Classes Introduction

1. Learning aims

At the end of these exercises you should be able to:

- understand why classes are useful
- Define a class having data members, methods, constructor
- Instantiate a class and use its functionality

2. Course Material

To make the exercises of this lab, you need the following parts of the book

- Processing : A programming Handbook for Visual Designers and Artists (C. Reas & B. Fry)
  - Chapter **Objects** (p. 359 > p. 376)
- https://www.youtube.com/results?search_query=processing+tutorial+from+beginner+to+games

3. Lab exercises

3.1. Sketch Demo_02_01_Cars

❖ **Car class**:

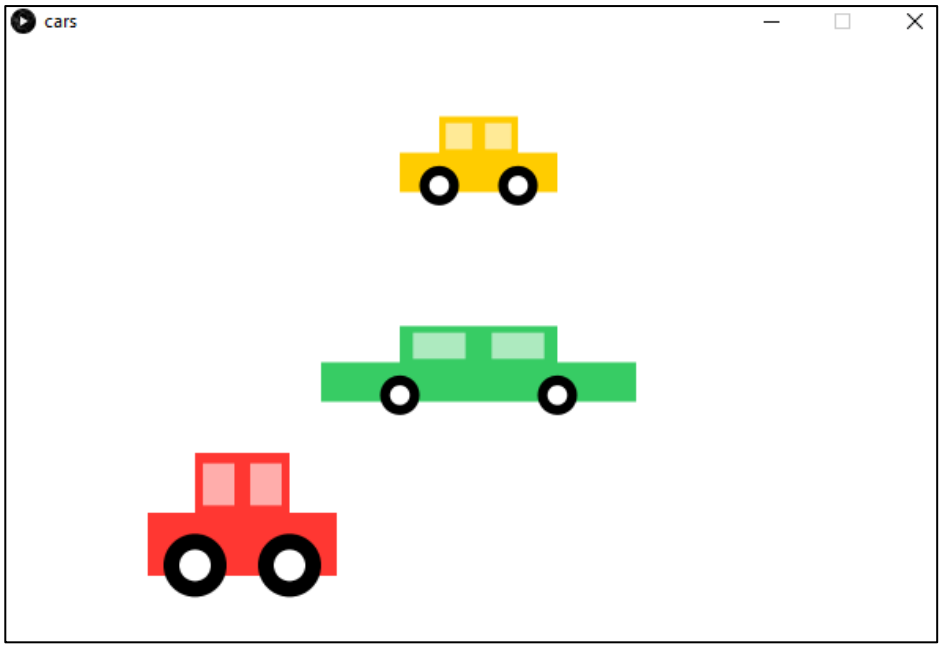
Data members:
xPos, yPos, w, h (float)
clr(color), started(boolean), speed(float)

Constructor:
Car(float xPos, float yPos, float w, float h, color c, float speed)

Methods:

- display(): displays a coloured car with the correct dimensions on a correct position
- move(float limit): updates the x-position of the cars with the speed variable, using the 'r' key
- setSpeed(float speed): sets a different speed for a specific car

❖ **Main application**
Make 3 instances of the car-class in 3 different colours (e.g. red, green and yellow) and display the 3 cars.



3.2. Sketch Lab_02_02_Lights

❖ **Light class**

The class represents a LED-light that can be set on or off.

Data members:

xPos, yPos, radius (float)
clr (color), isOn (boolean)

Constructor:

Write a **constructor** with 4 arguments (xPos, yPos, radius, clr) that initializes the corresponding data members. All lights are off when they are instantiated.

Methods:

Write a method **display()** that displays an ellipse at the correct position. The ellipse has the appropriate color when the light is on and has a grey colour (190,190,190) when the light is not working.

Write a method **hitTest(float mx, float my)** that changes the value of the datamember *isOn* when the mouse coordinates x and y falls within the region of the light.

Write a function **getIntegerValueOfIsOn()** that **returns** an **integer** with the value 1 or 0 depending on the boolean datamember *isOn*. This method can then be used to count how many lights are on.

❖ **Main application**

The size of the window is set to 500 x 100. Make 4 instances of the light class in the **setup** and give the correct values to the constructor of the class. The lights have a radius of 15 pixels, a yellow colour (255,255,20) and position them next to each other. The first one is on position (20, 20).

In the **draw** method, the lights are displayed and you count the number of lights that are on by calling the **getIntegerValueOfIsOn**() function of every light object. The total number of lights is shown below the lights.

In the **mousePressed** method, the hitTest-method of the 4 lights is called by passing the current mouse-position.

3.3. Sketch Lab_02_03_Fraction

❖ **Fraction Class**

The fraction class represents a simple fraction (e.g 2 / 4) consisting of a numerator (2) and a denominator (4).

Data members:

numerator(int) and denumerator (int)

Constructor:

The **constructor** has 2 arguments and initializes these data members.

Methods:

The **display** method has 4 arguments: float x, float y, float size, color fillColour. It draws the fraction as a portion of a rectangle and next to this as an arc. The **size** parameter (20px) indicates the size of **one** cell. Use a for-loop to display the rectangles.

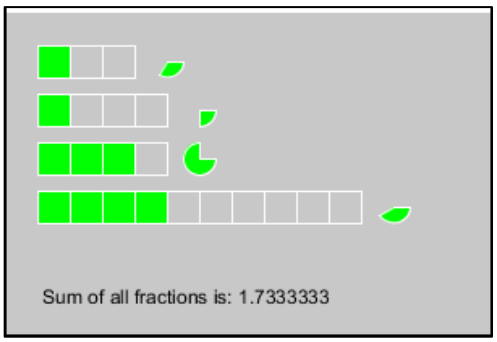
The **getValue** method returns the value of the fraction as a float.

❖ **Main Application**

The size of the window is set to 300 x 200. Now make **4 instances of the Fraction class** giving each object different numerator and denominator values:

1/3
1/4
3/4
4/10

Call the display method to display them below each other on the window as you can see in this screenshot. Also, show the sum of the fraction values.



3.4. Sketch Lab_02_04_Birds

❖ **Bird class**

A bird class is a representation of a flying bird using a spritesheet.

Data members:

xPos (float), yPos (float), speed (int), imgSprite (PImage), numFrames (int)

To make the animation we need 3 more variables:

frameCounter (int), totalTime (float),

frameTime = 1/8.0 (float) (this to determine how long an image is displayed per frame)

Constructor:

Write a **constructor** with 5 arguments (filename, numFrames, xPos, yPos, speed) that initializes the corresponding data members.

Methods:

Write a method **display()** that displays the correct frame of the image. Initialize the frameWidth and frameHeight by using the width and the height of the image and the number of frames and display the correct part of the image.

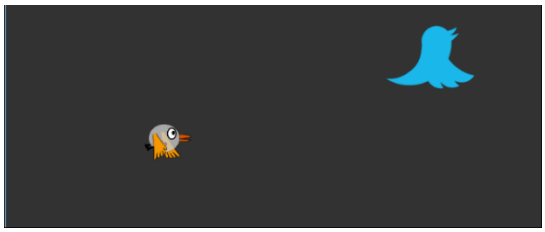
Write a method **update()** that updates the frameCounter and the X-position to show the right image.

To determine how long one image is shown per frame, start by adding an amount to the totalTime variable (f.e. 1/60.0). If the totalTime exceeds the frameTime, update the frameCounter and reset the totalTime. The frameCounter can be initialized again by using the numFrames.

As soon as the position gets out of the window, it is reset and the bird starts again at the left side of the screen.

❖ **Main Application**

The size of the window is set to 842 x 480. Make 2 bird-objects with the given images and make them fly by using the display and the update-method.



4. Reflection

What is the difference between a class and an instance of a class?
What is the use of data members in a class?

When is the constructor of a class called?

The dot-notation is used to call a method of an object. Do you understand clearly what this means? Check some examples in your code!

Why do we need to create getters and setters?

5. Saving instructions

- Make a folder named 1DAE***xx***_PROGFA**2**_**02**_***name_firstname*** (e.g. 1DAE03_PROGFA**2**_**02**_Van_der_Veken_Jan)
- Add all the folders with the corresponding Lab 02 pde-files.