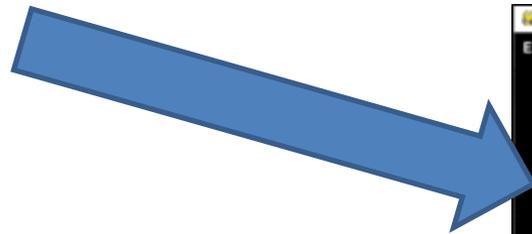


And now: let's draw some images

- In Mu editor, load the program: `example1_1.py` and **run it!**



- Can you “read” this program?
 - What do you think happens in line 5?
 - And in line 8?
 - What happens in like 9?

...

Change the code

- Create a new program
 - copy and paste all the code from `example1_1.py` into the new file
 - then save the new program with name: `morecakes.py`
- **Now:** modify the program in `morecakes.py` so that:
 - the 2 cakes are side by side, and with more space between them, like this: The image shows two small, round cakes placed side-by-side. The cake on the left is decorated with blue and white frosting, while the cake on the right is decorated with red and white frosting. They are both on a light-colored surface.
 - And place another cake (repeat the first cake!) at another position on the screen. *Can you place it in the center?*

What we could do next...?

- A. Draw some images, wait for user to click (or a few moments), then show the same images but in different positions: animation
- B. Draw in *random* positions
- C. Draw sometimes an image, sometimes another one
- D. **A more complex example:** compose parts of an infographic, then use random values to create a different infographic at every run

A. Draw some images, wait, draw them moved: animation

- In Mu editor, load the program:
`example1_2.py`
and **run it!**
- **Now read the code** and try to see how the commands match the behavior of the program.
 - What does the command **clear()** do?
 - And what does **wait(0.4)** do?

A. Change the code

- Create a new program, call it: `animation.py` and copy+paste the code from `example1_2.py` into your new program.
- **Now try to make the program to this:**
 1. Move the cake twice as it already does, but much slower
 2. Add another step in the animation, so that the new cake appears at position 145,180
 3. Change the cake image in the entire animation to be the other cake: *cake2.png*
 4. Add the text “Oishi!” below the cake, and make it move towards the right of the screen. The movement should be 10 pixels to the right at each new step of the animation.

B. Draw in random positions

- Load the program: `example1_3.py` and **run it!**
- **Now read the code** and try to see how the commands match the behavior of the program.
 - What does the command **`randint(1,3)`** do?
 - What would be drawn, if we change the code to use **`randint(1,4)`** ?

```
if test :  
    → do something
```

B. Change the code

- Create a new program, call it: `jumparound.py` and copy+paste the code from `example1_3.py` into your new program.
- **Now try these changes:**
 1. Change the position at which the `cake.png` image is drawn, by using `randint(10,50)` for its X coordinate. Run a few times to see the effect.
 2. Try instead with `randint(500,700)`, and run to see the effect.
 3. Create a second random number and use it for the Y coordinate. Run to see the effect of that on your cakes!

C. Draw sometimes an image, sometimes another one

- Load the program: `example1_4.py` and **run it a couple of times!**

```
Running: example5.py
pygame 1.9.6
Hello from the pygame community. https://www.pygame.org/contribute.html
Do you like blueberries on your cake? [yes/no] |
```

Type your answer here!

- **Now read the code...**
 - What does the **IF ...** do?
 - Is it possible that the program draws both cakes?
 - What happens if the you type “banana” instead of “yes”?

`if test :`
→ do something

C. Change the code

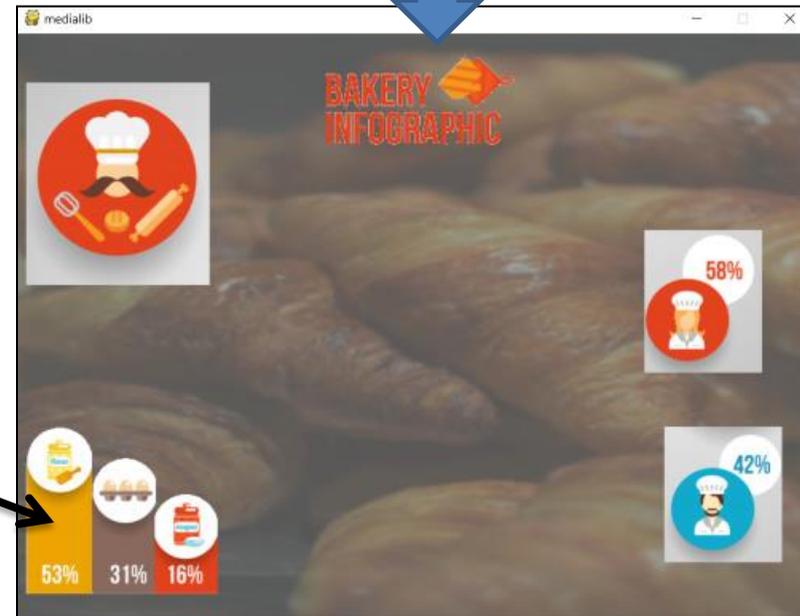
- Create a new program, call it: `decisions.py` and copy+paste the code from `example1_4.py` into your new program.
- **Now try these changes:**
 1. Add a new IF statement, so that when the user types “sure” it draws 3 copies of the first cake, one on the top of the other. Place the cakes where you like on the graphic window.
 2. Add a new IF statement, so that if the user types “banana”, the program write a text saying “Sorry, no banana cakes available”, in position 50,10, with a 32 pixel font.

D. Compose an infographic

- Load the program: `example1_5.py` and **run it**
- **Now read the code...**

– How is it possible that the **barchart** on the bottom-left is made of 3 separate images? Where are these 3 images positioned?

– What happens if line 7, that draws the *background.jpg* image, is moved to line 10?



D. Change the code

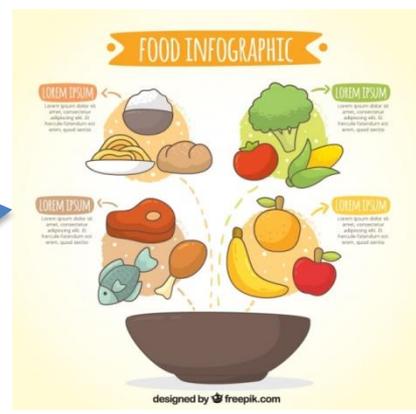


- Create a new program, call it: `myinfo.py` and copy+paste the code from `example1_6.py` into your new program.
- **Now try these changes:**
 1. Change the code to move the bar chart with flour, eggs and sugar to the center of the graphic window
 2. Align the 2 images *girls.png* and *boys.png* to the right, so they are one on top of the other vertically.
 3. Change the code so that after the user clicks, the 2 cakes (images *cake.png* and *cake2.png*) appear, at the center of the window, just under the title image. When the user clicks a second time the program should exit.

EXAMPLE OF ...

Task (for next lecture)

- Using the code examples from this lecture, create your graphics composition
 - first pick a theme: **for example “food”**
 - show a composition on the screen, and when the user clicks, show a variation of your composition; **example:**
 - (CHALLENGE)** Use random numbers to show different images in your composition each time the program runs



- Consider using **pixbay** to find more interesting images –
You can download your images, rename them
and place them in the code/images folder