PY01: Handbook

Student Name

© 2021 Exceed Robotics ™

Table Of Contents

© 2021 Exceed Robotics ™

© 2021 Exceed Robotics ™

ABOUT THIS WORKBOOK

OVERVIEW

This coursebook is designed to supplement the Python curriculum at Exceed Robotics, by providing reference materials, examples, and homework problems for extra practice. Students are encouraged to use this booklet as a reference and complete the homework problems to further their understanding of the programming languages and develop their programming skills.

Each week's material is divided into three sections.

1. Class summary
2. In-class problems
3. Homework

The class summary section offers students an opportunity to review the core topics covered in the weekly classes.

The in-class problems section describes the different problems the student should have completed during the class.

The homework section provides additional problems that can be used to assess your understanding of the material. The section consists of 4 different types of problems such as,

1. Multiple choice questions
2. Troubleshooting problems
3. Task-based problems
4. Game projects

USING THIS WORKBOOK

One feature of this workbook is that it is editable. The homework section contains multiple-choice questions, which require you to select the correct checkboxes. It also contains problems that require you to type in their answers. Therefore, you must save your handbook so that you do not lose your answers. To save your workbook at any time, you can use **Ctrl+S** for windows or **Cmd+S** for MacOS. When exiting, the handbook, you will also be prompted to save any unsaved changes.

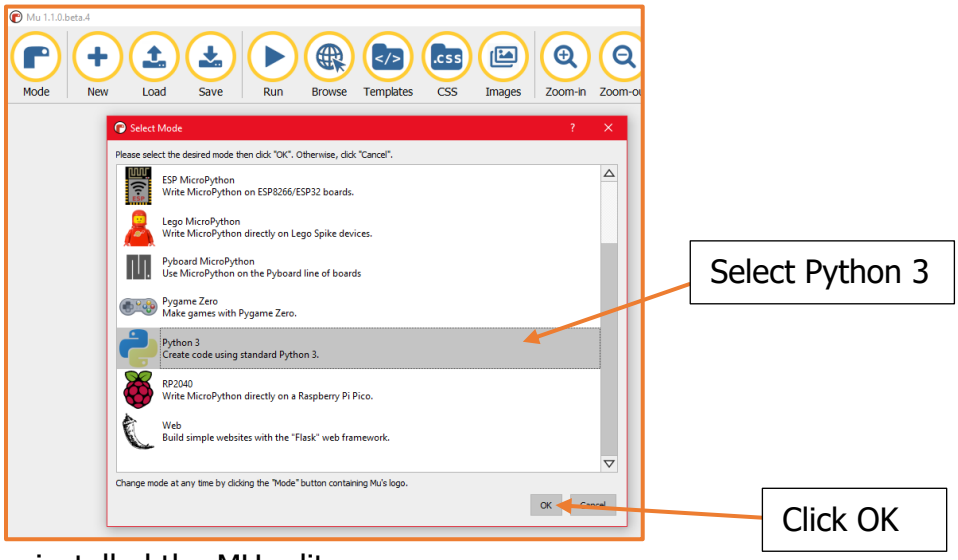
© 2021 Exceed Robotics ™

SETTING UP THE ENVIRONMENT

INSTALLING MU

We need an Integrated Development Environment (IDE) to program or code. **MU** is a simple Python IDE for beginners. To install MU follow the instructions below.

1. Open a web browser and go to https://codewith.mu/
2. Click on the **Download button**.
3. Choose from the three options below depending on your Operating System (MU does not support Windows 32-bit).
4. Once the download is complete, open the file and follow the onscreen instructions.
5. Open the MU editor. (Note: MU takes several minutes to load the first time)
6. MU will prompt you to select a mode. Make sure to select the Python 3 mode. Then click OK.



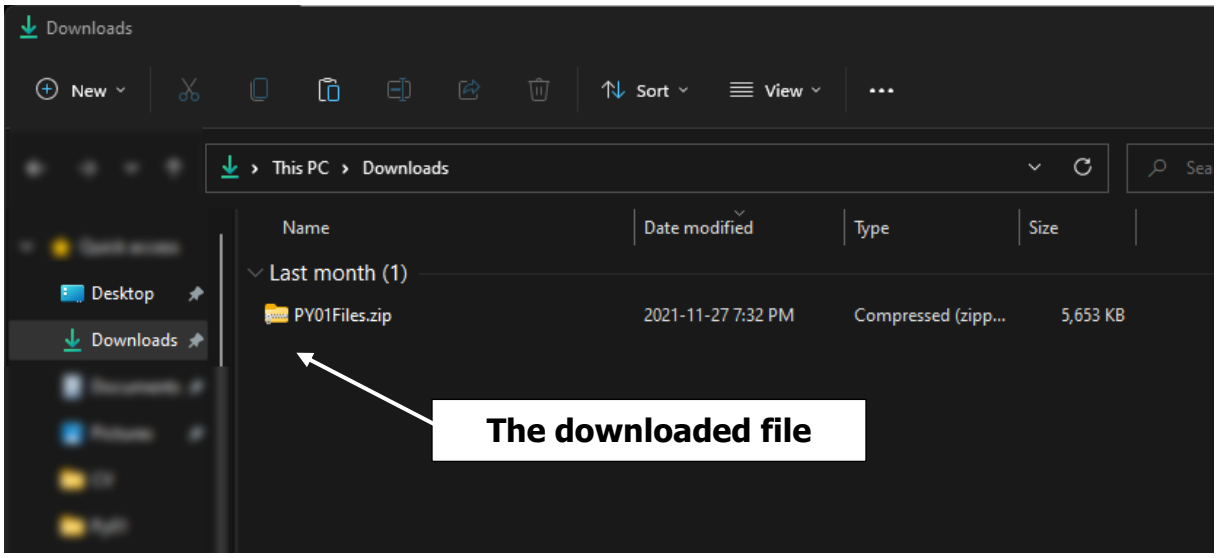
7. Congrats! You have installed the MU editor.

© 2021 Exceed Robotics ™

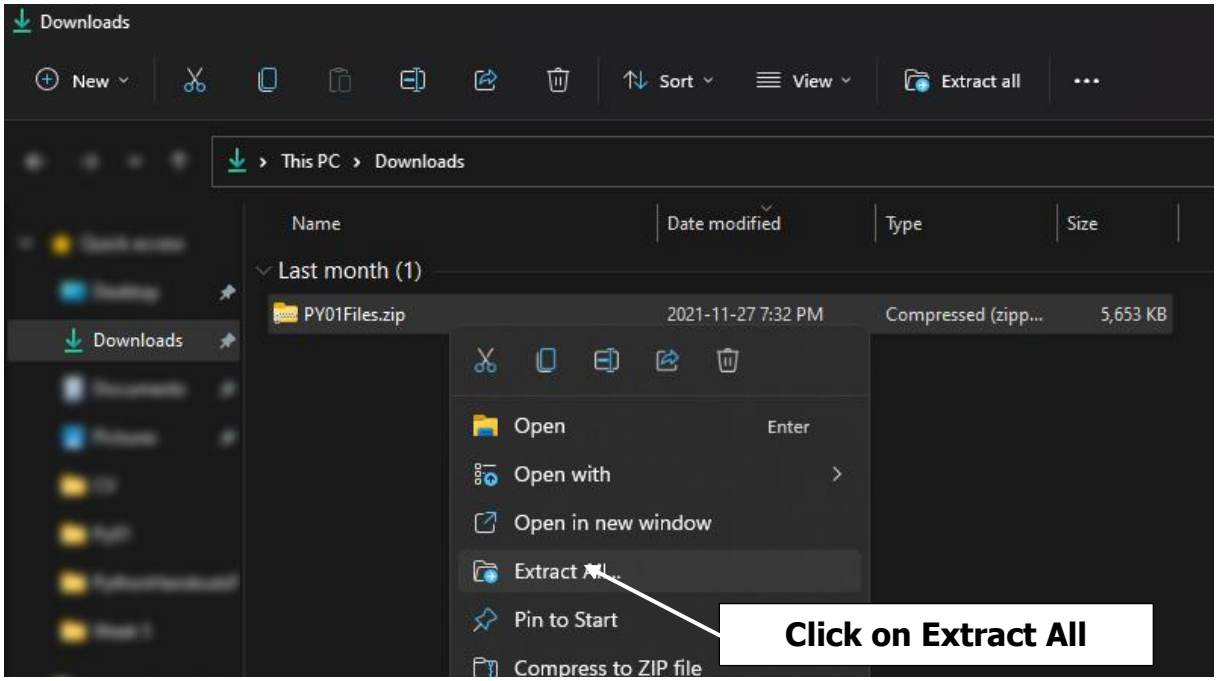
DOWNLOADING THE PY01 FOLDER

It is important to organize your files neatly so that they are not lost. To do so we need to implement a folder structure. Luckily for you, we have provided you with the entire folder structure. To download and set up the folder, follow the instructions below.

1. Click on this link to download the zipped folder for Py-01 (Click here).

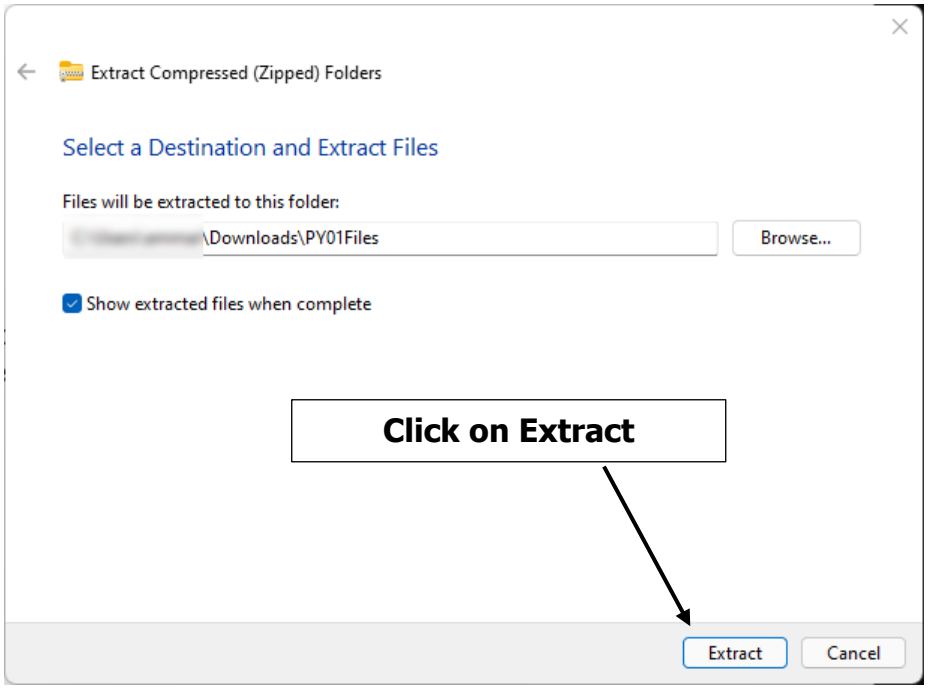


2. Right-Click on the downloaded file and click on "Extract All"

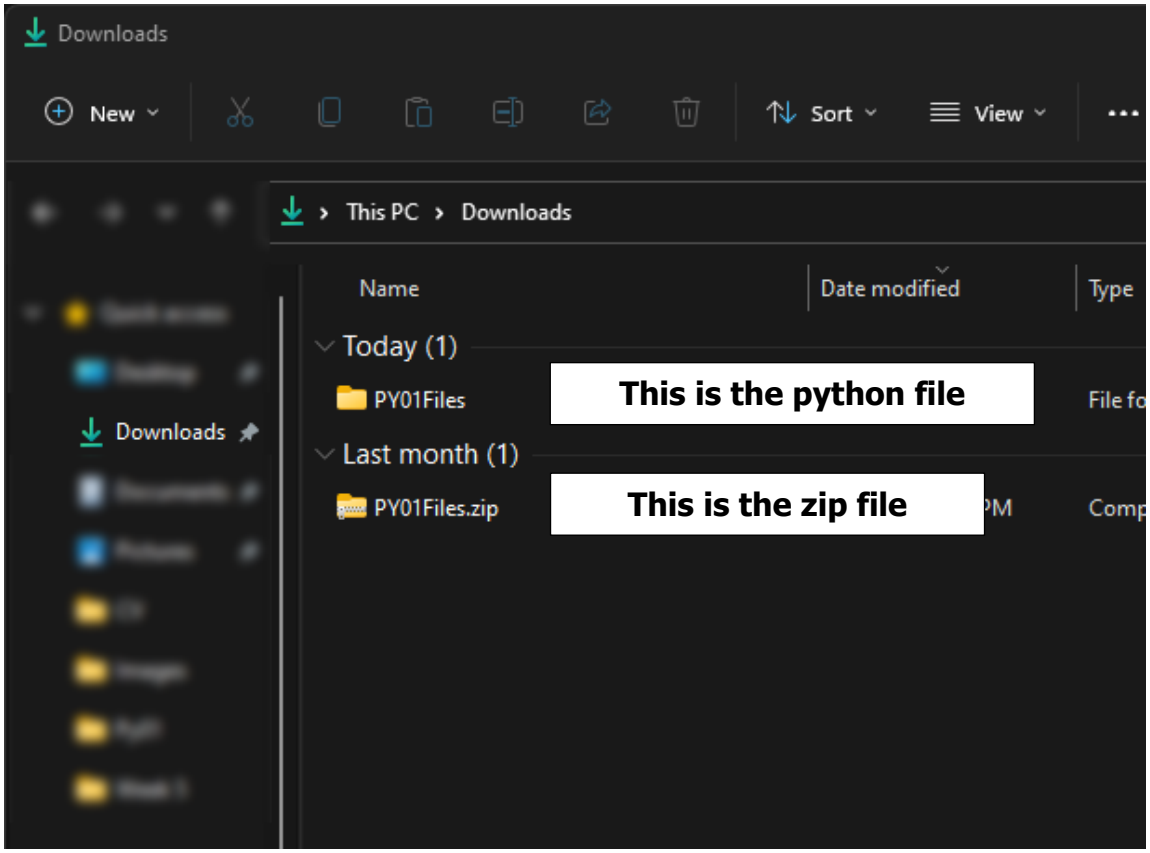


© 2021 Exceed Robotics ™

3. Click on extract



4. You should now have an extra folder in your downloads folder



© 2021 Exceed Robotics ™

5. You can delete the zip file if you want. It is recommended that you copy the PY01Files folder into a different location (preferably the desktop)
6. When you open the folder you should have 5 subfolders,
    a. **Week1**
    b. **Week2**
    c. **Week3**
    d. **Week4**
    e. **Week5to8**

© 2021 Exceed Robotics ™

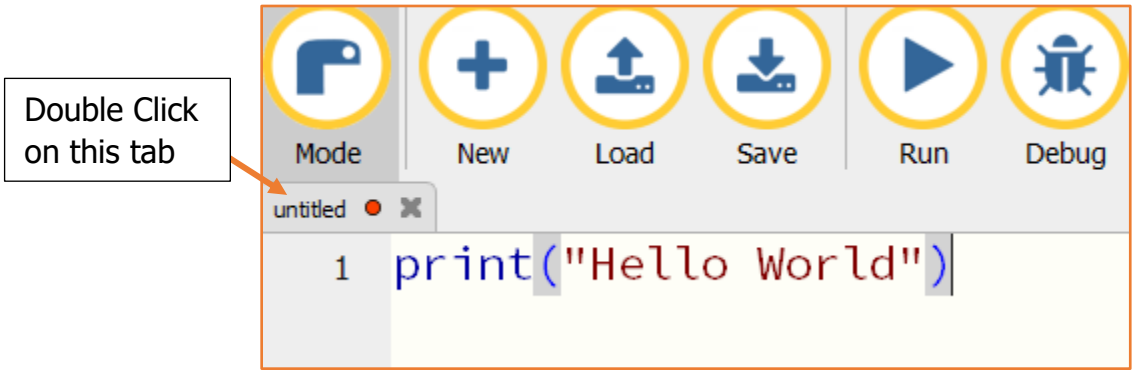
CREATING A FILE AND SAVING FILES

1. Open MU and create a new file by clicking on the "+" button.

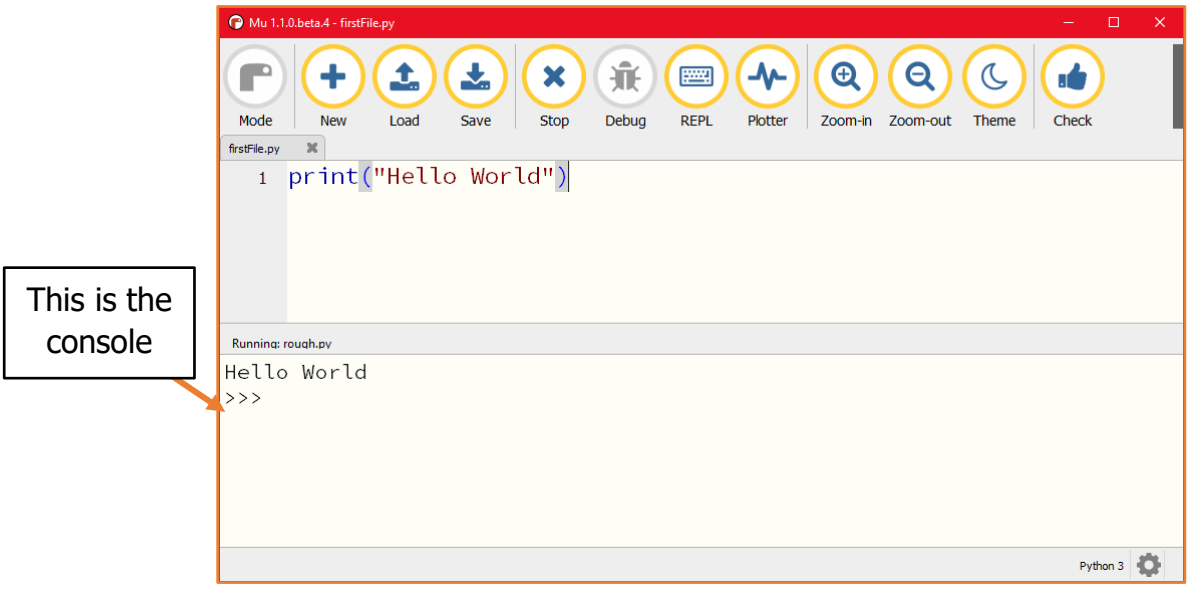
2. Copy and paste the code below into your file.

```
print("Hello World!")
```

3. To save the file, double-click on the file name tab.



4. Give your file a name and click on save. Make sure you save your file in the **PY01Exceed** folder you created before.
5. To run your program, click on the **Run** button. You should see the text **"Hello World!"** displayed on the console



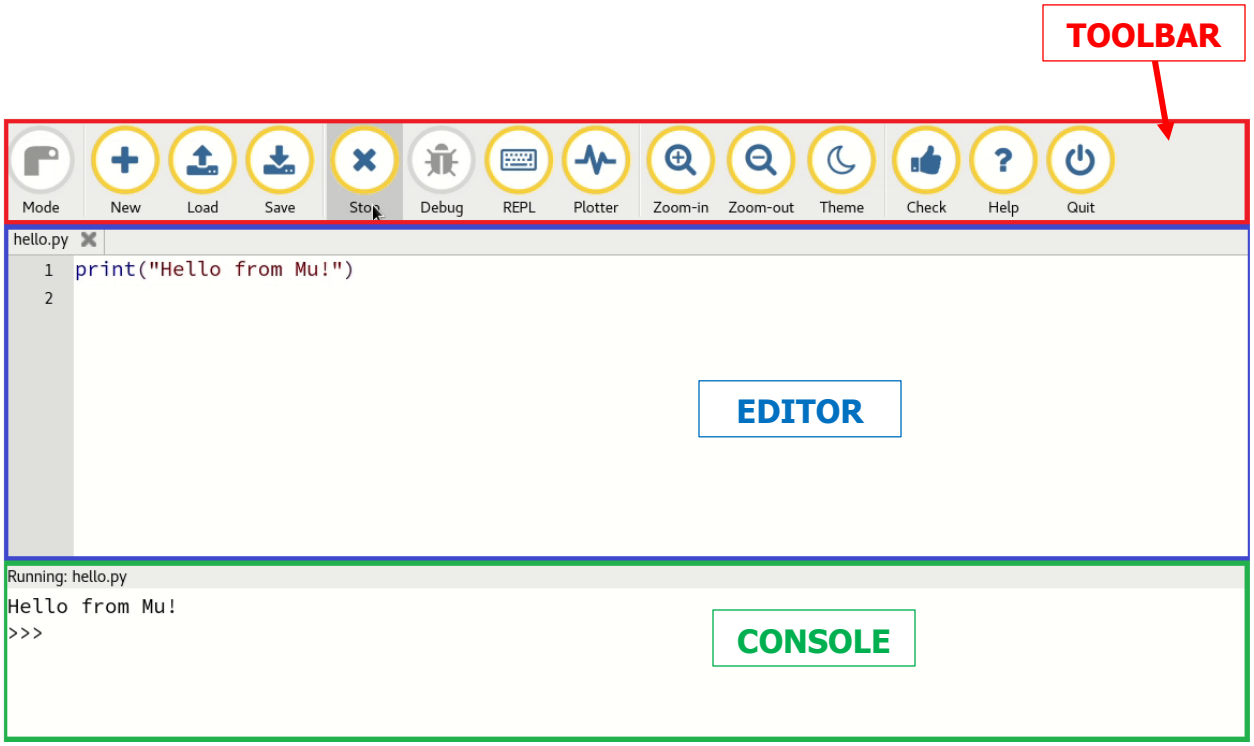
© 2021 Exceed Robotics ™

CLASS 1: VARIABLES AND IF STATEMENTS

MU OVERVIEW

MU is a simple editor for beginner Python programmers. The MU editor consists of 3 sections: Toolbar, Editor and Console.

- Toolbar: Consists of buttons for various operations. Some of the more important ones are New, Run, Stop, Load, Save and Zoom.
- Editor: This is the section where you will be writing your python program.
- Console: This is the area where all the outputs from your program will be displayed. This area is hidden until you run your file by clicking the Run button on the Toolbar.



© 2021 Exceed Robotics ™

INPUT/OUTPUT (I/O) FUNCTIONS

I/O FUNCTIONS

In Python, we have two built-in functions for I/O.

1. **print(str)**: This function can print the value stored inside the variable data to the console.

```
print(str)
```

2. **input(str)**: This function displays a prompt on the console. It then returns what the user inputs as a string.

```
input(str)
```

EXAMPLE: PRINT FUNCTION

Code

```
print("Hello World")
```

Explanation

This code prints the string *"Hello World"* to the console

EXAMPLE: INPUT FUNCTION

Code

```
myNumber = input('Enter a number between 1 and 10')
```

Explanation

This code asks the user to *Enter a number between 1 and 10*. The text that the user enters is then assigned to the variable *myNumber* as a string type.

© 2021 Exceed Robotics ™

VARIABLES

WHAT IS A VARIABLE?

In programming, a variable is a value that can change. By changing these values, we can have the program make decisions based on information that is gathered at run time (while the program is running).

You can think of a variable as a name attached to a particular object or value. In games, we will often pair variables with user input to store their actions, and then make decisions based on the results.

A variable consists of two parts: a name and a value. Once the variable is created (or declared) you can access the variable elsewhere in your program by calling its name.

VARIABLE NAMING RULES

There are some rules for naming variables

- Can include numbers and letters but no special characters
- Must start with a letter
- No spaces (underscore is allowed)
- Must be unique (no duplicates, even if they are different types)

TYPES OF VARIABLES

There are 4 basic data types in python. A value is an information stored inside the variable.

Data Type	Description	Examples
int	Integer	2, 6, -12, 0
str	String	'Hello', "Exceed"
bool	Binary Boolean	True, False, 1, 0
float	Real numbers	3.15, 2, -0.31, -15

© 2021 Exceed Robotics ™

TYPE CONVERSION

Sometimes, it may be useful to be able to change one data type to another. For example, the **input** function always stores data as a string. We may want to convert the text data to a numerical data type to do calculations or comparison

Conversion Functions	Description
int(data)	Converts to an integer
str(data)	Converts to a string
bool(data)	Converts to a boolean
float(data)	Converts to a float

Note: If you're converting from a string to an integer or float, make sure the string only contains numbers

You can use the **type(variable)** to find the type of value assigned to a particular variable.

REASSIGNMENT OF VARIABLES

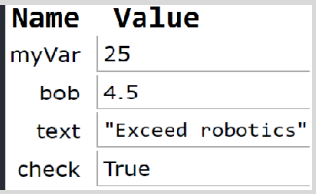
One of the main advantages of using variables in our program is that you can change the value stored inside the variable. For example, you can create a variable and set its value to the integer 5. You can later change the value stored in the variable to something else later. This is called **reassignment**.

© 2021 Exceed Robotics ™

EXAMPLE: CREATING VARIABLES

Code

```
#Program
myVar = 25 #Creates a integer variable
bob = 4.5 #Creates a float variable
text = "Exceed robotics" #Creates a string variable
check = True #Creates a bool variable
```



Name	Value
myVar	25
bob	4.5
text	"Exceed robotics"
check	True

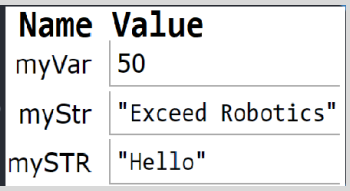
Explanation

This code is used to create 4 variables: myVar, bob, text and check. The figure on the right shows a visualization of the computer memory once the code is run.

EXAMPLE: REASSIGNING VARIABLES

Code

```
#Program
myVar = 25 #Creates a integer variable
myStr = "Exceed Robotics" #Creates a string variable
myVar = 25 + myVar #Updates the value of myVar
mySTR = "Hello"
```



Name	Value
myVar	50
myStr	"Exceed Robotics"
mySTR	"Hello"

Explanation

In the first line of this code, we create a variable called myVar whose value is set to 25. In the next line, we create another variable called myStr who is assigned the value "Exceed Robotics". In the next line, since myVar already exists, its value is changed to be equal to 25 + myVar which results in 50. In the next line, since mySTR does not exist, a new variable is created.

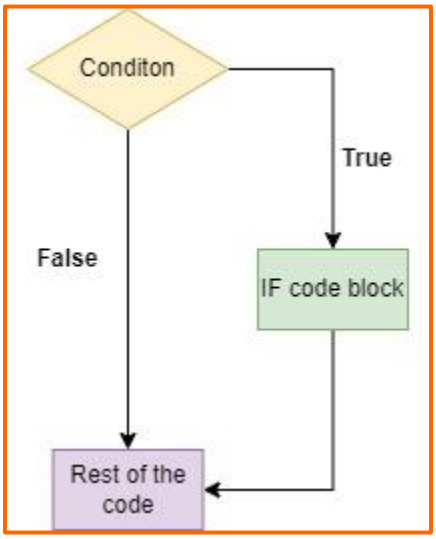
© 2021 Exceed Robotics ™

IF STATEMENTS

WHAT IS AN IF STATEMENT?

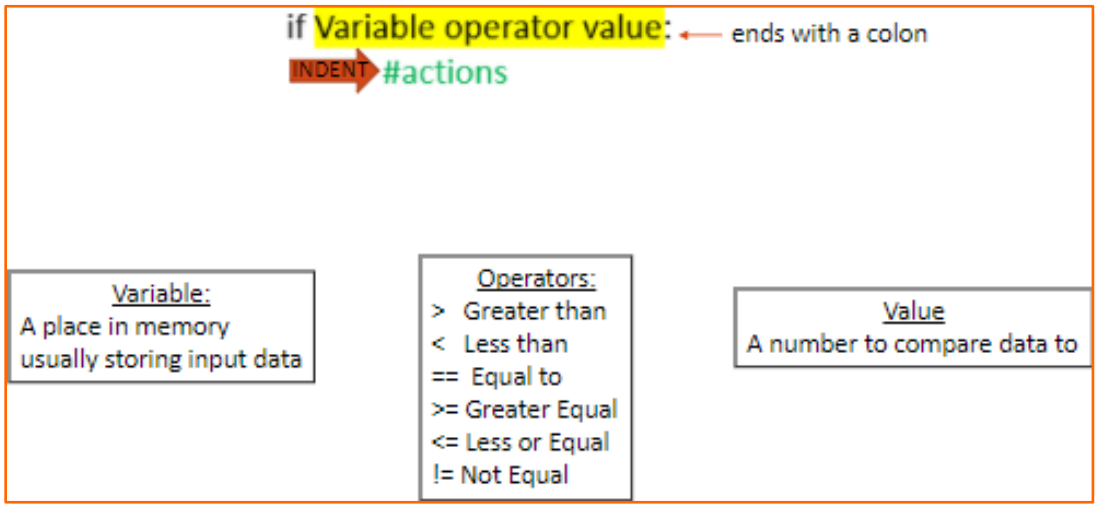
The **IF** statement tests the result of a logical expression/condition. If the logical expression equates to **True**, then the actions indented under the if statement is executed.

If the logical expression equates to **False**, then the actions under the if statement is skipped.



STRUCTURE OF AN IF STATEMENT

In Python, the syntax for writing a simple **IF** statement is,



© 2021 Exceed Robotics ™

EXAMPLE 1 (IF STATEMENTS)

Code

```
x = 3
if x < 10:
  print("Hello World")
```

Explanation

In this code, we first create a variable called x and assign it a value of 3. Next, we evaluate the logical expression x < 10. In this case, the expression is True, since 3 is less than 10. In this case, the statement under the if statement is executed. Therefore, Hello World is printed on the screen.

EXAMPLE 2 (IF STATEMENTS)

Code

```
x = 20
if x < 10:
  print("Hello World")
```

Explanation

In this code, we first create a variable called x and assign it a value of 20. Next, we evaluate the logical expression x < 10. In this case the expression is False, since 20 is not less than 10. In this case, the statement under the if statement is not executed. The program jumps out of the IF statement.

© 2021 Exceed Robotics ™

EXAMPLE 3 (IF STATEMENTS)

Code

```
myNumber = input("Enter a number bigger than 10")
myNumber = int(myNumber)

if myNumber > 10:
  print("Thank you!!!")
else:
  print("Sorry, that number is smaller than 10")
```

Description

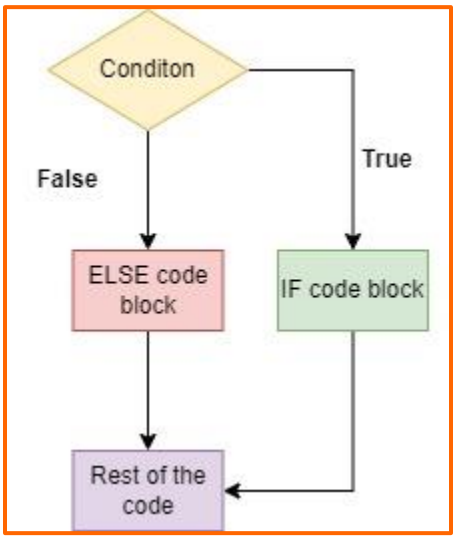
In this code, we first ask the user to enter a number bigger than 10. The text that the user enters is stored into the variable myNumber as a string. Next, we convert the string stored in myNumber to an integer and the assign it back to the variable myNumber.

In the IF statement, we evaluate the expression myNumber > 10. If the expression is True, then Thank You!!! Is printed. If the expression is False, then the else statement is executed. The text Sorry, that number is smaller than 10 is printed.

IF/ELSE STATEMENTS

An **IF/ELSE** statement can be used if you have two sets of actions: one set to be performed if the condition is **True** and another set which is to be performed if the condition is **False**.

© 2021 Exceed Robotics ™



EXAMPLE 3 (IF STATEMENTS)

Code

```
a = 5

if a < 10:
  print("a is a smaller than 10")
else:
  print("a is larger than 10")

if a > 3:
  print("a is bigger than 3")
else:
  print("a is smaller than 3")
```

Description

In this code, we have two If/Else statements. The first if statement checks if the value stored inside a is less than 10. Since this expression is False, the code in the else statement is performed i.e., "a is larger than 10" gets printed to the console. The next if statement checks if the value stored inside a is larger than 3. Since this expression is True, the code inside the if statement is performed i.e., "a is smaller than 3 is printed to the console.

© 2021 Exceed Robotics ™

COMMENTS

Comments are an extremely useful tool in programming. Comments enhance the readability of the code and help the programmers to understand the code. Comments do not affect the program. When python runs your file, the comments are skipped.

It is highly recommended that you comment on your code as it will help fellow coders understand what you are trying to accomplish. There are different ways to comment in python.

Single line comments are comments that only take up one line of space. Below is an example of a single-line comment.

```
# Create a variables A and B
A = 5
B = 10
# Variable C is the sum of A and B
C = A + B
# Print the value of C
print(C)
```

Multi-line comments are used when your comments are more than a few lines. Below is an example of a multi-line comment.

```
"""
The below code uses an if statement
The Statement checks if A is < 5
if True then "Yes" is printed
if False then "No" is printed
"""
A = 3
if (A < 5):
  print("Yes")
else:
  print("No")
```

© 2021 Exceed Robotics ™

IN-CLASS PROBLEMS

GROUP EXERCISE

Write a program that prompts the user to enter a username and password. If the username AND password match a preset pair, welcome the user. If the username and password are incorrect print "Login Failed"

INDEPENDENT WORK

Create a quiz Game!

Create a 5 question, multiple-choice quiz using a variable, if statements, and the Input/Print functions. The program should ask questions one at a time and tell the user whether the answer was correct.

EXTENSION

Extend your quiz game to include a score mechanic. The program should keep track of how many questions the user correctly answered. After all, the questions have been answered display the total score as a percentage and tell the user whether they passed

© 2021 Exceed Robotics ™

HOMEWORK

CLASS 1: VARIABLES AND IF STATEMENTS

MULTIPLE CHOICE QUESTIONS

1. The "**input**" function returns a _____ type value?
   - ☐ int
   - ☐ str
   - ☐ bool

2. Which of these is not a data type in python?
   - ☐ float
   - ☐ bool
   - ☐ decimals

3. Which of these is a valid variable name?
   - ☐ 1_variable
   - ☐ Variable_1
   - ☐ Variable 1

4. What happens when you run the following?

```
a = 5
if a > 6:
  print('Yeah, the IF statement executed')
```

   - ☐ Error
   - ☐ Nothing happens
   - ☐ The If statement executes (i.e., it prints: - **Yeah, the IF statement executed**)

5. What is printed when the following code is run?

```
Bob = 32
if Bob == '32':
  print('Yay, Bob is a string')
else:
  print('Bob is not a string')
```

⚠ Remember: To save the PDF once you are done answering the questions. Use **Ctrl+S** for windows or **Cmd+S** for MacOS

© 2021 Exceed Robotics ™

- ☐ Nothing is printed.
- ☐ **Yay, Bob is a string**
- ☐ **Bob is not a string**

6. What happens when the following code is run?

```
myNumber = input('Enter a number bigger than 10')
# I enter 25
myNumber = int(myNumber)
if myNumber > 10:
  print('Thank you!')
```

- ☐ There is an error
- ☐ **Thank you!**
- ☐ Nothing happens

7. What is printed when this code is run?

```
day = 'Thursday'
day = 32.5
Day = 19
print(day)
```

- ☐ **Thursday**
- ☐ **32.5**
- ☐ **19**

8. What are the values stored in variables A and B when this program is executed?

```
A = 15
B = A
A = 22
```

- ☐ A is 22 and B is 22
- ☐ A is 15 and B is 15
- ☐ A is 22 and B is 15

9. What is the value stored in variable A?

© 2021 Exceed Robotics ™

```
A = 12
A = A - 4
A = A + 6
A = A + 1
```

- [ ] 12
- [ ] 15
- [ ] 9

10. What is printed?

```
myNumber = input("Enter a number bigger than 10")
# I enter 25
print(type(myNumber))
```

- [ ] **<class 'str'>**
- [ ] **<class 'int'>**
- [ ] **25**

11. What is the correct way to create a comment in Python?

```
myNumber = input("Enter a number bigger than 10")
# I enter 25
print(type(myNumber))
```

- [ ] **//Here is a comment**
- [ ] **#Here is another comment**
- [ ] **/*Here is another comment*/**

12. Which of these statements are true in python? [Select All]

```
myNumber = input("Enter a number bigger than 10")
# I enter 25
print(type(myNumber))
```

- [ ] Comments are used to tell python, what is happening
- [ ] Comments can be a single line or multi-line
- [ ] Comments make your code more easily readable

© 2021 Exceed Robotics ™

TASK-BASED PROBLEMS

TASK A

NOTE: please make sure you download and extract the **PY01Files** folder. Click here if you haven't done so already

Open the MU editor. Load the file "**Week1TaskA-BiggerNumber"**. It should be in the **Week1** folder of your **PY01Files** folder.

Suppose you have two int variables A and B. Your task is to write a program that finds how the variables relate.

- If A is bigger than B: print ("A is bigger than B")
- If B is bigger than A: print ("B is bigger than A")
- If A is equal to B: print ("A and B are equal")

Task B

NOTE: please make sure you download and extract the **PY01Files** folder. Click here if you haven't done so already

Open the MU editor. Load the file "**Week1TaskB-HogwartsGrader"**. It should be in the **Week1** folder of your **PY01Files** folder.

Hogwarts school of witchcraft and wizardry has the following grade system:

- Above 90: Outstanding (O)
- 80 – 89: Exceeds Expectations (E)
- 70 – 79: Acceptable (A)
- 60 – 69: Poor (P)
- 50 – 59: Dreadful (D)
- Below 49: Troll (T)

Your task is to write a program that implements this grading system. The program should ask the user to enter a score. It should then print out the correct grade. For example, if a score of 85 is entered, then the following should be printed "Exceed expectations".

© 2021 Exceed Robotics ™

TASK C

NOTE: please make sure you download and extract the **PY01Files** folder. Click here if you haven't done so already

Open the MU editor. Load the file "**Week1TaskC-SquareChecker"**. It should be in the **Week1** folder of your **PY01Files** folder.

A square is a rectangle that has equal height and breadth.

Your task is to write a program that asks the user for two values, first the height and then the breadth. Using these two values you should check if the shape is a square. If the shape is a square, then print "Yes, the shape is a square" otherwise print "No, it is not a square".

TASK D

NOTE: please make sure you download and extract the **PY01Files** folder. Click here if you haven't done so already

Open the MU editor. Load the file "**Week1TaskD-EvenNumbersOnly"**. It should be in the **Week1** folder of your **PY01Files** folder.

Write a program to check if a number entered is even? Remember an even number is one that is fully divisible by 2. You can use the modulus operator (%) to calculate the remainder. This has been done for you.

Add statements that checks if the value stored in remainder is 0 or not. If remainder is 0 then, print *"Yes the number is an even number"* otherwise *"No the number is an odd number"*

© 2021 Exceed Robotics ™

FIND THE ERROR

In this section, the objective is to find the error in the code snippet. You need to write down what the error is and rewrite the code with the change.

PROBLEM A

Open the file "**Week1ProblemA-PrintingError"**. Look at the code in the file.

The code should print **Hello World!** But there is **one** error that causes the program to crash. Can you find the error and fix it? Write down what was wrong in the box.

PROBLEM B

Open the file "**Week1ProblemB-AreaCalculator"**. Look at the code in the file.

The code should print the area of square. The length of one side of the square is 5 and it is stored in the variable ***lengthSquare***. However, it doesn't work properly. It should be printing "The area of the square is 25". Can you find the error and fix it?

PROBLEM C

Open the file "**Week1ProblemB-IfError"**. Look at the code in the file.

The code uses if statements. There is **one** error that causes the program to crash. Find it and fix it.

© 2021 Exceed Robotics ™

CLASS 2: DRAWING AND TUPLES

TUPLES

WHAT IS A TUPLE?

In Python, a Tuple is a data type that can store multiple values. We store the data as a list enclosed by brackets.

For example, the code below creates a tuple with 5 items.

```
y = (5,2,'Orange',False,23.43)
```

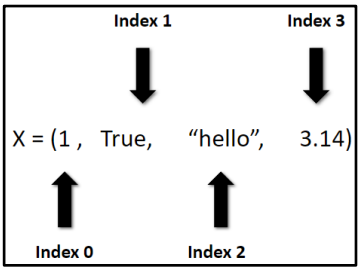
A tuple is immutable, which means the values of the items inside the tuple cannot be changed. To find the length of a tuple, you can use the **len(tuple)** function.

TUPLE INDEXING

The code below creates a tuple with 4 items.

```
x = (1, True, 'Hello', 3.14)
```

To access specific elements in a tuple, we can use **indexing**. Each item inside the tuple is assigned an index. Index number starts from 0.



We can access an element by typing the name of the tuple, followed by the index surrounded by square brackets.

For example,

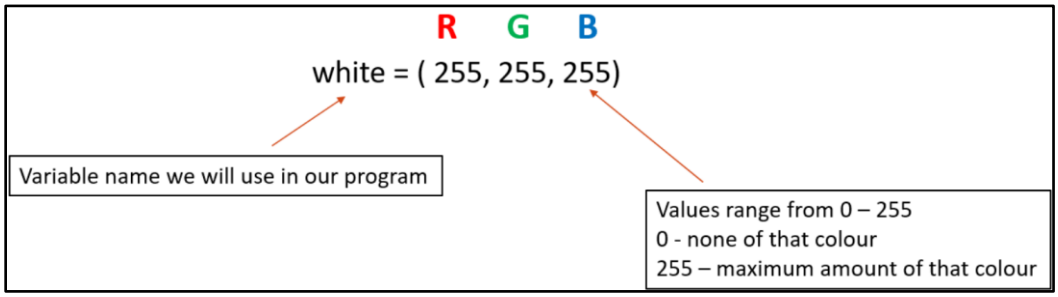
X [2] gives you the string *"Hello"*

© 2021 Exceed Robotics ™

COLORS AND PIXELS

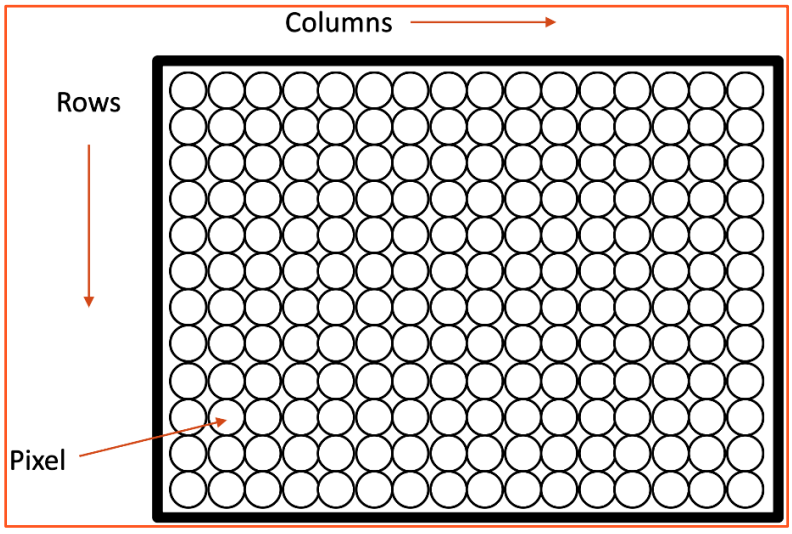
RGB TUPLES

In computer science, colours are described by R, G, and B values. In Python, we will store these values using tuples. An RGB tuple should only have items



PIXELS

Every computer screen is made up of thousands of tiny dots called pixels. Pixels are arranged in rows and columns. Each pixel can be assigned an RGB colour. Images are created, by changing the colours assigned to different pixels.

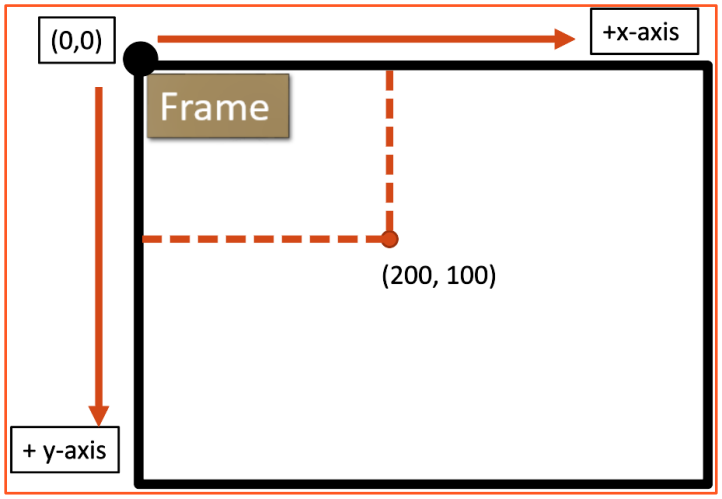


© 2021 Exceed Robotics ™

PYGAME DRAWING FUNCTIONS

CARTESIAN COORDINATES

In Pygame, we are going to use the cartesian coordinate system. However, there are a couple of differences. Here, the origin will always be the top-left corner and the y-axis increases from top to bottom.



Points on the screen will be addressed using their pixel coordinates (x,y)

CREATING A SIMPLE GAME WINDOW IN PYTHON

To create a simple window in pygame, we can run the following code.

```
import pygame as pg #Imports the pygame library
pg.init() #Starts the pygame tools
screen = pg.display.set_mode((800,600)) #Screen size
```

This code creates a screen that has a width of 800px and a height of 600px.

BACKGROUND COLOR

Once, we have created a screen. We can change the colour of the entire screen. To change the background colour of the screen we use the function,

```
screen.fill(color)
```

© 2021 Exceed Robotics ™

EXAMPLE 1: CREATING A SCREEN AND ADDING A BACKGROUND COLOR

Code

```
import pygame as pg
pg.init()
screen = pg.display.set_mode((800,600)) #Create a screen 800px X
600px

blue = (0,0,255)    #RGB tuple for blue
screen.fill(blue)   #Change the color of the screen to blue
pg.display.flip()   #Update the screen
```

Description

In this code, we first create a screen which has a width and height of 800px and 600px respectively. Next, we create a variable to store the RGB value for blue. Next, we change the color of the screen to blue. Finally, we update the game.

DRAWING A RECTANGLE

To create a rectangle, we can use the function,

```
pg.draw.rect(screen, color, (x, y, w, h))
```

DRAWING AN ELLIPSE

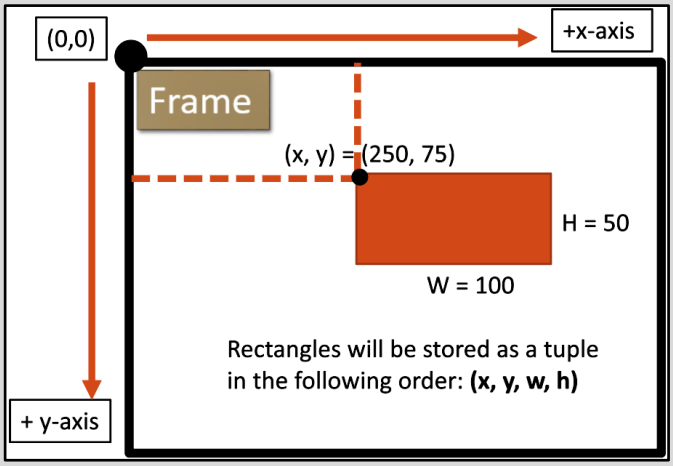
Ellipses work similarly to rectangles. The ellipse drawn is the largest one that would fit in the bounding rectangle. To create an ellipse, we can use the function,

```
pg.draw.ellipse(screen, color, (x, y, w, h))
```

© 2021 Exceed Robotics ™

EXAMPLE 2: DRAWING A RECTANGLE

In this example, let's create the rectangle below.



Code

```
white = (0,0,255)   #RGB TUPLE FOR white
red = (0,0,255) #RGB TUPLE FOR red
screen.fill(white)  #CHANGE BACKGROUND COLOR TO white
pg.draw.rect(screen,red,(250,75,100,50)) #Draw the rectangle
pg.display.flip()   #UPDATE THE SCREEN
```

Description

Here we create 2 RGB tuples for white and red. Next, we change the background color of the screen to white. Next, we draw the rectangle with an x, y, w and h of 250, 75, 100, 50. The color of the rectangle is set to red. Finally, we update the screen.

DRAWING A HOLLOW RECTANGLE OR ELLIPSE

An extra thickness (**T**) argument can be added to both the ellipse and the rectangle functions to create hollow ellipses and rectangles respectively.

```
pg.draw.ellipse(screen, color, (x, y, w,
```

```
pg.draw.rect(screen, color, (x, y, w,
```

© 2021 Exceed Robotics ™

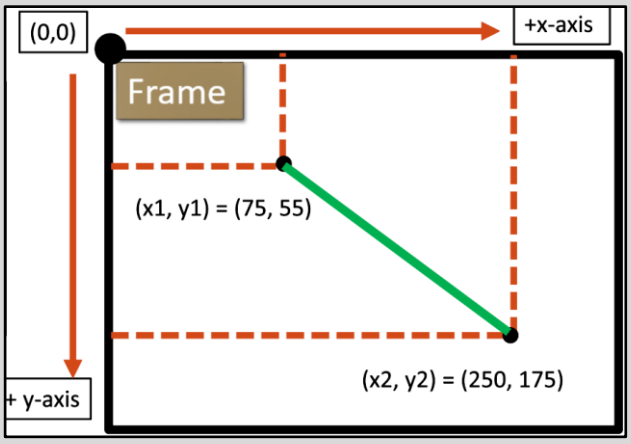
DRAWING A LINE

To create a line, we can use the function,

```
pg.draw.line(screen, color, (x1, y1),(x2,y2),
```

EXAMPLE 3: DRAWING A LINE

In this example, let's create the line shown below.



Code

```
white = (255, 255, 255) #RGB tuple for white
green = (0, 255, 0) #RGB tuple for green
screen.fill(white)  #Change the color of the screen to blue
pg.draw.line(screen, green, (75,55),(250,175),10) #Draw the line
pg.display.flip()   #Update the screen
```

Description

Here we create 2 RGB tuples for white and green. Next, we change the background color of the screen to white. Next, we draw the line with an x1, y1, x2 and y2 of 75, 55, 250 and 175 respectively. The color of the line is set to green. Finally, we update the screen.

© 2021 Exceed Robotics ™

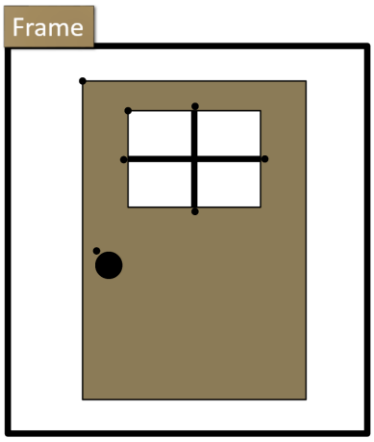
IN-CLASS PROBLEMS

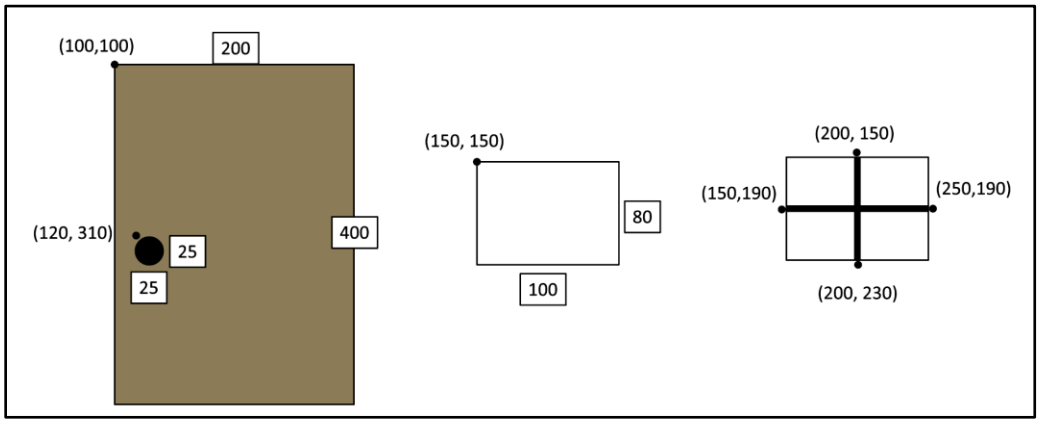
GROUP EXERCISE

As a group, write code to draw the door using the following:

- 2 rectangles
- 2 lines
- 1 ellipse

A cheat sheet with all the functions is provided at the end of the handout.





INDEPENDENT EXERCISE

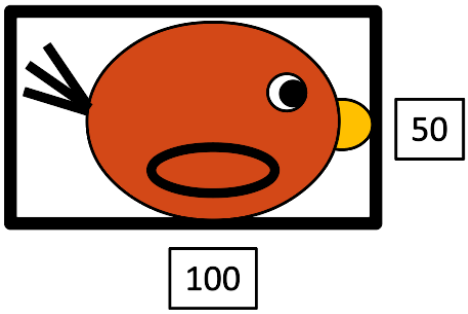
At the end of the term, everyone will be creating their version of a flappy bird game!

Using the drawing tools learned today, create your character to be later used in your flappy-bird style game.

Note: It doesn't have to be a bird! You can think of other kinds of characters that would fit in well in this game.

© 2021 Exceed Robotics ™

Hint: You may want to shrink your screen size to (100,50). This will make it easier to add to the game later.



© 2021 Exceed Robotics ™

HOMEWORK PROBLEMS

CLASS 2: DRAWING TOOLS

MULTIPLE CHOICE QUESTIONS

1. The correct way to create a tuple is

   - ☐ myTuple = [123 , 321 , '23' , 'RED']
   - ☐ myTuple = (123 , 321 , '23' , 'RED')
   - ☐ myTuple = {123 , 321 , '23' , 'RED'}

2. Tuples are not mutable. What does that mean?

   - ☐ Tuples cannot be changed
   - ☐ Tuples cannot be printed
   - ☐ Tuples are quiet

3. True/False: Tuples can only contain other tuples.

   - ☐ True
   - ☐ False
   - ☐ Don't know

4. True/False: Tuples can only contain int values.

   - ☐ True
   - ☐ False
   - ☐ Don't know

5. Consider the code below. What item is in the gets printed?

```
myTuple = ('Bob',213,'Umbrella',(923,1,32),'Big')
print(myTuple[2])
```

   - ☐ **213**
   - ☐ Error
   - ☐ **Umbrella**

© 2021 Exceed Robotics ™

6. Consider the code below. What gets printed?

```
myTuple = ('Bob',213,'Umbrella',(923,1,32),'Big')
print(myTuple[3])
```

- ☐ **(923,1,32)**
- ☐ **Umbrella**
- ☐ **923**

7. What is a valid variable name?

- ☐ Bob123
- ☐ 123Bob
- ☐ Bob#asd

8. The range of values for RGB is …….?

- ☐ 0 – 215
- ☐ 0 – 255
- ☐ 0 – 512

9. What is the RGB for white?

- ☐ (0,0,0)
- ☐ (215,215,215)
- ☐ (255,255,255)

10. In Python, a variable may be assigned a value of one type, and then later assigned a value of a different type?

- ☐ True
- ☐ False
- ☐ Don't know

11. What happens when the following code is run?

© 2021 Exceed Robotics ™

```
myNumber = input('Enter a number bigger than 10')
# I enter 25
myNumber = int(myNumber)
if myNumber > 10:
  print('Thank you!')
```

- ☐ There is an error
- ☐ **Thank you!**
- ☐ Nothing happens

12. Consider the code below. Which of these logical expressions would return **True**? [Select all]

```
myTuple = ('Bob',213,'Umbrella',(923,1,32),'Big')
print(myTuple[2])
```

- ☐ **len(myTuple) == 5**
- ☐ **myTuple [1]! = "Bob"**
- ☐ **myTuple [3][1] == 1**

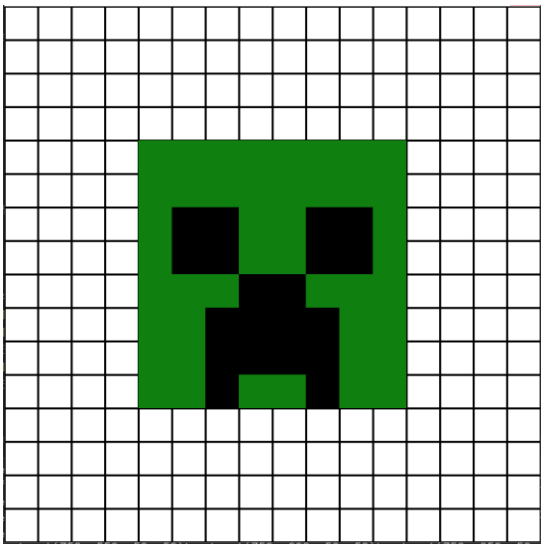
⚠

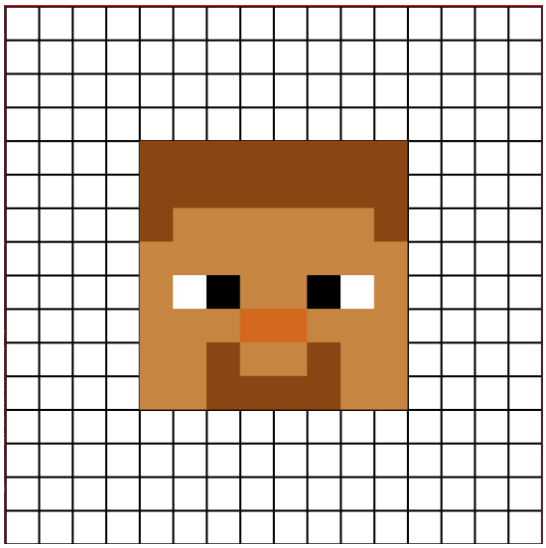
Remember: To save the PDF once you are done answering the questions. Use **Ctrl+S** for windows or **Cmd+S** for MacOS

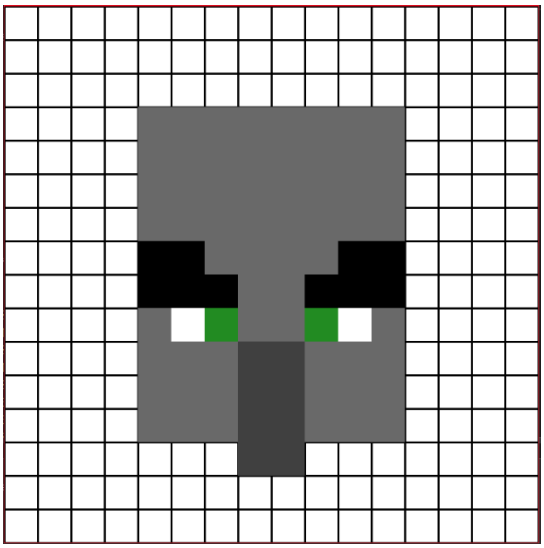
© 2021 Exceed Robotics ™

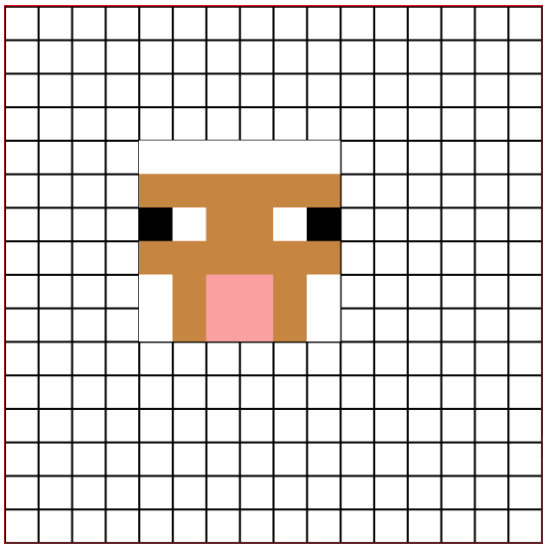
DRAWING CHALLENGE

Mojang Studios, the developer of Minecraft, has hired you as a game character developer. Your responsibility is to draw the faces of some Minecraft characters using the drawing tools used in class. Note: that the size of the grid is 50x50 i.e., each small square has a height of 50 and a width of 50. The screen size should be set to 800 x 800.









© 2021 Exceed Robotics ™

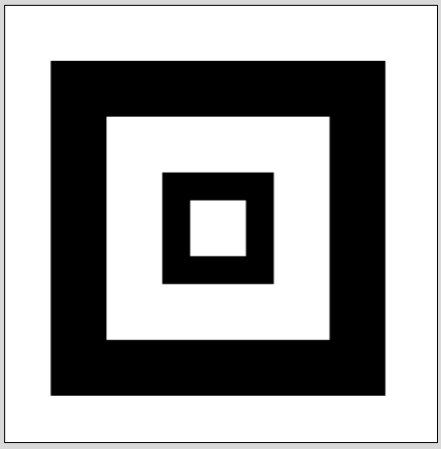
FIND THE ERROR (TROUBLESHOOT)

PROBLEM A

Open the file "**Week2ProblemA-Logo"**. It should be in the **Week 2** subfolder of **PY01Files**.

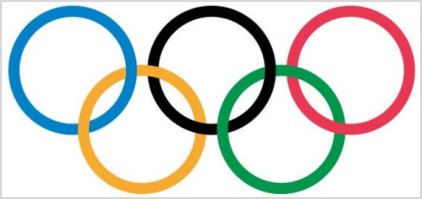
Look at the code in the file. The code is supposed draw a nice logo. However, when you run the program only a black screen appears. Find out what is wrong. There are **two** mistakes that's causing the game to not work properly. In the box below list the two errors.

Desired output



© 2021 Exceed Robotics ™

PROBLEM B



Open the file as "**Week2ProblemB-OlympicRings"**. The code in the file is supposed to draw the Olympic rings. However, the red ring doesn't show. Find the issue and fix it. There are **two** errors.

Problem C

Open the file "**Week2ProblemC-SumOfATuple"**.

The code in the file is supposed to add up all the items in the tuple and store the final sum into the variable finalSum. However, there is an error. Find the error and fix it. There are **two** errors.

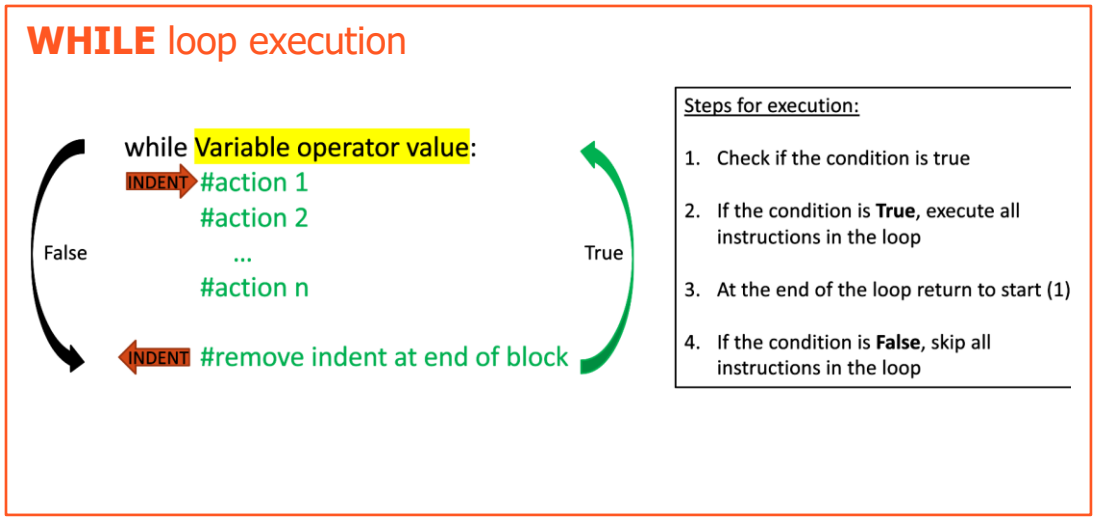
© 2021 Exceed Robotics ™

CLASS 3: WHILE LOOPS & MOVEMENTS

WHILE LOOPS

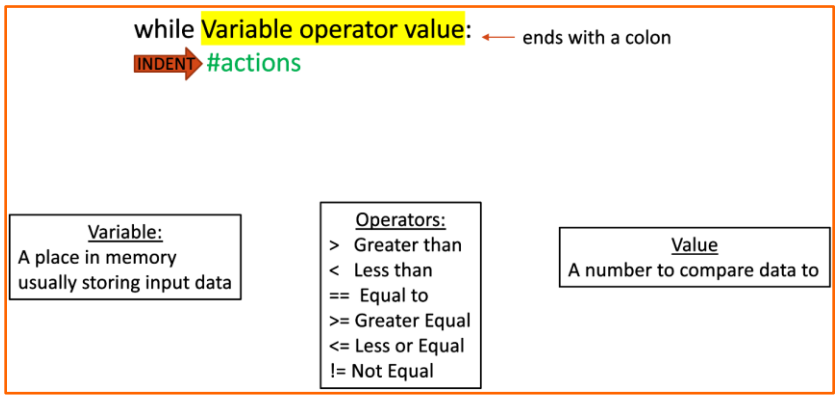
WHAT IS A WHILE LOOP?

A **While** loop tests the result of a logical expression. If the logical expression equates to **True**, then the actions indented under the while loop are executed. Once the actions are completed, the logical expression is checked again. If it still returns **True**, the actions indented under the while loop are executed. This is repeated until the logical expression becomes **False**.



WHILE LOOP STRUCTURE

In Python, the syntax to write a while loop is,



© 2021 Exceed Robotics ™

INCREMENTING/DECREMENTING

One of the most common operations in computer science is to increase/decrease the values stored inside a variable. Hence, there are various shortcuts for this operation in python.

To increase the value of a variable you can use:

- **number = number + 1**
- **number += 1**

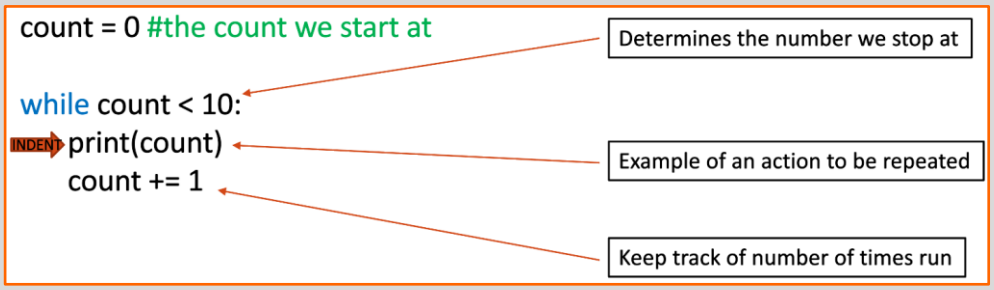
To decrease the value of a variable you can use:

- **number = number – 1**
- **number += -1**
- **number -= 1**

EXAMPLE 1: WHILE LOOPS EXAMPLE

A common usage of loops and iterating is a counting loop. In this example, we create a loop that runs a set number of times.

Code



Description

In this code, we first create a variable called count which has a value of 0. Next, we enter the while loop where the logical expression is checked. Since count (which is 0) is less than 10, the expression is True. The actions inside the while loop are executed: the value of count is printed, and the count is increased by 1. The logical expression is checked again, and since it is still True, the actions are executed again. This repeats until, eventually, the value of count is 10. At this point, the logical expression is False. The program jumps out of the while loop.

© 2021 Exceed Robotics ™

EXAMPLE 2: WHILE LOOPS & TUPLES

Another use of loops is to go over all the items in a tuple. In this example, we will use a tuple containing the names of students and a while loop to print a greeting for each student.

Code

```
nameTup = ("Harry", "Roberta", "Ali", "Rohit", "Marcus", "Victoria")
counter = 0

while counter < 6:
  currentName = nameTup[counter]
  print("Hello " + currentName)
  counter += 1
```

Console

```
>> Hello Harry
>> Hello Roberta
>> Hello Ali
>> Hello Rohit
>> Hello Marcus
>> Hello Victoria
```

Description

In this example, we have a tuple called nameTup which contains 6 names. We can use the while loop counter to index the tuple. We start by creating a counter variable that begins at 0. We know that the last item in the tuple ("Victoria") has an index of 5, therefore we want to make sure that loop terminates when the counter goes above 5. This is important as it will cause an error otherwise. Inside the variable we have a temporary variable that stores the name that is in the current index of the tuple. We also write a print statement that adds the string "Hello "before the current name and outputs the string to the console. Remember to increment your counter variable, otherwise your code will crash as python will enter a forever loop.

Here is a couple of questions to think about.

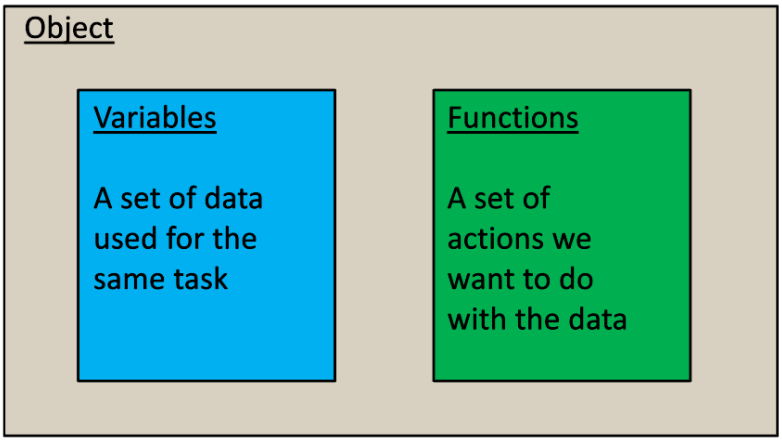
1. Instead of printing all the names, what if you want to print every other name.
2. Can I reverse the order of the names without editing the tuple?

© 2021 Exceed Robotics ™

OBJECTS

WHAT IS AN OBJECT?

In programming, an object is a collection of variables and functions grouped under a single name. This allows us to group variables that have similar purposes together along with functions that work with them.



THE PG. RECT OBJECT

The first object that we are going to learn about in python is a pygame rectangle object. This object will combine a list of variables used to draw and keep track of the rectangle with actions that can be used to detect clicks and collisions in games.

The **pg. Rect** objects can be used to store the x,y,w and h values of a particular object. The code also shows you how to access information inside the object. One such information we can find is whether the rectangle collides with a point on the screen. This can be done using,

```
name.collidepoint(x,y)
```

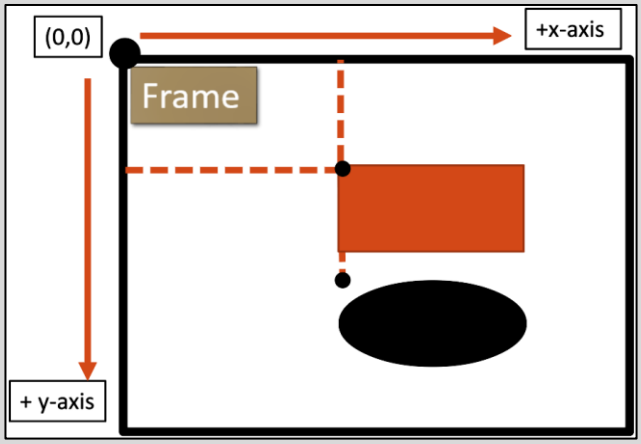
The "dot operator" means "access data of".

```
red = (255,0,0) #RGB for red
#Creates a pg.Rect object
rectangle1 = pg.Rect(0,0,50,50)
#Checks if rectangle collides with point (10,10)
rectangle1.collidepoint(10,10)
#We can use the rectangle object to draw the rectangle
pg.draw.rect(screen,red,rectangle1)
```

© 2021 Exceed Robotics ™

EXAMPLE 2: USING PG. RECT OBJECT

Let's say we want to draw the following but using the **pg. Rect** object.



Code

```
red = (255, 0, 0) #RGB for red
black = (0, 0, 0) #RGB for black
white = (255, 255, 255) #RGB for white

myRect = pg.Rect(250,175,100,50)
myOtherRect = pg.Rect(250,175,100,50)

screen.fill(white)
pg.draw.rect(screen,red,myRect)
pg.draw.ellipse(screen,red,myOtherRect)
pg.display.flip()
```

© 2021 Exceed Robotics ™

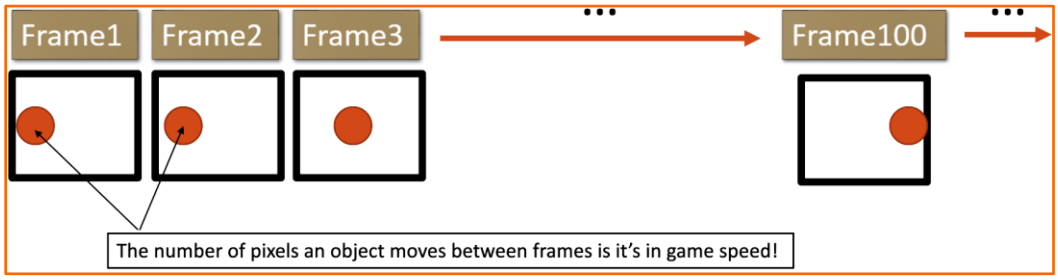
FRAMES AND MOVEMENTS

WHAT ARE FRAMES AND FPS?

A game is simply a series of frames played one after the other. The speed at which we go from one frame to another frame is controlled using **FPS** (Frames Per Second).

HOW DO YOU MOVE OBJECTS IN A GAME?

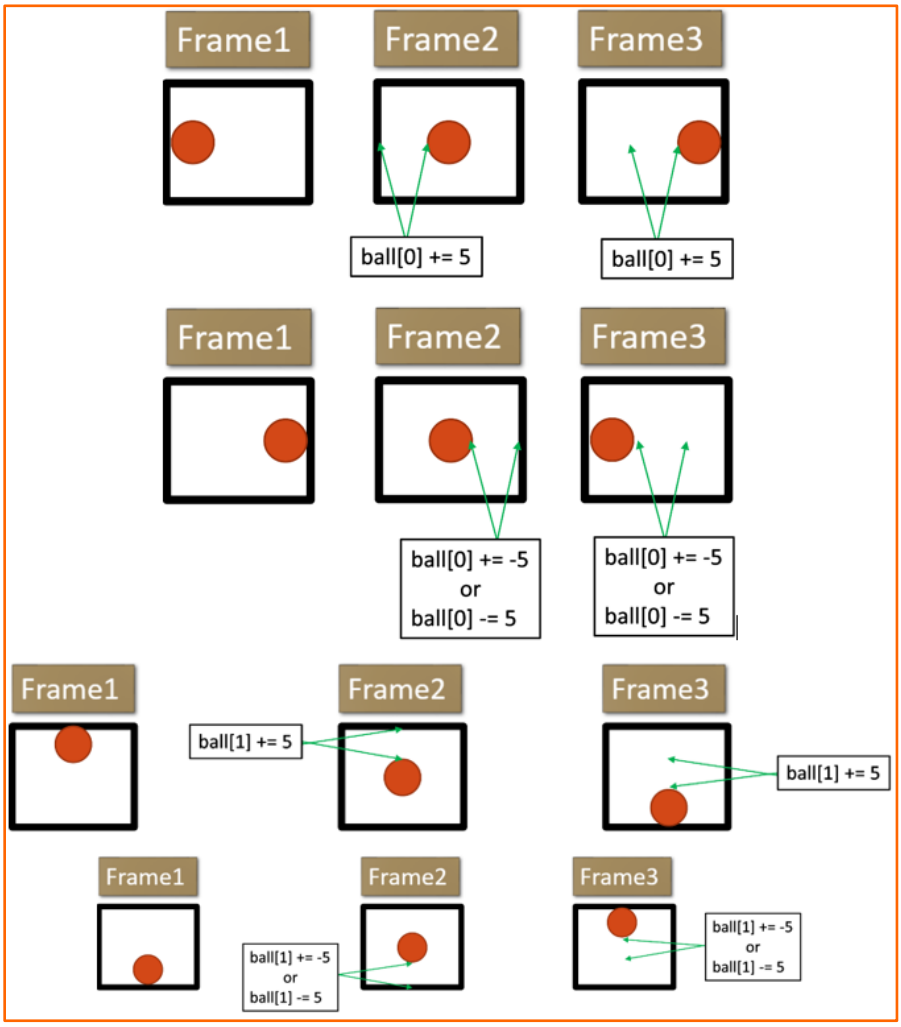
Movement in a game is achieved by drawing an object in one position in one frame. Then drawing the object in a slightly different position in the next frame.



To move an object in the,

- **Right direction**: Increase the x-coordinate of the object.
- **Left direction**: Decrease the x-coordinate of the object.
- **Up direction**: Decrease the y-coordinate of the object.
- **Down direction**: Increase the y-coordinate of the object.

© 2021 Exceed Robotics ™



© 2021 Exceed Robotics ™

STARTER TEMPLATE

Shown below is the starter template that we will use for all the games that we will build from PY1 to PY4.

```
#Loads the pygame library
import pygame as pg
#Starts the tools in pygame
pg.init()
#Creates a game screen that is 800px wide and 800 px
#high
screen = pg.display.set_mode((800,800))

#Color: Define your color variables here

#Objects: Define your pg.Rect objects here

#Game variables: Define your game variables here

#Main game loop
while True:
  #1.Inputs
  #Check input devices

  #2.Updates
  #Move objects to new locations based on the speed

  #3. Events
  #Set of if statements for different events such as
  #colliding etc

  #4. Drawing
  #Draw the current state of everything in the game

  #5. Clock
  #Set how many frames per second the game will run
  #at
```

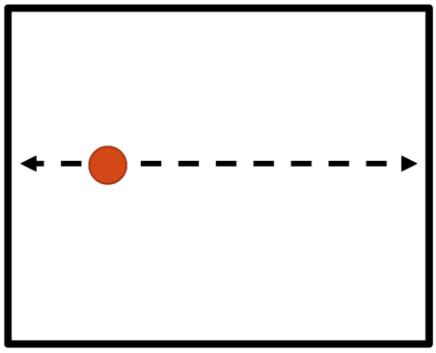
© 2021 Exceed Robotics ™

IN-CLASS EXERCISES

INDEPENDENT EXERCISE

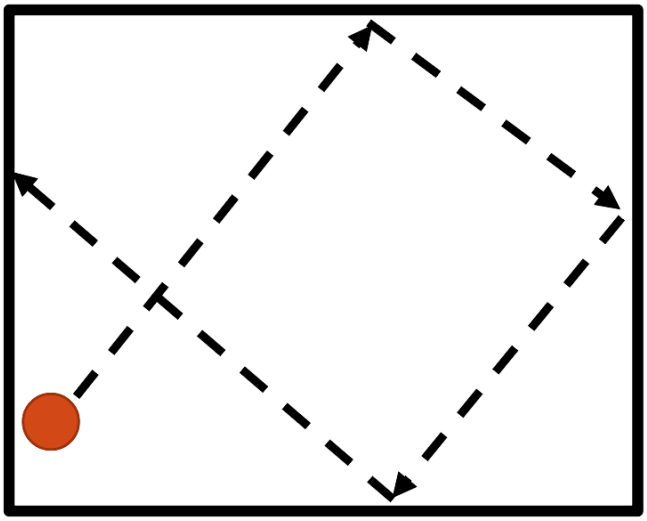
Using a **pg. Rect** and the setup for a standard game loop, write a program that makes a ball bounce back and forth the left and right sides of the screen.

Hint: since the speed is going to change, use a variable



EXTENSION

Modify the program to make the ball move in two dimensions, bouncing off the top, left, bottom and right edges of the screen.



© 2021 Exceed Robotics ™

HOMEWORK

CLASS 3: WHILE LOOPS

MULTIPLE CHOICE QUESTIONS

1. What is the difference between If statements and while loops?

   - ☐ An if repeats action a specific number of times. A while loop does its actions once.
   - ☐ A while repeats actions for a specific amount of time. An if statement performs its actions only once.
   - ☐ An if statement performs its actions if the logical expression is **True**. A while loop runs its actions if the logical expression is **False**.

2. What does the code below output?

```
A = ('Apple', 'Orange', 'Lemon', 'Strawberry')
count = 0
while count < len(A):
    print(A[count])
    count = count + 1
print('Printing complete')
```

   - ☐ Nothing. There is an Error!
   - ☐ **Printing complete**
   - ☐ **Apple**
     Orange
     Lemon
     Strawberry
     Printing Complete

3. What are valid python ways to increase the value stored in x by 1 (i.e., increment)? (Select All)

   - ☐ x ++
   - ☐ x = x +1
   - ☐ x += 1

4. What are valid python ways to decrease the value stored in x by 1 (i.e., decrement)? (Select all)

© 2021 Exceed Robotics ™

- ☐ x += -1
- ☐ x = x - 1
- ☐ x -= 1

5. What does the code below output?

```
A = ('Apple', 'Orange', 'Lemon', 'Strawberry')
count = 0
while count > len(A):
    print(A[count])
    count = count + 1
print('Printing complete')
```

- ☐ Nothing. There is an Error!
- ☐ **Printing complete**
- ☐ **Apple**
  Orange
  Lemon
  Strawberry
  Printing Complete

6. How many times will this while loop be executed?

```
count = 0
while count < 10:
    print(count)
    count += 1
```

- ☐ 10 times
- ☐ Infinite times
- ☐ 0 times

7. How many times will this while loop be executed?

```
count = 0
while count == 10:
    print(count)
    count += 1
```

- ☐ 10 times

© 2021 Exceed Robotics ™

- ☐ Infinite times
- ☐ 0 times

8. How many times will this while loop be executed?

```
count = 0
while count < 10:
    print(count)
```

- ☐ 10 times
- ☐ Infinite times
- ☐ 0 times

Remember: To save the PDF once you are done answering the questions. Use **Ctrl+S** for windows or **Cmd+S** for MacOS

TASK-BASED PROBLEMS

TASK A

Open the file "**Week3TaskA-AddingInTuple.** Write a program that sums all the numbers in the tuple A.

TASK B

Open the file "**Week3TaskB-EvenInATuple"**. Write a program that counts the number of even numbers in a tuple.

TASK C

Open the file "**Week3TaskC-WalmartReciept"**. Walmart wants you to use your coding skills. They have provided you with 2 tuples with equal number of items. One tuple contains the names of items sold at their stores. The other contains prices of the items. Your job is to write a program that prints both the name and price of the item together. For example, a Milky Way bar costs 1.5. For this item we should print **'The price of Milky Way is $1.5'**.

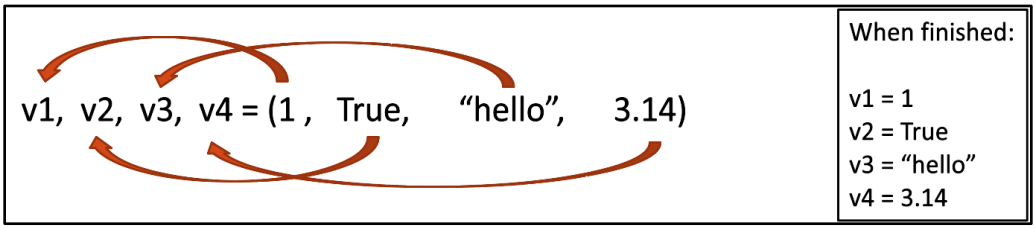
CLASS 4: MOUSE INPUT

TUPLES UNPACKING

HOW CAN YOU UNPACK A TUPLE?

Tuples are a convenient and compact way of storing and working with a list of data. However, sometimes it might be easier or more desired to work with elements individually. If we want to separate the data stored in a tuple, we can unpack it.

Unpacking a tuple requires one variable for each element in the tuple. For example, if we have a tuple with 10 items then we require 10 variables to unpack the tuple.



BOOLEAN LOGIC

HOW TO COMBINE LOGICAL EXPRESSIONS?

Sometimes, when writing logical expressions for a **while** loop or an **if** statement, we may need to combine expressions. This can be done using **and/or**.

For example, a person eats dinner if he/she is hungry AND it’s 6 pm. In this case, the logical expression can be written as,

The person is hungry **AND** the time is 6 pm

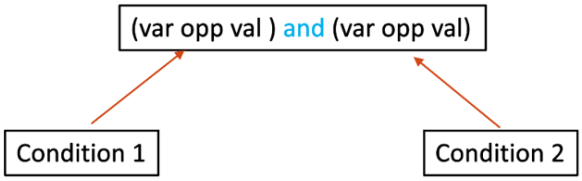
Sometimes only one of the expressions needs to be true. In this case, we use an OR.

The person is hungry **OR** the time is 6 pm

© 2021 Exceed Robotics ™

AND LOGIC

When using the **AND** logic both the statements must be **True**.

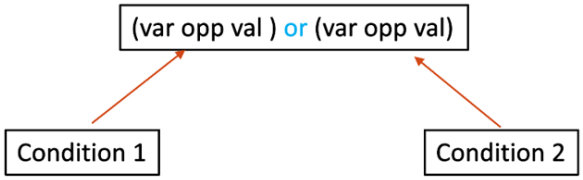


Truth Table - AND

Condition1	Condition 2	Result
F	F	F
F	T	F
T	F	F
T	T	T

OR LOGIC

When using the **OR** logic at least one of the statements must be **True**.



Truth Table - OR

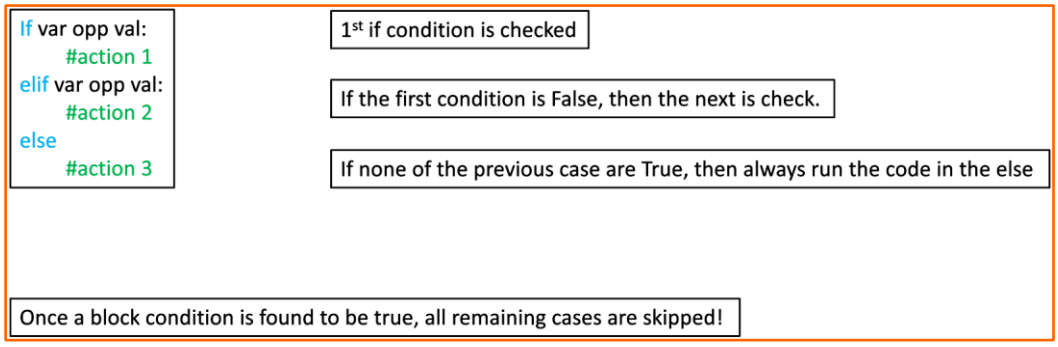
Condition1	Condition 2	Result
F	F	F
F	T	T
T	F	T
T	T	T

© 2021 Exceed Robotics ™

COMBINED IF STATEMENTS

COMPLETE IF STATEMENT STRUCTURE

It is also possible to link **IF** statements together. In this case, we check them one at a time, looking for the first True one. Once one condition is **True**, the rest is skipped. This is very useful when the answer to previous cases could impact the next.



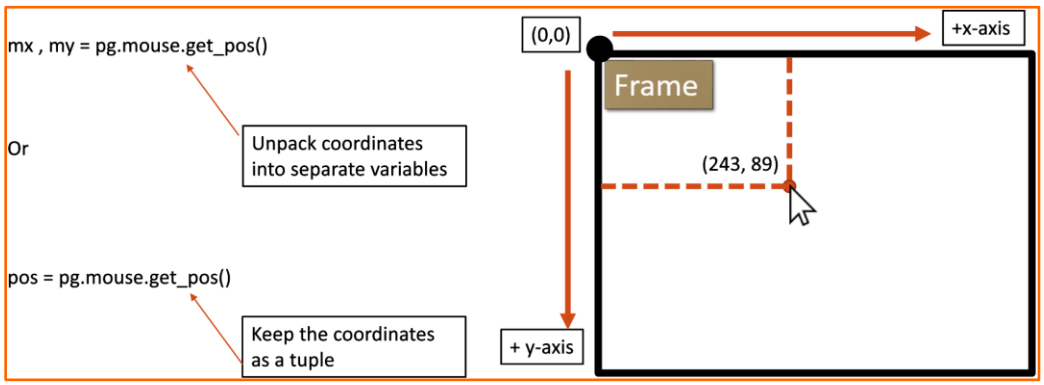
© 2021 Exceed Robotics ™

MOUSE INPUT

GETTING MOUSE POSITION

To get the x and y position of the mouse cursor on the screen, we can use the function,

```
pg.mouse.get_pos()
```



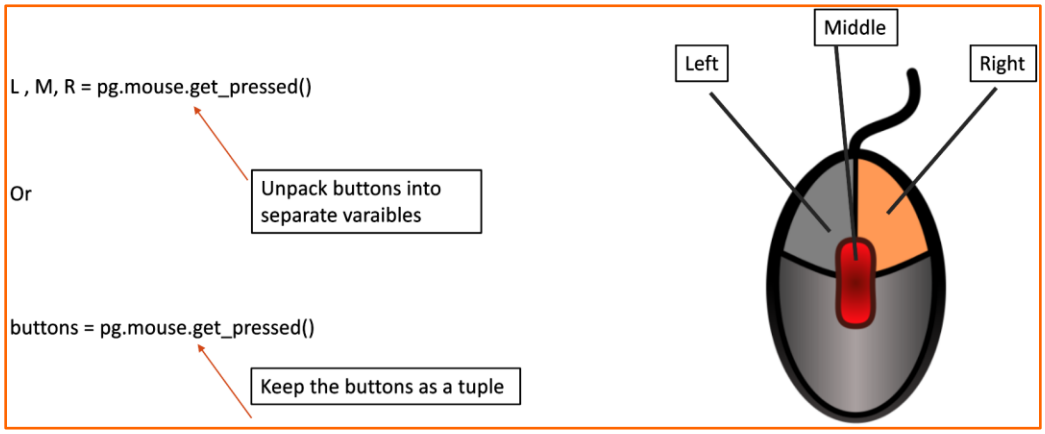
When using the mouse commands, you must use

```
pg.event.pump()
```

GETTING MOUSE BUTTON STATUS

To get the mouse button readings, we can use the function,

```
pg.mouse.get_pressed()
```



© 2021 Exceed Robotics ™

EXAMPLE 2: USING MOUSE POSITION

In this example, we are writing a program that prints the position of the cursor.

Code

```
import pygame as pg
pg.init()
screen = pg.display.set_mode((400,400))
clock = pg.time.Clock()
while True:
    #Inputs
    pg.event.pump()
    mousePos = pg.mouse.get_pos()
    mx = mousePos[0]
    my = mousePos[1]
    #Event
    if mx < 200 and my < 200:
        print("Top left")
    elif mx > 200 and my < 200:
        print("Top right")
    elif mx < 200 and my > 200:
        print("Bottom left")
    else:
        print("Bottom right")
    #Timer
    clock.tick(1)
```

Explanation

The main game loop starts with pg.event.pump(). Next, we read the mouse cursor position using the command pg.mouse.get_pos(). This command outputs a two-item tuple. We separate this tuple into two separate variables: mx and my. Next, we have a If/Elif/else statements, that compares the mx and my values and prints an appropriate message.

© 2021 Exceed Robotics ™

EXAMPLE 3: MOUSE BUTTONS

In this example, we are going to write a program that checks prints text based on which mouse button is pressed.

Code

```
import pygame as pg
pg.init()
screen = pg.display.set_mode((200,200))
clock = pg.time.Clock()

while True:
    #Inputs
    pg.event.pump()
    mouseButton = pg.mouse.get_pressed()
    LMB,MMB,RMB = mouseButton
    #Event
    if LMB == 1:
        print("Duck!")
    if MMB == 1:
        print("Mouse!")
    if RMB == 1:
        print("Lion!")
    #Timer
    clock.tick(10)
```

Explanation

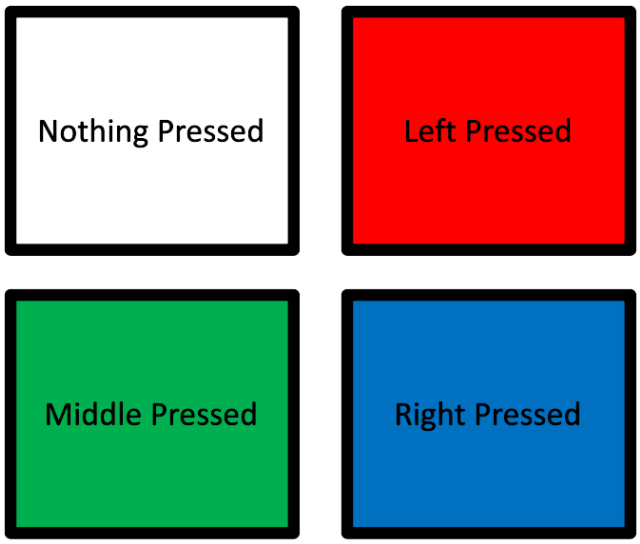
We start the main game loop with a pg. event. pump () command. Next, we read the status of the mouse buttons using the pg.mouse.get_pressed() command. This command outputs a 3-item tuple which is stored inside the variable mouseButton. Next, we unpack the tuple into 3 separate variables. Finally, IF statements are created for each of three scenarios.

© 2021 Exceed Robotics ™

IN CLASS PROBLEMS

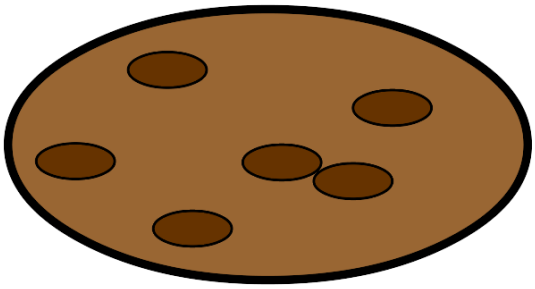
GROUP EXERCISE

Write a program that changes the background colour of the screen depending on what button is pressed.



INDEPENDENT EXERCISE

Create a program that makes the cookie flash green when it's clicked on.



© 2021 Exceed Robotics ™

HOMEWORK

Class 4: Mouse Input

MULTIPLE CHOICE QUESTIONS

1. Which of the following is required for a complete tuple unpacking? (Select all)

   - ☐ There should be the same number of variables, as the number of items in a tuple.
   - ☐ The tuple must be on the right-hand side of the equal sign
   - ☐ The variables must be separated by commas

2. The "**pg.event.pump()**" statement is used to…….

   - ☐ Start the event
   - ☐ Ignore all the background activity
   - ☐ Check the mouse input

3. In game development, FPS stands for

   - ☐ Flashes Per Second
   - ☐ Feet Per Second
   - ☐ Frames Per Second

4. Look at the following code. What is the value stored inside the variable **c**?

```
myTuple = (123, 45, 43)
a, b, c = myTuple
```

   - ☐ 123
   - ☐ 45
   - ☐ 43

5. Look at the following code. What is the value stored inside the variable b?

```
myTuple = (123, 45, 43)
a, b, c = myTuple
b * 10
```

© 2021 Exceed Robotics ™

- ☐ 45
- ☐ Nothing. There is an error
- ☐ 450

6. What does the code below output?

```
A = ('Apple', 'Orange', 'Lemon', 'Strawberry')
count = 0
while count > len(A):
    print(A[count])
    count = count + 1
print('Printing complete')
```

- ☐ Nothing. There is an Error!
- ☐ **Printing complete**
- ☐ **Apple**
  Orange
  Lemon
  Strawberry
  Printing Complete

7. Look at the code below which of the below expressions would return True? [Select All]

```
A = 5
B = 25
C = 22.3
```

- ☐ **(B/A) <= 10**
- ☐ **A > 5**
- ☐ **C-A <= 0**

8. Look at the code below which of the below expressions would return True? [Select All]

```
A = 5
B = 25
C = 22.3
```

- ☐ **(A < 0) or (B == 25)**
- ☐ **(A < 0) and (C > 20)**
- ☐ **(B > C) and (C > A)**

© 2021 Exceed Robotics ™

9. Look at the code below which of the below expressions would return True? [Select All]

```
A, B, C = (25,'Bob',True)
```

- ☐ `(A != 10) or C`
- ☐ `C and (len(B) == 3)`
- ☐ `A > 25 and B = ‘bob’`

TASK-BASED PROBLEMS

TASK A

Open the file "**Week4TaskA-AdamGrades”**.

Adam’s last 10 scores in his math quiz are stored in a tuple. Write a program to calculate his average score. You can calculate the average by first calculating the total score and then dividing that by the number of scores.

TROUBLESHOOTING

PROBLEM A

Open the file "**Week4ProblemA-RandomError”**. Look at the code in the file.

There are 4 errors in the code below can you find them all and fix the code.

© 2021 Exceed Robotics ™

PROBLEM B

Open the file "**Week4ProblemB-WhileError"**. Look at the code in the file. The program should calculate the sum of all the numbers inside the tuple. However, there are 2 errors that you need to find and fix.

Remember: To save the PDF once you are done answering the questions. Use **Ctrl+S** for windows or **Cmd+S** for MacOS

© 2021 Exceed Robotics ™

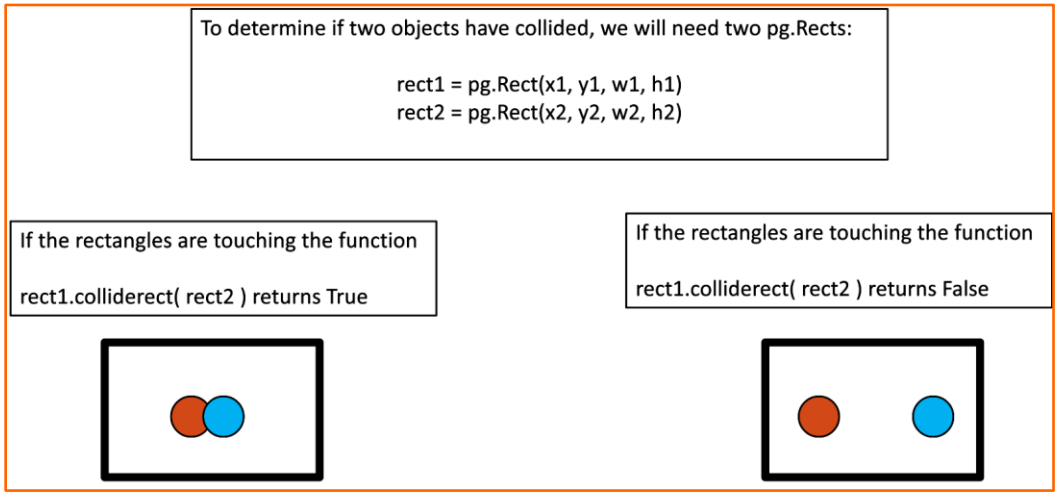
CLASS 5 - 8: COLLISIONS AND PROJECT

COLLISIONS WITH OTHER OBJECTS

CHECKING FOR COLLISIONS WITH ANOTHER OBJECT

Let's recall the **pg. Rect** object. The command to check for collisions between two rectangle objects is,

```
rect1.colliderect(rect2)
```



ADDING IMAGES

You may want to use images from other sources and incorporate them into your game. You may want to use images as either characters or scenery

The next section explains the different steps you need to complete to add and edit images in python.

© 2021 Exceed Robotics ™

IMPORTING IMAGES

To add images into your game, follow the following steps:

1. Find/Download an image. Place the image in the same folder as your python game file.
2. Copy and paste the **expy.py** file into the same folder as your python game file
3. Copy and paste the following lines into your game file.

```
from sys import path
from sys import exit
import os
my_path = os.path.dirname(os.path.realpath(__file__))
os.chdir(my_path)
path.append(my_path)
```

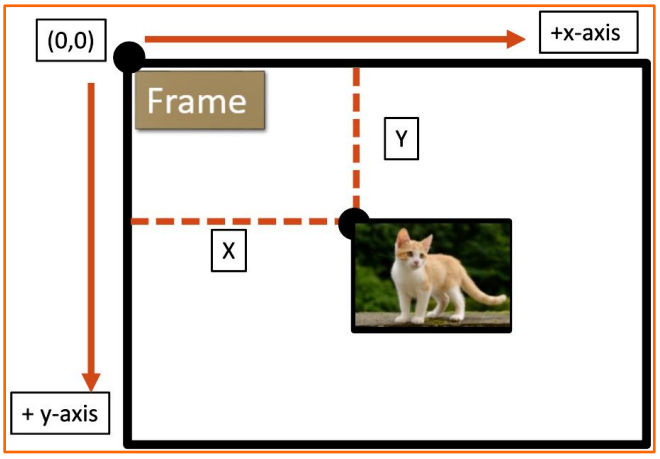
4. Load the image into your pygame using:

```
myImage = pg.image.load( "Filename.ext" )
```

DISPLAYING IMAGES

To display an image that has been imported, we can use the function:

```
screen.blit(imageName,(x,y))
```



© 2021 Exceed Robotics ™

EXAMPLE 1: IMPORTING IMAGES

Code

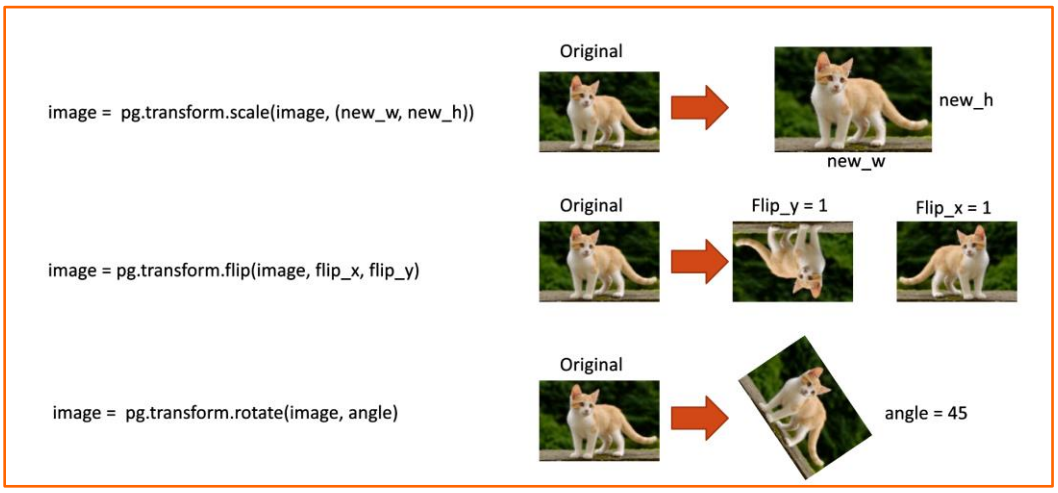
```
catImage = pg.image.load("cat.png")
screen.blit(catImage,(100,50))
pg.display.flip()
```

Description

First, the image file is loaded and stored into a variable called *catImage*. Next, the image stored inside the variable is displayed on the screen. The position of the image is (100px,50px). The frame is updated.

EDITING IMAGES

Generally, editing images is usually done outside of pygame. Microsoft Paint is one option to edit images. However, if you want to edit images within pygame using some image tools. These are,

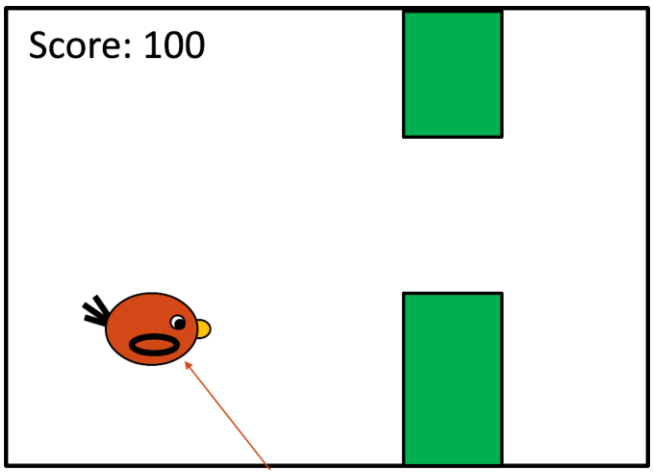


© 2021 Exceed Robotics ™

IN-CLASS PROBLEMS

FLAPPY BIRD PROJECT

Create your version of the flappy-bird game! The project should combine all the skills you have learned this term!



An outline for the game has been provided:

```
while True:
    pg.event.pump()

    #UPDATES
    #move the pipes left
    #move the bird down

    #INPUTS
    #check the mouse buttons

    #EVENTS (if-statements)
    #check if the mouse is pressed
    #check if the pipe went off the screen
    #check if the bird hits the pipe

    #DRAWING
    #clear the screen
    #draw pipes, bird, score
    #update screen

    #CLOCK
    set frame rate
```

© 2021 Exceed Robotics ™

Some of the variables that are required

Variables Needed:

Colours
Speed_bird
Speed_pipe
Score
bird_alive

Objects Needed:

3 pg.Rects (2 pipes, 1 bird)

Some elements of the games are:

1. Make a program that draws the bird and makes it fall when the mouse isn't pressed
2. Add in the pipes, and make them move left until they hit the left side of the screen. The pipe should then respawn on the right side of the screen.
3. Detect collisions between the bird and the pipes, force the game to end when this happens
4. Add in a scoring system.
5. Gradually, increase the difficulty by changing the FPS.

Possible extensions:

1. Random pipe separation
2. Pipe details
3. Background Image/Drawing
4. Game over screen
5. Ability to reset the game
6. Wing flap animation

Anything else you want! Be creative!

© 2021 Exceed Robotics ™

HOMEWORK

CLASS 5-8: PROJECT

MULTIPLE CHOICE QUESTIONS

1. What is the correct statement to check for collisions between 2 rectangle objects **A** and **B**?

☐
```
if A.collideRect(B) == 1:
    #Action
```

☐
```
if A.colliderect(B) == 1:
    #Action
```

☐
```
if A.colliderectangle(B) == 1:
    #Action
```

2. What is the correct statement to check for collisions between a rectangle and a point?

☐ **myRect.collideRect(myPoint)**
☐ **myRect.collidepoint(x,y)**
☐ **myRect.collidePoint(x,y)**

3. The "**pg.event.pump()**" statement is used to _____.

☐ Start the event
☐ Ignore all the background activity
☐ Get mouse input

4. What is the value stored in variable x?

```
A = 12
A = A - 4
A = A + 6
A = A + 1
```

☐ A is 12

© 2021 Exceed Robotics ™

- ☐ A is 15
- ☐ A is 9

Remember: To save the PDF once you are done answering the questions. Use **Ctrl+S** for windows or **Cmd+S** for MacOS

TASK-BASED PROBLEMS

TASK A

Open the file "**Week5TaskA-RandomBouncingBalls"**.

Your task is to create a program that has 3 balls bouncing across the screen. You can either set your own values for the speeds or use random computer-generated values. To generate random integers, you can use the random function. To use the random function, you must do the following.

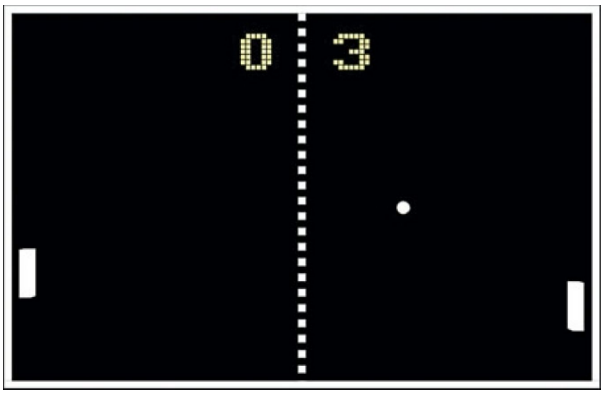
a. Import the random library, by typing **import random** at the top of your code.
b. Then use **random.randint(A, B)** to generate a random integer between A and B

As an extension, check for collisions between the balls. If there is collision make the balls move in the opposite directions

© 2021 Exceed Robotics ™

PONG PROJECT

For the next two classes, we are going to build our version of PONG. The game has 3 main elements. A ball and two paddles. For our game, we want one of the paddles to be controlled by a human and the other one to be controlled by a 'CPU'.



Let's work on the setup of the game first. We need to set up our screen size and initialize a few colours.

TASK 1: INITIALIZING THE GAME: SETTING UP MU

1. Open the file "**PONG Code - Starter Code.py**".
2. Choose some colours for the various elements and assign RGB tuples to the variables under colour
3. Make sure the screen size is set to 660px X 480px
4. Next, we need to add initialize our 3 game elements: the ball, userPaddle and cpuPaddle. Let's also set up some speed variables.

```
# Game objects
userPaddle = pg.Rect(600,20,20,100)
cpuPaddle = pg.Rect(40,20, 20,100)
ball = pg.Rect(320,230,20,20)
paddleSpeed = 5
ballXSpeed = 10
ballYSpeed = 15
```

TASK 2: PROGRAMMING THE CPU PADDLE

The CPU paddle is going to be controlled by a variable called *paddleSpeed*. If the value stored in *paddleSpeed* is positive, the *cpuPaddle* moves down. If the value stored in *paddleSpeed* is negative, the *cpuPaddle* moves up.

© 2021 Exceed Robotics ™

1. In the update section, increase the y position values of the *cpuPaddle* by *paddleSpeed*.
2. In the update section, use IF statements to change the value of paddle speed. If the *cpuPaddle* hits the bottom edge of the screen, the *paddleSpeed* should be negative. Otherwise, it should be positive.
3. In the drawing section, draw the *cpuPaddle* rectangle.

TASK 3: PROGRAMMING THE BALL

The ball position is going to be controlled by ball speed variables.

1. In the update section, increase the x and y position values of the ball by *ballXSpeed* and *ballYSpeed* respectively.
2. In the update section, use IF statements to change the values of the ball speed variables. The variables should change if one of 4 things happen:
    a. The ball hits the top edge of the screen (i.e., ball[1] < 0)
    b. The ball hits the bottom edge of the screen (i.e., ball[1] + ball[3] > 480)
    c. The ball collides with the *userPaddle*
    d. The ball collides with the *cpuPaddle*
3. In the update section, use IF statements to reset the x position of the ball if it crosses the left or the right edge of the screen.
    a. Additionally, you could change the ball speed to a random number.
4. In the drawing section, draw the *ball* as an ellipse.

TASK 4: PROGRAMMING THE USER PADDLE

The user paddle is going to be controlled using the mouse. To do so we need to first find the position of the mouse.

1. In the input section, read and store the x and y mouse position values into mx and my variables respectively. Make sure to use *pg.event.pump()*
2. In the input section, read the x and y mouse cursor position and store them in the variables *mx* and *my* respectively
3. In the update section, change the y position values of the *userPaddle* to *my*.
4. In the update section, use IF statements to ensure that the *userPaddle* does not go out of the screen.
5. In the drawing section, draw the *userPaddle* rectangle.

© 2021 Exceed Robotics ™

TASK 5: ADDING A SCORING SYSTEM AND OTHER AESTHETICS

This section is for you to be creative. You could add a scoring system or any other thing you want.

1. Add a scoring system
2. Add a game title
3. Add images instead for the paddles and the ball.

© 2021 Exceed Robotics ™

Python cheatsheets

Python 1 API

Drawing Fucntions
colourName = (R,G,B)

pg.draw.rect(surface , colour ,rect, w=0)
pg.draw.ellipse(surface , colour ,rect , w=0)
pg.draw.line(surface,colour, (x1,y1), (x2,y2) , w=0)

screen.fill(colour)
pg.display.flip()

Input Functions
pg.event.pump()#ignore all events

mx, my = pg.mouse.get_pos()
L, M, R = pg.mouse.get_pressed()

Clock
clock.tick(rate)

Rectangle
myRect = pg.Rect(x,y,w,h)

myRect.collidepoint(x,y) -> returns a Boolean (1/0)
myRect.colliderect(Rect) -> returns a Boolean (1/0)

Surfaces
pg.Surface((w,h))

Image Functions
pg.image.load('fileName.ext')
screen.blit(surface, (x,y))

myImage = pg.transform.scale(surface, (w, h))
myImage = pg.transform.rotate(surface, angle)
myImage = pg.transform.flip(surface, flip_x, flip_y)

Expy Functions
printText(surface, colour, text, x, y , h=25)
check_exit()

© 2021 Exceed Robotics ™