Unit 1: Algorithm

Autor: Guillermo Gómez

Contents

Textos Marea Verde

Prior knowledge

Activity: Summarize your general knowledge on this topic.

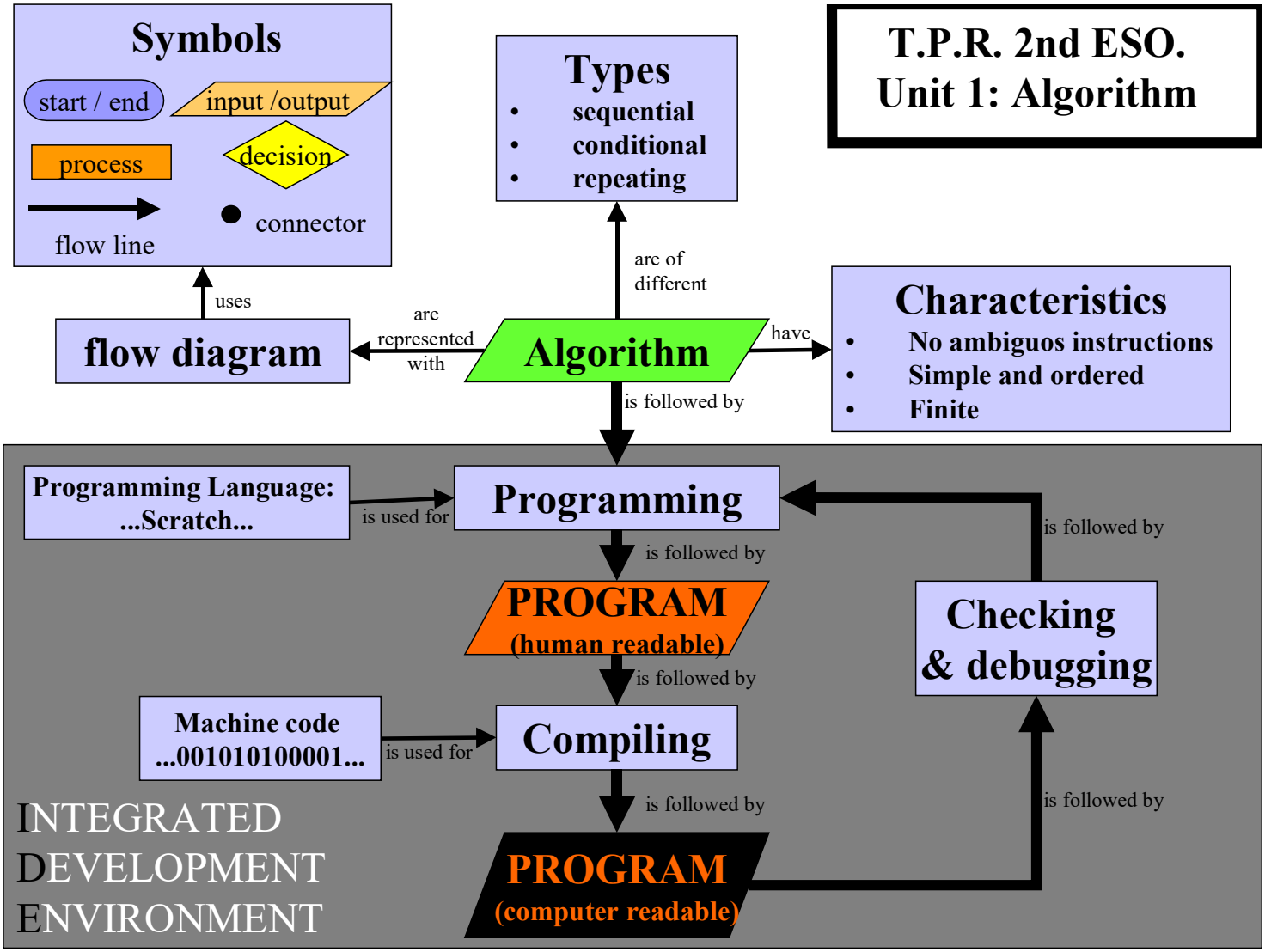
Keywords

Activity: Copy following keywords, understand their meaning and translate them into English.

algoritmo	diagrama de flujo	proceso
codificación	entrada / salida de información	secuencia
decisión	iteración	terminal

Mindmap of the unit

Activity: Analize and try to understand following mindmap



1.1. Technological algorithm

DEFINITION:

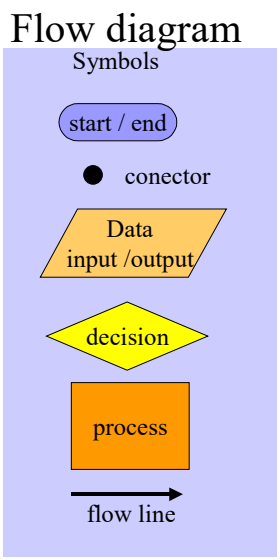
An algorithm is a series of instructions or ordered steps for performing an activity or resolving a problem.

Characteristics of an algorithm:

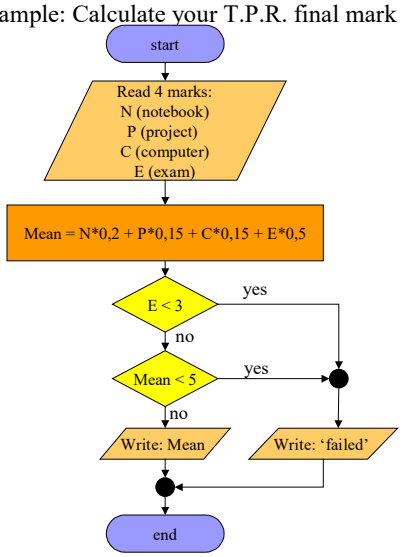
- Instructions with no ambiguity
- Finite (it has an end)
- Simple and ordered steps

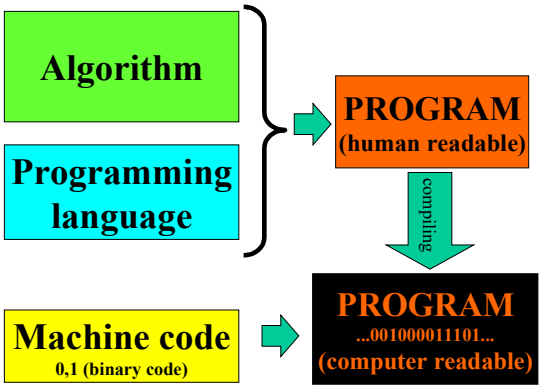
Graphical representation of an algorithm:

Algorithm can be represented graphically in a flow diagram (also called flowchart).



Example: Calculate your T.P.R. final mark





Program:

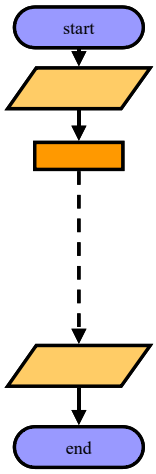
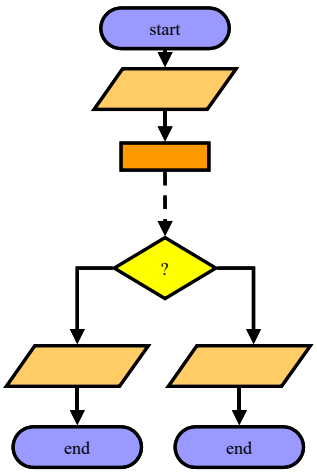
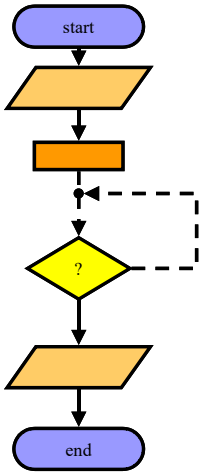
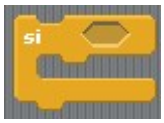
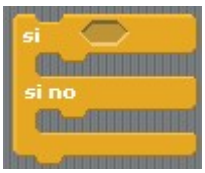
A program is the translation of an algorithm into a programming language. To be executed by a computer , the program must be compiled into machine code (binary code: ...00100101....)

Activity: Copy following exercises and solve them in your notebook

1) Design one of following algorithms: a) How to prepare an exam or b) how a doctor tries to heal a patient.
2) Design an algorithm you follow before crossing across a street with a traffic light.
3) Last year you learnt to solve problems working with the "technological process" (also called "project approach"). Remember each step and represent them in a flow diagram. Do you think that the "technological process" is an algorithm? Why? Why not?
4) Do you think that the mindmap at the beginning of this unit is an algorithm? Why? Why not?

1.2. Types of algorithms

Algorithm can be classified into sequential, conditional (selective) or iterative (repeating) algorithms.

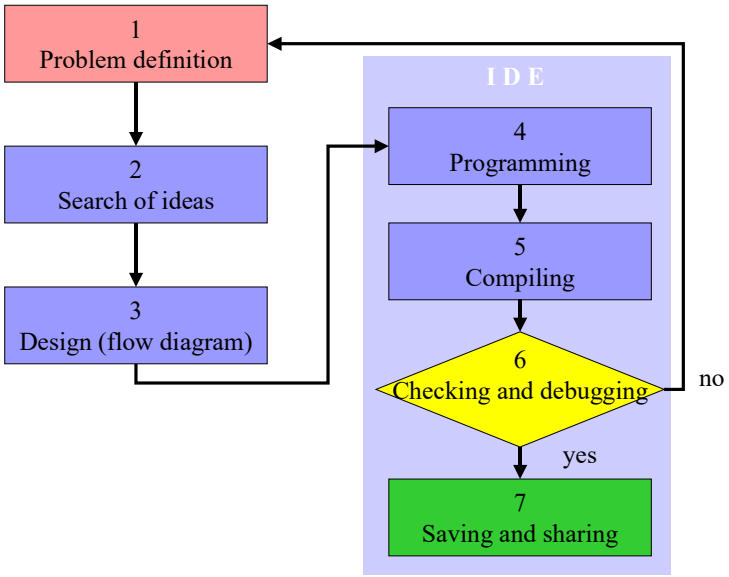
	Type of algorithm		
	Sequential	**Conditional (selective)**	**Iterative (repeating)**
Instructions are executed...	... in the same order as they appear	... depending on whether or not a condition is met	... in loops (or repetitions)
Flowchart aspect			
Blocks used with Scratch:		• if.....then  • if....then / else 	• repeat....  • repeat until ....  • forever....  • forever if .... 

Activity: Copy following exercises and solve them in your notebook

5) Invent and represent a sequential algorithm.
6) Invent and represent a conditional algorithm.
7) Invent and represent a iterative algorithm.

1.3. Programming.

The process of programming follows itself an algorithm, as you can recognize in next flow diagram.

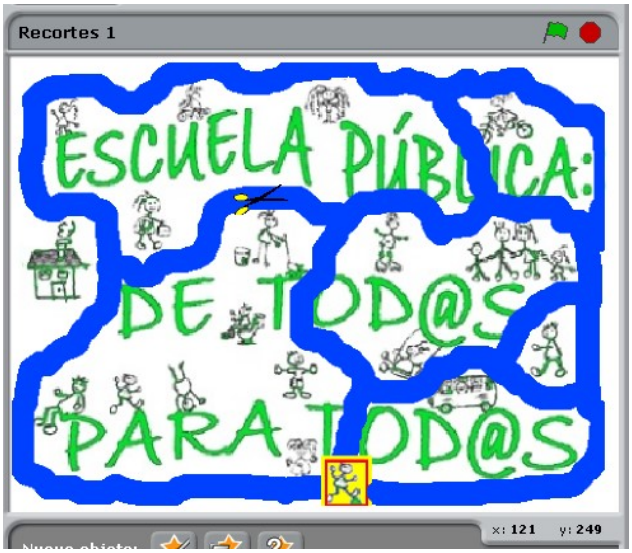


An **Integrated Developmet Environment** (IDE) is a program that allows to:

1. **program**
2. **compile**
3. **run** the program
4. **save** the program

In the computer room we have already worked with "Scratch", which is a free IDE. It`s time to program again!

Example 1 (game)



1)**Problem definition:** It's about programming a game similar to the classic pacman: using the arrow keys to move, you have to guide a scissors within a maze to reach a rioter.

1.1) The scissors moves forward as long it is in contact with the maze's path

1.2) Whenever you push an arrow key, the scissors points in the direction you push

1.3) When the scissors reaches a rioter, a new rioter appears in another part.

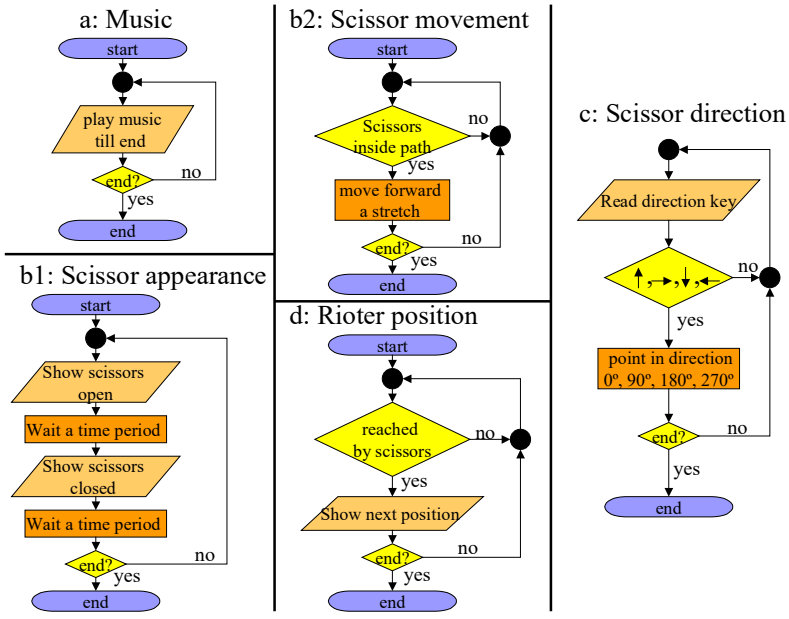
2) **Search for ideas**:

2.1) Stage: Design or import a backdrop that you like and add a continuous music.

2.2) Sprites: a) scissors, b) the maze with the rioter

2.3) Subroutines: a) start the continuous music, b) to put the scissors in the starting point and make it move if it is in contact with maze path, c) to rotate the scissors with the keyboard, d) to change the rioter's position if touched by the scissors

3) **Design**: Look at the flow diagrams.



4) **Programming** (IDE)**:**

4.1) Stage: Import the background and the music, and program it to play continuosly (subroutine a)

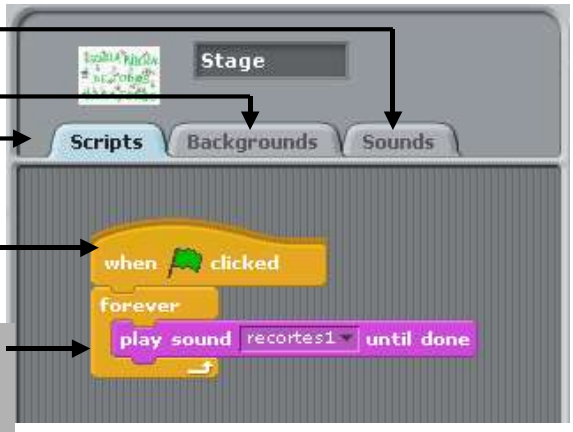
click to import the music

click to import the background

click to program

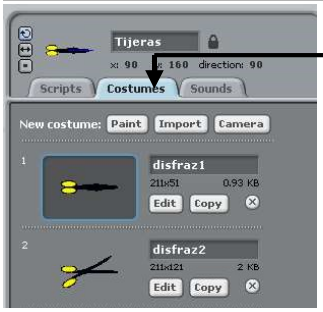
Use the *when green flag is clicked* event to make play the music when the program starts

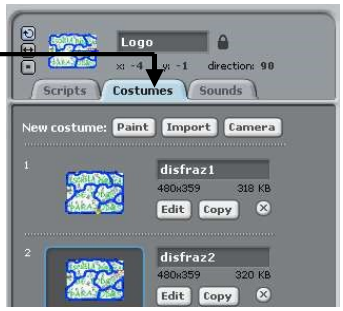
Use the control block *forever* to create a loop (part of a program that repeats itself continuosly)



4.2) Sprites: Design two costumes for the scissors, and at least two costumes of rioters appearing in different positions of the maze

click to design the different costumes for the sprites

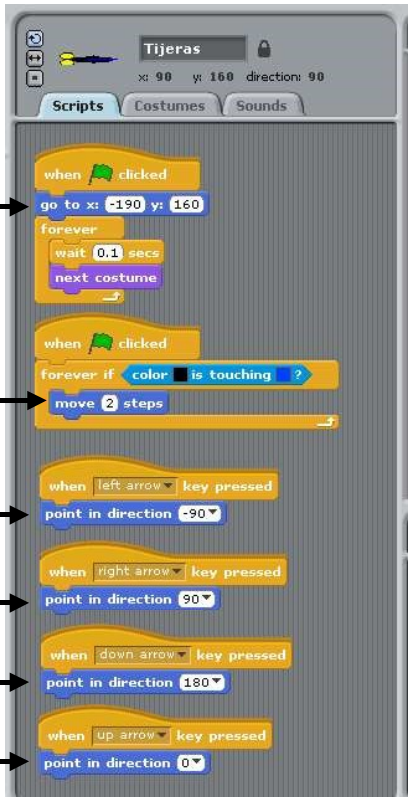


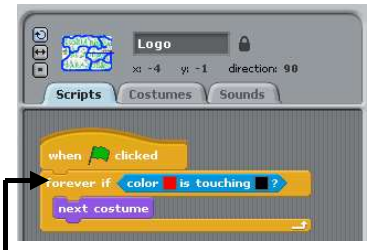


4.3) Subroutines for the sprites: program subroutine b and c for the scissors and subroutine d for the maze with the rioter.

Use the motion blocks *go to* and *move* to put the scissors in the starting point and make it move if it is in contact with maze path

Use the motion blocks *point in direction* to rotate the scissors with the keyboard





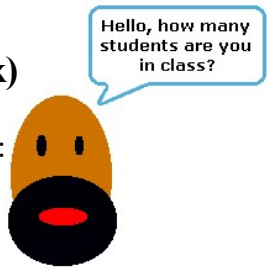
Use the control block *forever if* to change the position of the rioter (red color) each time it is touched by the scissors (black color)

5) **Compiling** (IDE).

6) **Checking and debugging** (IDE): Test, correct the errors and check that the program meets the requirements.

7) **Saving and sharing** (IDE). Don't forget to shave the final program and if you want to share it you can do it through the IDE.

Example 2 (working with variables: calculate your T.P.R. final mark)



1)**Problem definition:** It's about calculating the mean mark according to the formula:

1.1) Mean = 20% Notebook + 15% Project + 15% Computer + 50% Exam

1.2) You will calculate the mean as many times as students there are.

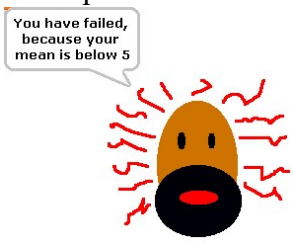
1.3) The student will fail if : Mean < 5 **and / or** Exam < 3

2) **Search for ideas**:

2.1) Stage: Design or import a backdrop.

2.2) Sprites: Design a sprite. It may vary its disguise depending on the output.

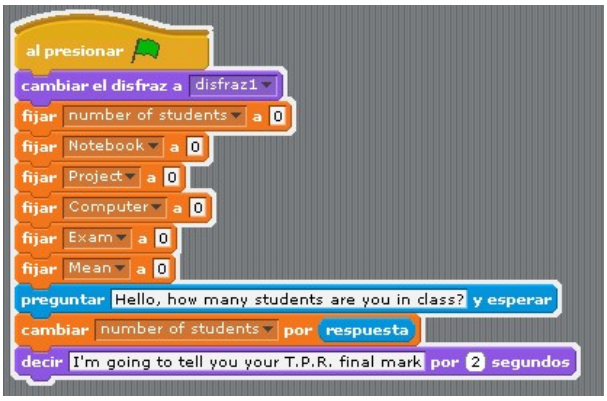
2.3) Dialog: Define the questions for the inputs and the answers for the outputs.

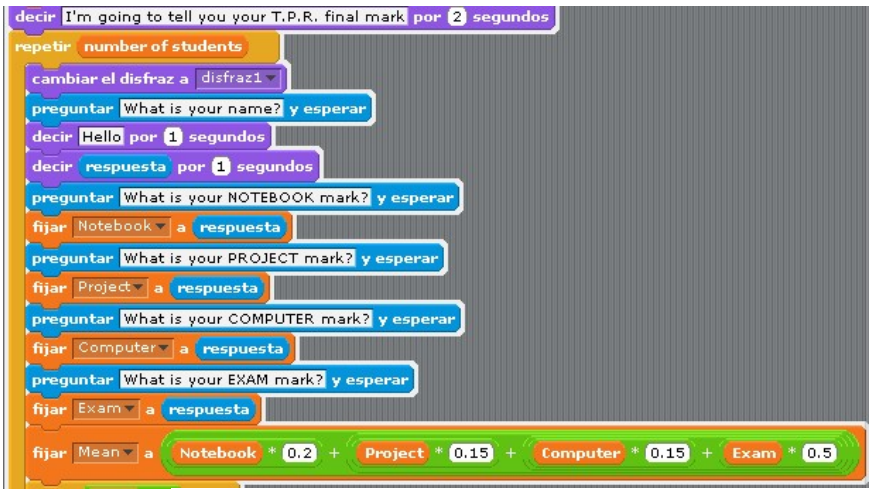


3) **Design**: Look at the flow diagram of page 3.

4) **Programming** (IDE):

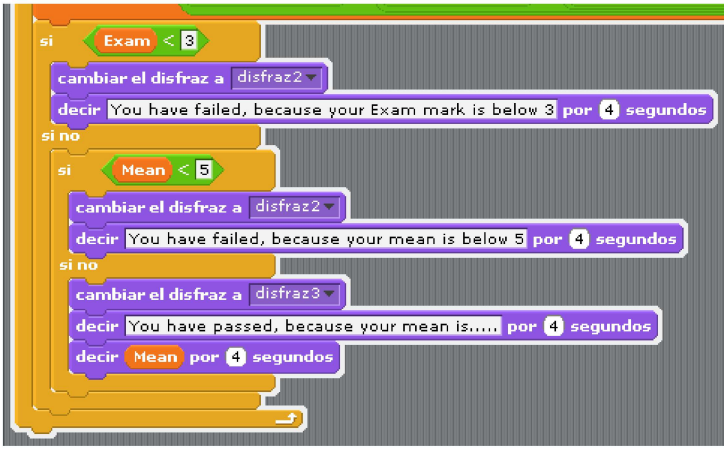
4.1) After designing the sprite, lets start defining the 6 variables ("Number of students", "Notebook", "Project", "Computer", "Exam", "Mean") and set them equal to 0. Ask for the number of students and save the input as "Number of students".





4.2) Then starts the iterative algorithm for each student. Ask the four marks and calculate the mean.

4.3) Within the previous iteration there are two conditional algorithms: the output depends on the exam mark (exam < 3 ?) and the mean value (mean < 5 ?) .



5) **Compiling** (IDE).

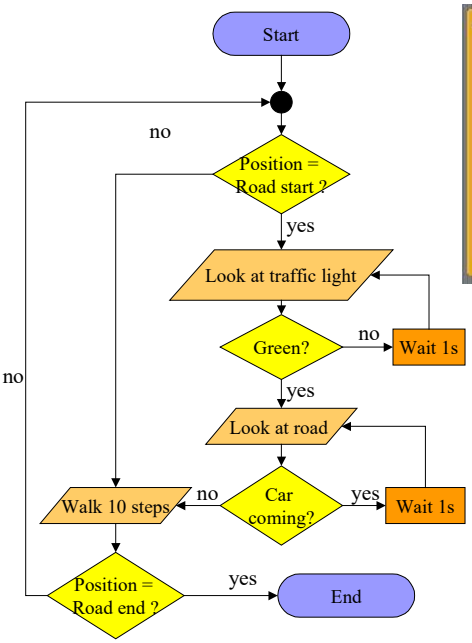
6) **Checking and debugging** (IDE):

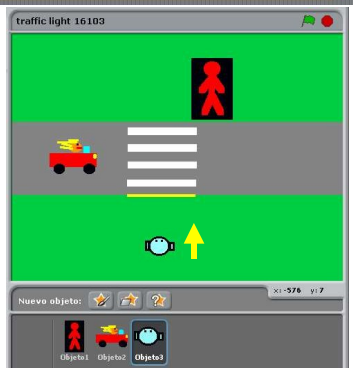
7) **Saving and sharing** (IDE).

Example 3 (robot at the traffic light)

1)**Problem definition:** It's about simulating a robot trying to cross a street at a crosswalk with a traffic light and with cars driving through the street:

1.1) The crosswalk should appear more or less in the middle of the screen.
1.2) The traffic light changes each period of time from red to green and back to red (e.g. each 5 or 6 s).
1.3) A car passes each period of time (e.g. each 10 s), and does not stop at the traffic light, even if it is green for the pedestrian.
1.4) The robot will only cross if the traffic light is green and no car is coming. If it succeeds crossing, it will start again. If it is run over by the car the whole program should end.





2) **Search for ideas**:

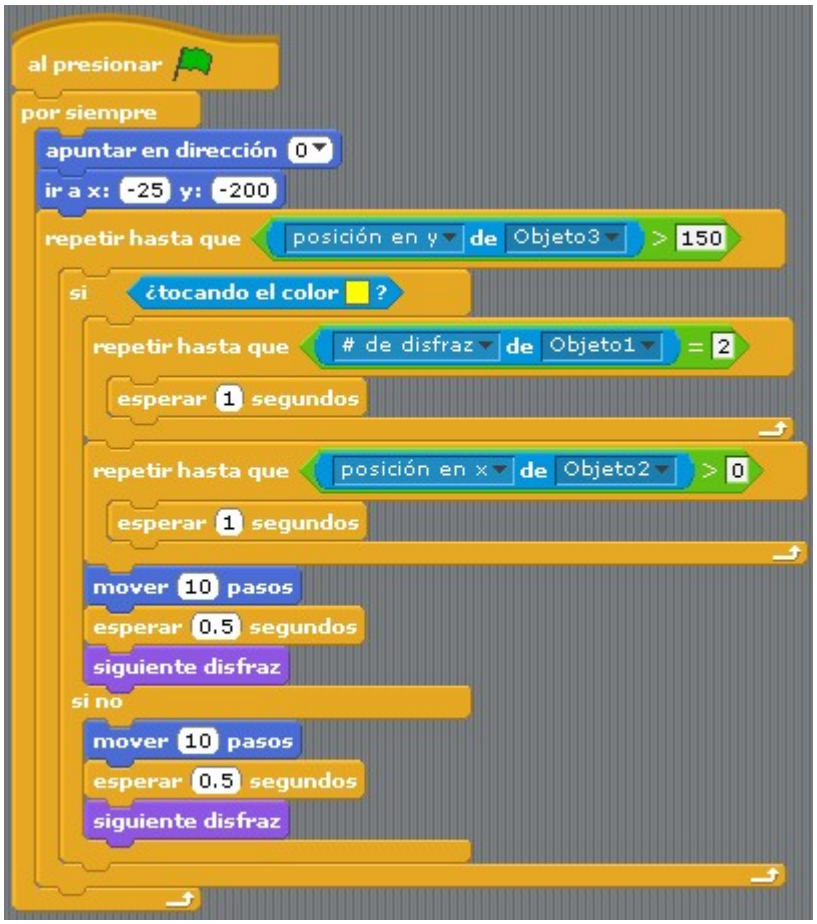
2.1) Stage: Design a backdrop.
2.2) Sprites: Design the traffic light with two disguises (red and green), a car with two disguises (to simulate speed: e.g. hair of the driver moving) and a robot with two disguises (also to simulate motion).

3) **Design**: Design four flow diagrams: one for the traffic light and a second one for the car. You will find the flow diagram for the robot at the top of this page. The fourth flow diagram refers to the accident situation (the robot is "touched" by the car).

4) **Programming** (IDE):

4.1) Traffic light: During 5s it will stay red and then it changes to green and the sound of a bird will sound till it changes back to red.

4.2) Car: It starts at the left of the screen, then it starts moving to the right changing continuosly its disguise (to simulate speed). When reached the right part of the screen, it starts again at the left.



4.3) Robot: **A)** It starts at the bottom of the screen and moves upwards, till it reaches the crosswalk (yellow color). **B)** There, it will wait till the light is green (disguise no.2) and no car is at the right part of the screen (car's x-position > 0) **C)** Once it has crossed completely the crosswalk (robot's y-position > 150), it starts again.

4.4) Robot (accident): If the robot is touched by the car (red color), it dissapears, a bell will sound and the program ends.

Activity: Solve following exercises in the computerrom:

1) Example 1 (Scissors rioter-game).
   a. Program with Scratch the given example.
   b. Improve the game:
      i. Add a timer
      ii. Add a scoreboard with the number of times the scissors reaches the rioter.
      iii. Additionaly each time the rioter is reached by the scissors, he should shout through the speakers.

2) Example 2 (TPR final mark).
   a. Program with Scratch the given example.
   b. Improve the program:
      i. Add different sounds to the output of each the conditional algorithm.
      ii. Each iteration you take 1 off the 'number of students'-variable, so that the counter always tells you the number of resting students.
      iii. How would you proceed to make the program calculate the whole mean of the class once all the iterations have finished? Let's try to program it!

3) Example 3 (Traffic light).
   a. Program with Scratch the given example.
   b. Improve the program:
      i. Add several robots with similar subprograms but different speeds, and let see how they behave!
      ii. Add a second car, which stops at the traffic light if the robots are crossing.

4) New example (Table tennis):
   a. Do the search-for-ideas step for following problem: programming a ping-pong game with following features:
      i. The program is designed for two players
      ii. It will have a scoreboard.
      iii. The game is over when one of the players gets four points.
      iv. Each time the racquet hits the ball, there should be a popping sound.
      v. Once the game is finished the program declares the winner.
   b. Design the flow diagrams for the table tennis game.
   c. Program with Scratch the table tennis game.