Unit 5: Programming

Autor: Guillermo Gómez

Contents

Prior knowledge 2
Keywords 2
Mindmap of the unit 2
5.1. Programming basics 3
5.2. Example of IDE: Scratch 4
5.2.1. Introduction 4
5.2.2. GUI (Interface) 4
5.2.3. Example 5
5.3. Programming apps for mobile devices: MIT App inventor 7

Textos Marea Verde

Prior knowledge

Activity: Summarize your general knowledge on this topic.

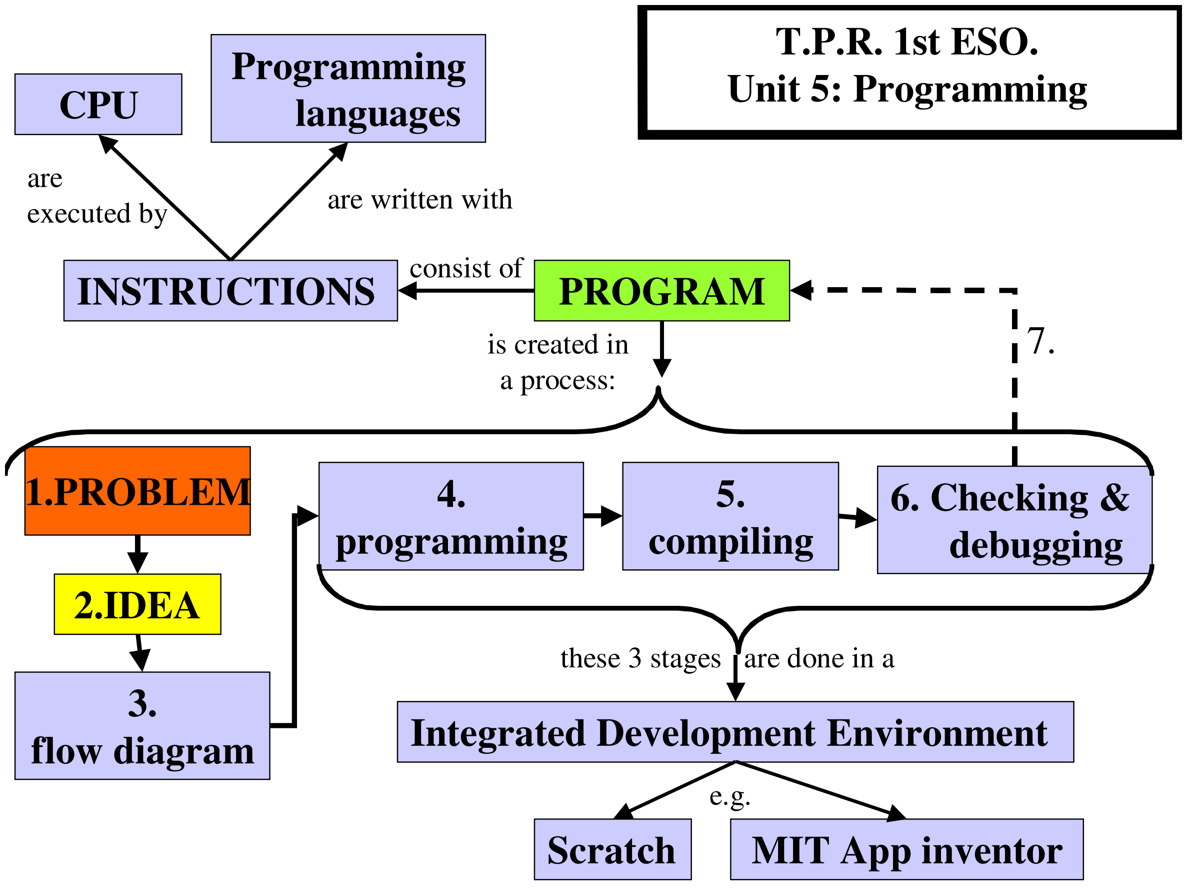
Keywords

Activity: Copy following keywords, explaining their meaning and translate them into Spanish.

Command	Algorithm	Script
Flow diagram	GUI	Event
IDE	duplicate	Operator
compile	shrink	Timer
machine code	Variable	Scoreboard
Bug	Backdrop	Random
debug	Sprite	Grid
Subroutine	Tab	Screenshot

Mindmap of the unit

Activity: Analize and try to understand following mindmap



5.1. Programming basics

Program, languages and IDE:

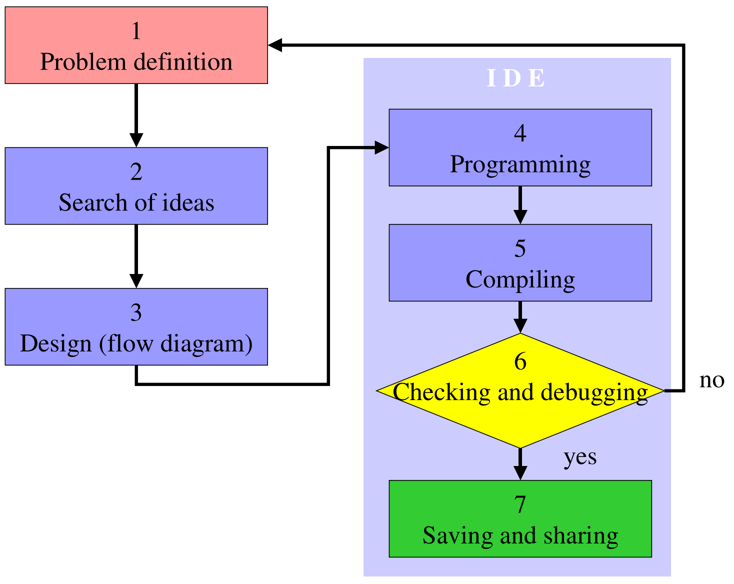
A **program** is a collection of instructions that performs a specific task when executed by the CPU of a computer. A part of a program that performs a well-defined task is known as an **algorithm**; you may also find the names **subroutine**, **procedure** or **subprogram**.

Programs are written in a **programming language**, which is human-readable. There are different **programming languages**, each of them with their own commands (vocabulary) and syntax (way of using commands).

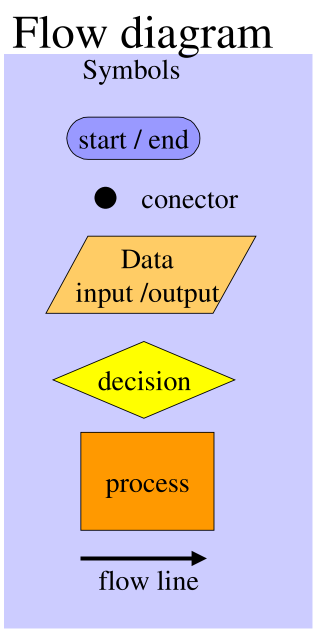
The computer only can directly execute instructions written in a special language, the **machine code**. The process of translating from a programming language into a machine code is called **compiling**.

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

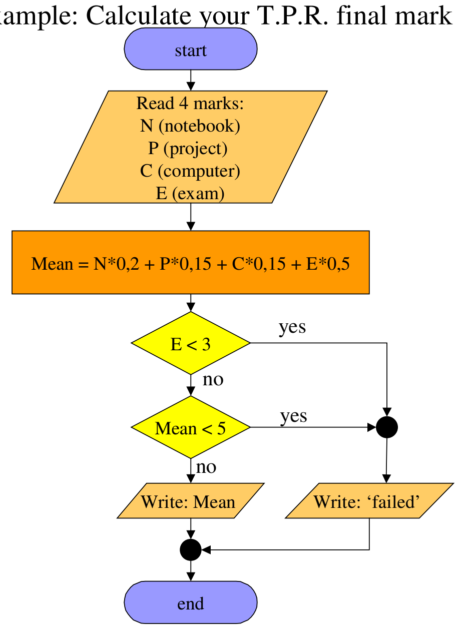
1. **program** with a programming language (human-readable form)
2. **compile** into machine code- (computer-executable form)
3. **run** the program
4. **save** the program



Programming process:
1)**Problem definition:** Identify clearly the needs.
2) **Search for ideas**: Choose the right programming language and study the needed algorithms (variables, operations and formulas).
3) **Design**: Represent each algorithm with a flow diagram
4) **Programming** (IDE)**:** write the program using the choosen language
5) **Compiling** (IDE).
6) **Checking and debugging** (IDE): run the program and if it doesn't meet the needs (problem definition) fix the errors (bugs).
7) **Saving and sharing** (IDE).



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



Activity: Copy following exercises and solve them in your notebook

1) Explain what does the program designed in the flow diagram of page 3 do.
2) Design a flow diagram in which the user writes three califications and the program calculates the mean.
3) Design a flow diagram in which the user choose two two-digit numbers and the program adds them together.
4) Design a flow diagram in which the user writes two numbers and the program shows the biggest one.

5.2. Example of IDE: Scratch

5.2.1. Introduction

Scratch is a new programming language based on a free IDE, that makes it easy to create your own interactive stories, games, and animations – and share your creations with others on the web.

5.2.2. GUI (Interface)

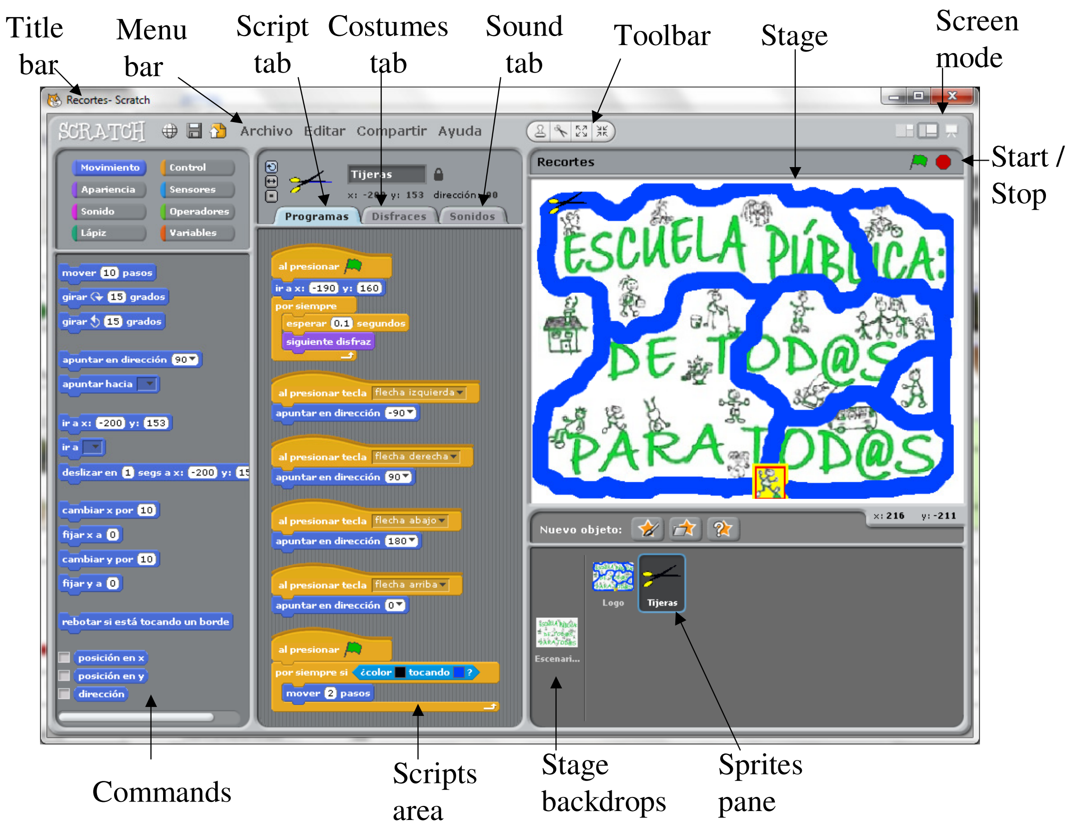
The interface consist of 4 bars:

- title
- menu
- tool
- screen mode

and 4 areas:

- commands
- tabs: scripts / costumes / sounds
- stage
- object panes

Depending on the object selected on the panes (sprites or backdrops), the corresponding information will appear on the tabs.



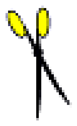
```
al presionar 🏁
ir a x: -190 y: 160
por siempre
    esperar 0.1 segundos
    siguiente disfraz

al presionar tecla flecha izquierda
apuntar en dirección -90

al presionar tecla flecha derecha
apuntar en dirección 90

al presionar tecla flecha abajo
apuntar en dirección 180

al presionar tecla flecha arriba
apuntar en dirección 0

al presionar 🏁
por siempre si ¿color ■ tocando ■ ?
    mover 2 pasos
```

Programming, checking and debugging

The program blocks (commands) are dragged into the script area. Their shape indicates how they are assembled. When pressing the start button the action takes place in the stage area, where you can check and decide if debugging is necessary.

5.2.3. Example

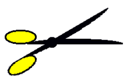
1)**Problem definition:** It’s about programming a game similar to the classic pacman: using the arrow keys to move, you have to guide a scissors within a maze to reach a rioter.

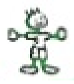


1.1) The scissors moves forward as long it is in contact with the maze’s path
1.2) Whenever you push an arrow key, the scissors points in the direction you push
1.3) When the scissors reaches a rioter, a new rioter appears in another part.

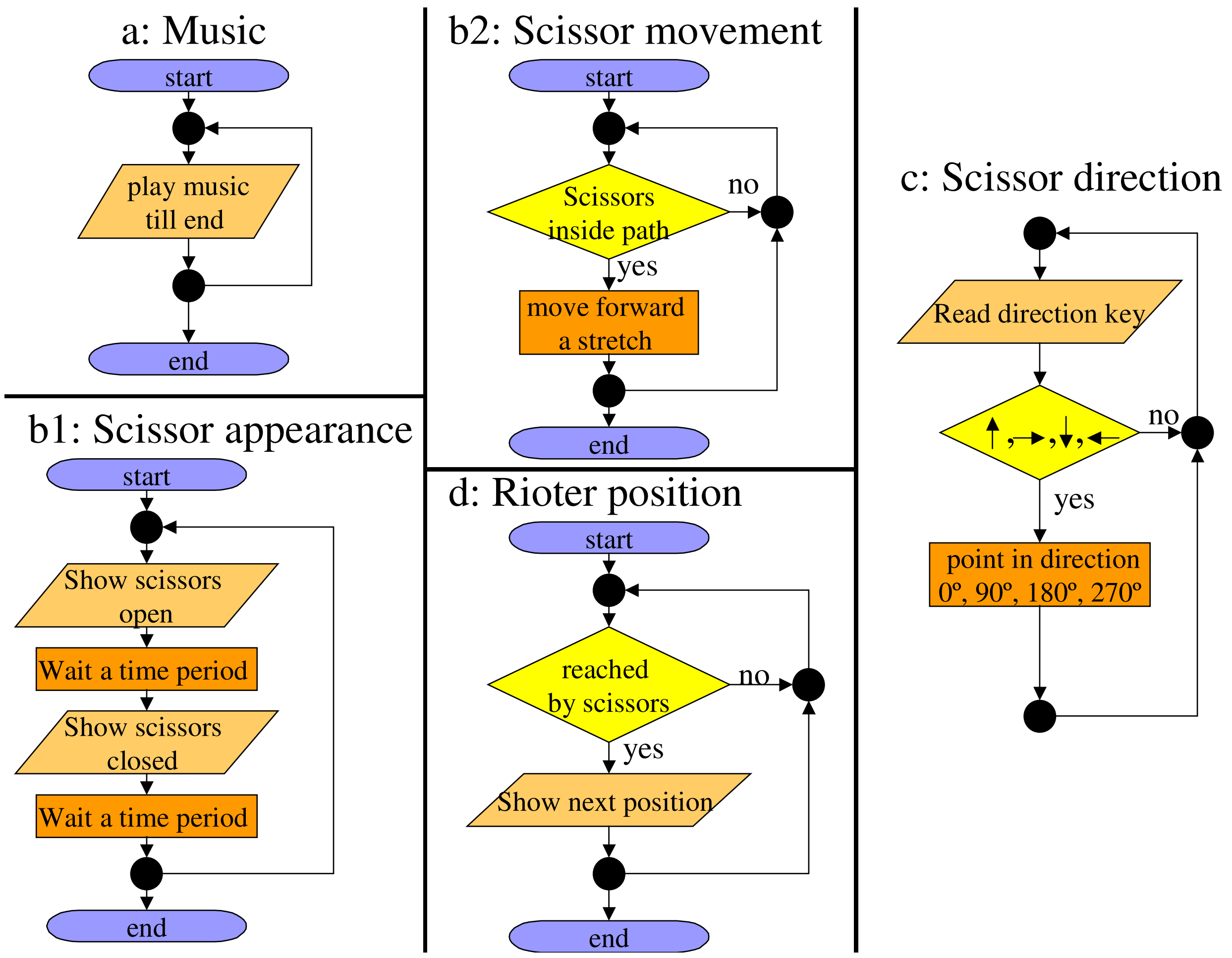
2) **Search for ideas**:

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**:



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

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

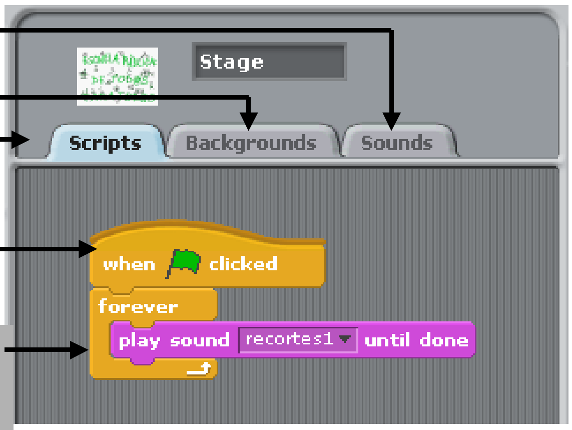
click to import the music

click to import the background

click to program

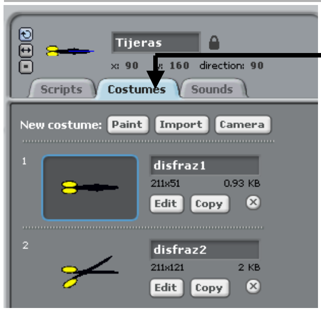
Use the *when green flag is clicked* event to make play the music when the program starts

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

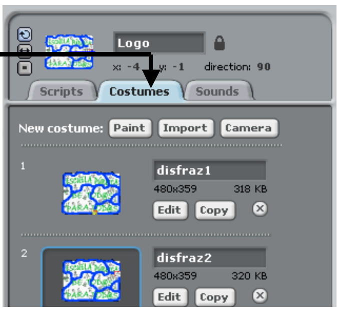


```
when [flag] clicked
forever
    play sound recortes1 until done
```

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


Tijeras
x: 90 y: 160 direction: 90
Scripts
Costumes
Sounds
New costume: Paint Import Camera
1 disfraz1 211x51 0.93 KB Edit Copy
2 disfraz2 211x121 2 KB Edit Copy

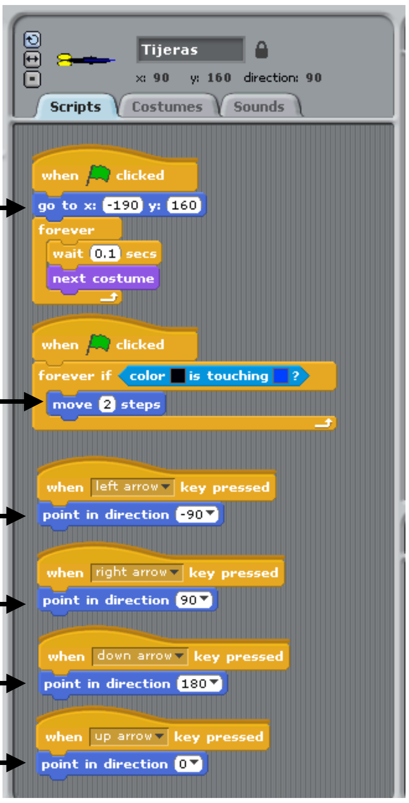
click to design the different costumes for the sprites


Logo
x: -4 y: -1 direction: 90
Scripts
Costumes
Sounds
New costume: Paint Import Camera
1 disfraz1 480x359 318 KB Edit Copy
2 disfraz2 480x359 320 KB Edit Copy

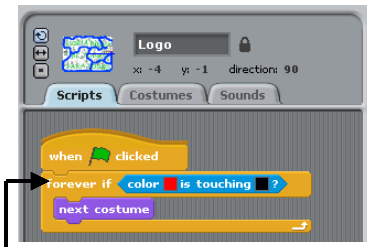
4.3) Subroutines for the sprites: program subroutine b and c for the scissors and subroutine d for the maze with the rioter.

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



```
when [flag] clicked
go to x: -190 y: 160
forever
    wait 0.1 secs
    next costume

when [flag] clicked
forever if color [■] is touching [■] ?
    move 2 steps

when left arrow key pressed
point in direction -90

when right arrow key pressed
point in direction 90

when down arrow key pressed
point in direction 180

when up arrow key pressed
point in direction 0
```

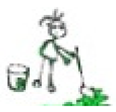


```
when [flag] clicked
forever if color [■] is touching [■] ?
    next costume
```

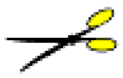
Use the control block *forever if* to change the position of the rioter (red color) each time it is touched by the scissors (black color)

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.



Activity: Copy following exercises and solve them in your notebook

5) Program with Scratch the given example (Scissors rioter-game).
6) Improve the game: Design a new flow diagram for a subroutine in which the stage will show the time on a timer and a scoreboard with the number of times the scissors reaches the rioter. Improve the existing flow diagram so that it meets this new requirement; additionaly each time the rioter is reached by the scissors, he should shout through the speakers.
7) Add with Scratch the improvements designed in exercise 6.
8) Do the search-for-ideas step for following problem: programming a ping-pong game with following features:
    a. The program is designed for two players
    b. It will have a scoreboard.
    c. The game is over when one of the players gets four points.
    d. Each time the racquet hits the ball, there should be a popping sound.
    e. Once the game is finished the program declares the winner.
9) Design the flow diagrams for the ping-pong game.
10) Program with Scratch the ping-pong game.

5.3. Programming apps for mobile devices: MIT App inventor

App Inventor is a IDE for designing mobile apps for the Android operating system. It was created by Google, but now the responsible for its support is the Massachusetts Institute of Technology (MIT).

The programming is very similar to the one in Scratch. To get started, just follow the instructions of the video: http://appinventor.mit.edu/explore/ai2/beginner-videos.html

As a student of second grade, you probably have a mobile device,... but have you ever heard about **nomofobia**? The term is an abbreviation for “no-mobile-phone phobia”, a term describing the fear of being without a mobile device. Among today’s high schools it’s on the rise: more than two of three students sleep with or next to their smart phones!

Using mobile devices you can get an addiction that dictates your behavior!

Although it is true that mobile devices make life easier and enables us to work more efficiently, don’t forget that they should stay ‘always as servants, never as masters’.