COMP1021
Introduction to Computer Science

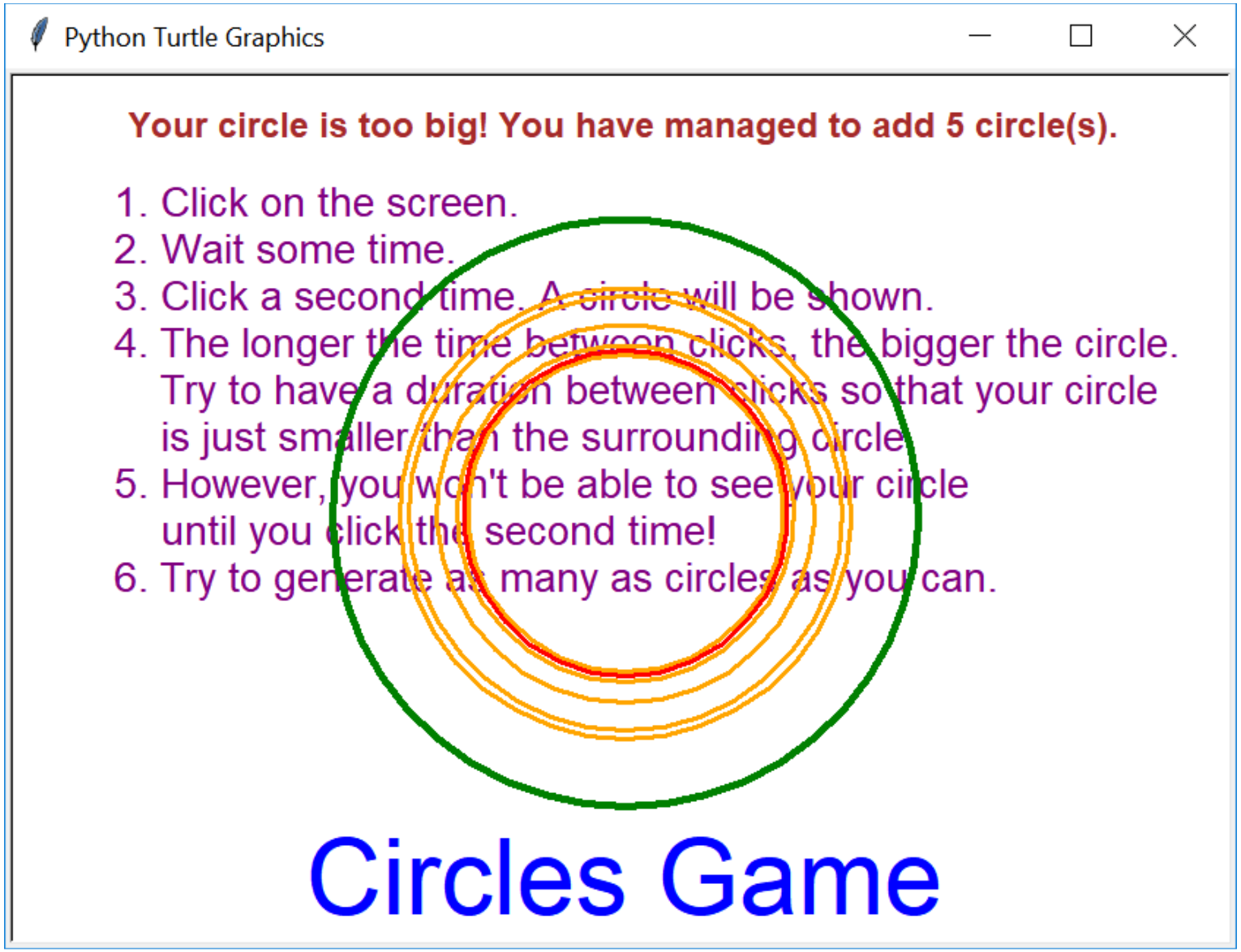
An Example Game – Circles Game

David Rossiter and Gibson Lam

Outcomes

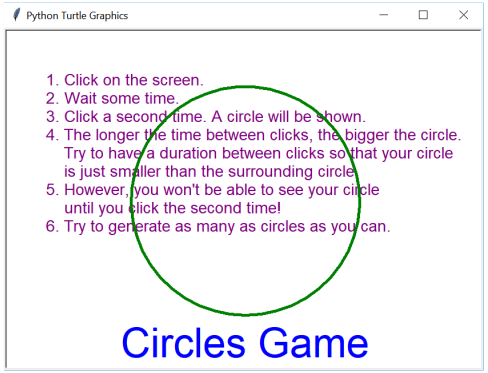
- After completing this presentation, you are expected to be able to:
  1. Understand the Python code for creating an example circles game

An Example Game – Circles Game



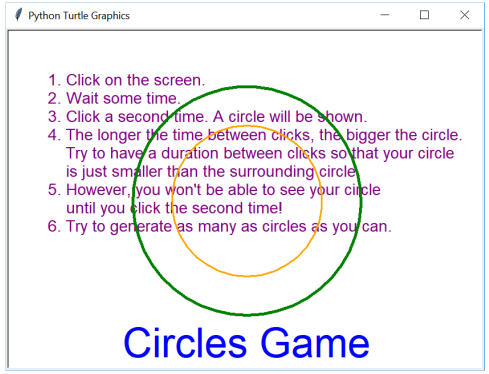
Making Circles in the Game

- In this game, the player has to click on the screen (meaning the window) once, wait for a bit of time and click on the screen the second time



Click anywhere on the screen once

Wait for a bit of time

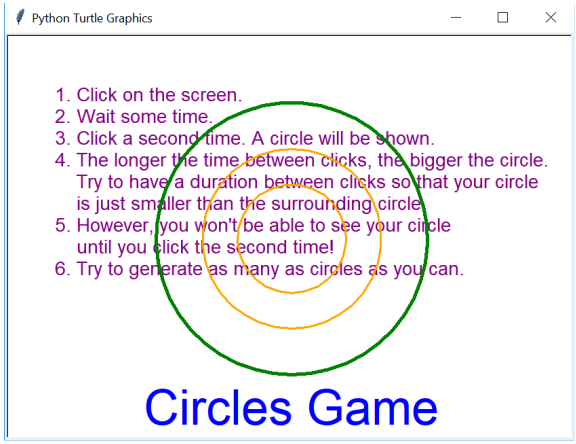


Click again and an orange circle is shown

- A circle is then drawn with a size that is proportional to the time between the clicks

Size of the Circles

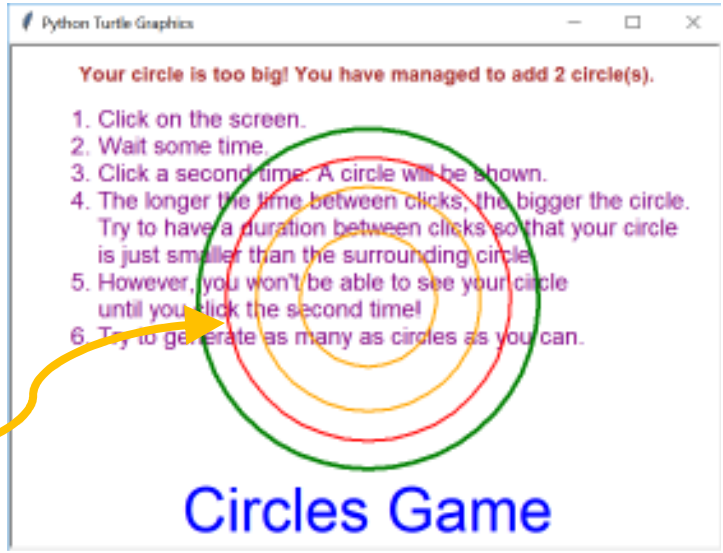
- Any circle that the player draws is not allowed to grow larger than the size of the previously drawn circle
- Therefore, the player has to click the second time when he/she thinks the size of the invisible circle is getting close to the size of the previous circle



In this example, the second orange circle is smaller than the first one so it is okay

The Objective of the Game

- If the circle drawn by the player is too big, the game will be over
- The circle that is too big is shown in red
- The objective of the game is to draw as many circles as possible



Initializing Global Variables

- The game has a few global variables that store the information of the game:

```
current_maximum_radius = 200
```

Store the maximum radius that the player can draw, which is initialized to 200

```
number_of_drawn_circles = 0
```

Store the number of drawn circles so far

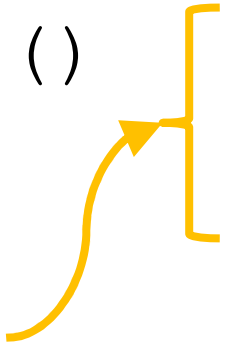
```
content_font = ("Arial", 14, "normal")
title_font = ("Arial", 36, "normal")
game_over_font = ("Arial", 12, "bold")
```

These are the fonts used by the program

The Main Part of the Program

- This is the main part of the program, which sets up the size of the window and the turtle
- The game is started by running the `start_game()` function

The turtle is always hidden in this game



```
turtle.setup(840, 600)

turtle.speed(0)
turtle.up()
turtle.hideturtle()

start_game()

turtle.done()
```

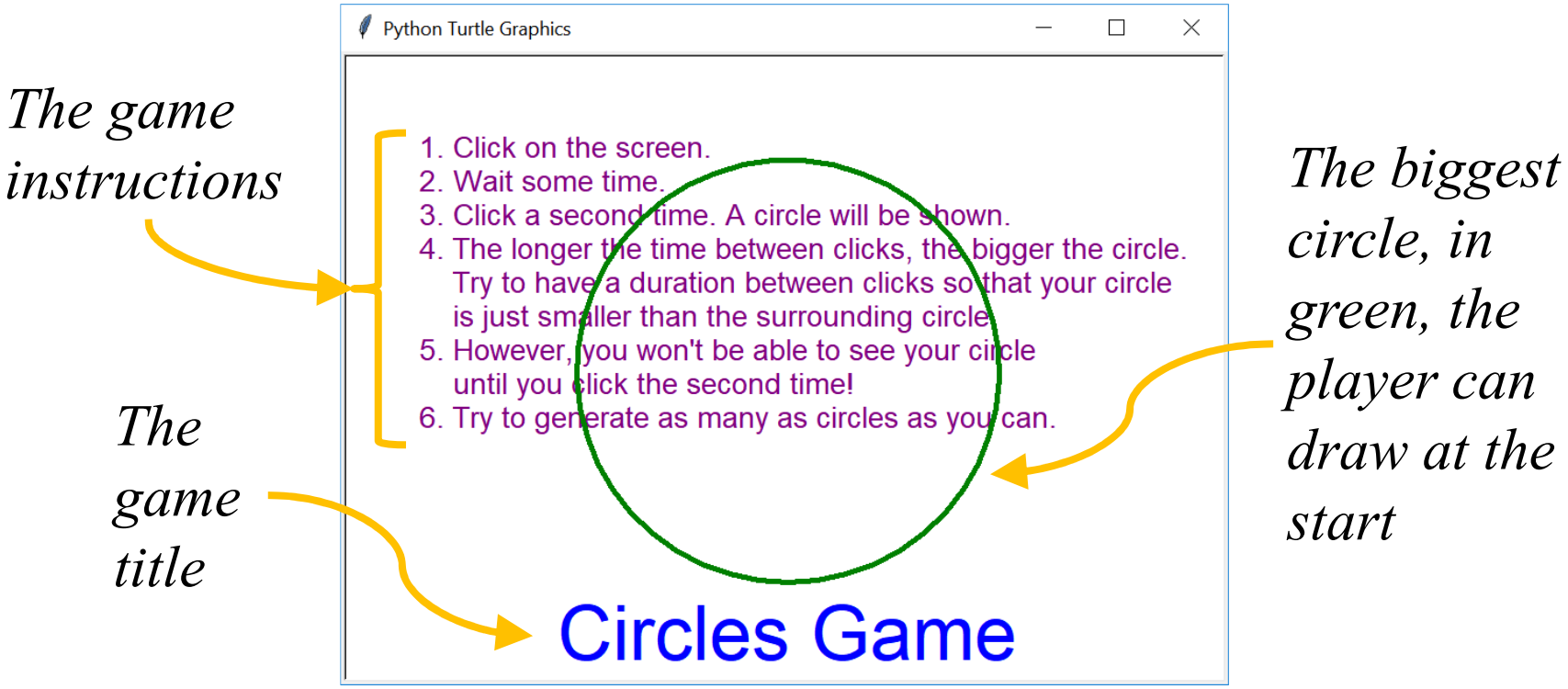
The Start Game Function

```
def start_game():
    show_instructions_and_outer_circle()
    turtle.onscreenclick(first_click)
```

- The `start_game()` function starts the game by doing two things:
  1. Showing the game title, instructions, and the biggest circle that the player can draw at the start, in green
  2. Setting up the screen click event for the first click, which can start the drawing process

Starting the Game

- When the game starts, the display of the window is like this:



The First Click Event Handler

- At the start of the game, the screen click event is assigned to the `first_click()` function
- The function attempts to capture the ‘first click’ from the player
- If the player clicks anywhere in the turtle window, the function will start the drawing process by storing the time the player clicks and then pass the event to the `second_click()` function
- The code of the function is shown on the next slide

The Code of the First Click Function

```
def first_click(x, y):
    global start_time

    print("First click!")

    start_time = time.time()

    turtle.onscreenclick(second_click)
```

Store the time using `time.time()`, which returns the current time in seconds

Use this global variable to store the start time; if the variable has not been created, it will be created here

Set the screen click event to use the other function, which handles the second click

The Second Click Event Handler

- When the player clicks on the window the second time, the game finishes the circle drawing process
- This is handled by the `second_click()` function, which does these:
  1. Turn off the screen click event
  2. Determine the size of the newly drawn circle
  3. If the size is smaller than the currently allowed maximum size, the circle will be drawn and the player can make another circle
  4. Otherwise, the game will be over and a red circle will be shown

Turning Off the Event

- The function uses the following code to turn off the screen click event:

```
turtle.onscreenclick(None)
```

- After checking the size of the newly drawn circle, the screen click event will be assigned to the first click event handler again, if the size of the circle is correct

Finding the Radius of the Circle

- Once the player clicks on the window the second time, the game has to decide the size of the circle, using the time difference between the two clicks
- This is done by the following code:

```
end_time = time.time()
duration = end_time - start_time
current_radius = int(duration * 40)
```

The size of the circle is estimated by multiplying the time duration by 40

Calculate the time between the two clicks

The Circle is Smaller Than the Previous One

- If the circle is smaller than the previous circle, the newly drawn circle will be shown as orange
- The global variables are then updated:

```
current_maximum_radius = current_radius

number_of_drawn_circles =

    number_of_drawn_circles + 1
```

Add the number of drawn circles so far by one

Update the currently allowed radius to be the radius of the newly drawn circle

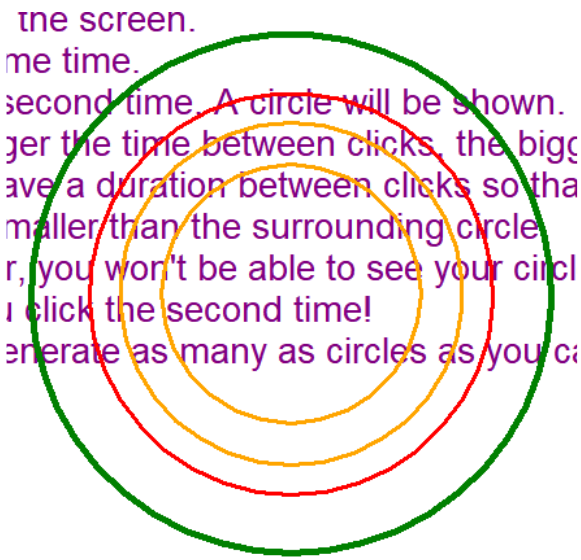
Using the First Click Handler Again

- If the circle has been correctly drawn, the game will continue and wait for the next circle
- The screen click event is then handled by the `first_click()` function again to wait for the circle:

```
turtle.onscreenclick(first_click)
```

- Therefore, the screen click event will be handled by `first_click()` and `second_click()` alternatively, if the player draws circles with the correct size

The Circle is Bigger Than the Previous One



- If the newly drawn circle is bigger than (or equal to) the previous one, the newly drawn circle will be shown as red
- A message is shown at the top of the window and the game is over, for example:

Your circle is too big! You have managed to add 2 circle(s).

A Simple Flow of the Game