 21st, 22nd, 23rd, 24th, ..., 30th, 31st).

16. Write a Java method for the `Date` class that returns a new `Date` object with the date of the next day, supposing the current date is correct. You can use methods implemented in previous exercises.

17. Write a Java `Triangle` class that defines a triangle by three pairs of coordinates (real numbers) that form the vertex of a triangle. Implement the constructor method that receives the six coordinates. Implement a method that returns the length of the largest side of the triangle.

18. Write a method for the `Triangle` class that determines if the triangle is equilateral (all sides of the same length), isosceles (two sides with the same length, not the other), or scalene (all three sides differents); the method must return a `String` with values "Equilateral", "Isosceles", or "Scalene".

19. Write a method for the `Triangle` class that returns the distance of the vertex closest to the origin of coordinates (0,0) to that origin.

20. Write a method for the `Triangle` class that returns if it is a rectangle triangle or not (clue: use Pythagoras theorem).

21. Write a Java program class whose `main` method determines the lowest number of euro bank notes and coins for a given euro amount (optimal change). For example, for 1755.45 euros the optimal change is three notes of 500 euros, one of 200, one of 50, one of 5, two coins of 20 cents and one of 5 cents.

22. Write a Java datatype class that represents data for a worker with two attributes that store the number of worked hours during the week and the wage per hour. Write the method that returns the weekly pay. It must be considered that extra hours (those that exceed 40) must be paid with an extra of 50%.

23. Consider the following case: "In a company, to calculate the payment for the holidays the following process is considered: a person with less than a year in the company has right to two days for each month in the company; in other case, it must be at least 28 days. For a director which is less than 35 and has been more than 3 years in the company, 2 days are added; in case is less than 45 and has been more than 5 years in the company, 4 days are added.". Write the Java datatype class that stores the data for an employee: number of months in the company (`mic`), age of the worker (`age`), and if it is director or not (`dir`). Write the method that returns the number of days of paid holidays. Write a Java program class that ask for the data of an employee object and shows the number of days of paid holidays. Complete the program class using verifications such as age lower than 65 or greater than 18, etc.

24. Write a Java program class whose `main` method calculates the amount to be paid in a driving school; the amount depends on the type of license (A, B, C, or D) and the number of driving lessons. The inscriptions cost: A 150 euros, B 325 euros, C 520 euros, D 610 euros. The driving lessons cost: A 15 euros, B 21 euros, C 36 euros, D 50 euros.

25. Write a Java program class to determine the amount of ECTS credits that you can take in a year. You must ask the amount of ECTS you got inscribed last year, the amount of ECTS you passed, and the standard number of ECTS per year. Now you must follow these rules:

    - If you passed less than 6 ECTS and they are less than you got inscribed, you will not be allowed to inscribe in next year independently of any other condition (0 ECTS).

- If you passed less than 70% of ECTS that you got inscribed, you will be allowed to take only the amount of ECTS equal to the difference between those you inscribed and those you passed (e.g., if you got inscribed in 60 ECTS and you passed only 36 ECTS, you will be allowed to get inscribed in 24 ECTS).
- If you passed more than 70% of ECTS that you got inscribed but not all, you will be allowed to take only the amount of ECTS you passed (e.g., if you got inscribed in 60 ECTS and you passed 48 ECTS, you will be allowed to get inscribed in 48 ECTS).
- If you passed all the ECTS you got inscribed, you will be allowed to take the standard amount of ECTS (this includes the case of first inscription, where previously inscribed ECTS are 0 and passed ECTS are 0).

At the end of the program you must inform the user of how many ECTS will be allowed to get inscribed.

26. Let be the following game: a player (A) chooses a number between 1 and 16; another player (B) can ask four time questions such as "Is the number 13?", and player A will answer "equal", "lower" or "higher", depending on the number proposed by B is equal to, lower than or higher than the chosen one by A.

    (a) Describe a winning strategy

    (b) Write a Java program class in whose `main` method the game is developed and the user is player A and the computer is player B

27. It is needed to write a code to calculate the square root of any real number inputed by the user. Since that operation is not defined for negative numbers, it is necessary to manage that possible error and avoid the program stop. Write different Java solutions for this problem, by using: shortcut operators, conditional instructions, ternary operator, etc.

28. Write a Java program class whose `main` method implements the "rock, paper, scissors" game. The "rock, paper, scissors" game is played by two players. In a turn of the game, each player says simultaneosly to the other one one of the following words, "rock", "paper", and "scissors", and then:

    - When a player says "rock" and the other "scissors", the player that says "rock" is the winner (rock can break scissors)
    - When a player says "scissors" and the other "paper", the player that says "scissors" is the winner (scissors cuts paper)
    - When a player says "paper" and the other "rock", the player that says "paper" is the winner (paper envelopes rock)
    - When both players say the same word, there is a draw

    In the program, the computer is one of the players, whereas the user is the other one. When executing, the program must:

    (a) Select randomly one of the three elements ("rock", "paper", "scissors")
    (b) Ask the user for one of the elements; the input should be a string, and when the input is not one of the three possibilities the program will finish after saying that the word was not recognised
    (c) Depending on the random selection on the first step and the value given by the user, following the rules of the game, the program will write the winner or the draw result.

29. The spoof game (`http://en.wikipedia.org/wiki/Spoof_(game)`) is a classic game played by friends when they are deciding which one is paying the drinks. Each player may keep in his/her closed hand 0, 1, 2, or 3 sticks or little coins. Each player says how many sticks and coins are in total for all the players, and not repeated values are admisible. Implement a Java program class that simulates this game between two automatic players and a human player. The program must generate randomly the number of coins for the two automatic players and ask the human user for the number of coins. Then the program must generate the hypothesis on the total number of coins by following these rules:

    - The first automatic player will say its number of coins plus a random value between 2 and 4 (both included).
    - The second automatic player will say the number of the first player divided by two plus its number of coins; if the value is the same than the first player, it will randomly add or substract 1 to the value.
    - The human player will enter its hypothesis.

    The winner is the one that gets closer to the total number of coins. If there is a draw, the program must say "It is a draw! Play again!", without saying the involved players. If there is a winner, the program must print out which player is. Any incorrect input will cause the program finish with the message "Don't cheat! You'll pay this round of drinks, moron!".

30. The problem is obtaining the general solution of a second degree equation

$$ax^2 + bx + c = 0$$

where a, b, and c, coefficients of the equation, are any real number and x is the variable whose value is unknown and that can be a real or complex number. The solution of the equation depends on the value of the coefficients, and the following cases can be distingued:

- If $a = 0$ (there is no second order term), then:
  - If $b = 0$, depending on c, the equation presents infinite solutions ($c = 0$) or no solution ($c \neq 0$)
  - If $b \neq 0$, it is a first degree equation with solution $-\frac{c}{b}$
- If $a \neq 0$, the equation can be solved by using the following formula:

$$\frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

  where $(b^2 - 4ac)$ is called *discriminant*; in this case:
  - If the discriminant is 0, there is a single solution and it is a real number
  - If the discriminant is positive, there are two solutions and both are real numbers
  - If the discriminant is negative, there are two solutions and both are complex numbers

To solve the general problem by using a Java program, the following strategy can be followed:

(a) Ask the user for the coefficients and get them from the keyboard

(b) Depending on the coefficients, make the following decisions:
- Equation without solution
- Equation with infinite solutions
- First degree equation with real solution
- Second degree equation with:
  - A single real solution
  - Two real solutions
  - Two complex solutions

It is recommended to follow the case analysis described above.

(c) Write on the screen the warning message or the solution depending on the situation.

All the coefficients a, b, and c, and other possible numbers (such as the descriminant) are real numbers and will be kept during the execution in the appropriate variables, e.g., of `double` datatype. When the solutions are complex numbers, since the Java language lacks of a datatype to represent these numbers, and an alternative representation must be adopted. The most convenient is the cartesian form, where a complex number is represented by a pair of real (`double`) variables, one for the real part and the other for the imaginary part. In this representation, if n is any real negative number, the complex number $\sqrt{n}$ is represented by $\sqrt{|n|}i$, e.g., the complex value $\sqrt{-16}$ is represented by $\sqrt{|-16|}i = 4i$.