Assignment#4 More Cool Math in Python

Again, solving problems and creating cool stuff with python sometimes includes some math. Let's take a bit of time to look a more of the handy math functions you will use need to be familiar later on.

Modulus operator % (REMAINDER)

In computer programming the **modulus** operation "**%**" finds the **remainder** of a division operation. It's very common and super important for solving a variety of problems.

Examples:

$5 \% 2 = \mathbf{1}$ ($5 \div 2 = 2$ with a reminder of **1**)

$7 \% 5 = \mathbf{2}$ ($7 \div 5 = 1$ with a remainder of **2**)

$9 \% 3 = 0$

$15 \% 3 = 0$

$17 \% 3 = 2$

Again, the Modulus operator **outputs the remainder** of a division problem.
The Modulus operator comes in very handy in computer programming.
The Modulus operator can help us:

- Distinguish between odd and even numbers.
- Determine if numbers can be divided evenly into other numbers,
- Find the remainder of division problems.
- Do a ton of other cool things!

Exercise #1

The following Python program that will tell the user the remainder of any division problem they wish.
Type in the following into trinket. See what it does and **save** your work.

```
num1 = int(input('Yo Bro! Enter a number that you want to divide:'))
num2 = int(input('Awesome. What would you like to divide your number by?'))
remainder = num1%num2
remainder = str(remainder) # this turns the integer into a string
print ('Sweet Bro! Your remainder of this division will be:' + remainder)
```

Exercise #2

Using the **Modulus Operator** write a Python program that can find out whether a given number is **even** or **odd**, then print out an appropriate message to the user.

Hints:

- **If you do modulus division of any number by 2** and you get an answer of zero then the number is even! (example: 8 % 2 = 0; Therefore 8 is an even number!)
- You will have to use an **`if statement`** for this exercise. See Assignment#1 exercises.
- You will have to compare the answer to zero (for example: `if answer==0:`)
  Notice the double == sign this means: "compare to see if it is equal to"

Sample Input:

Yo Bro…Give me a number and I will tell you if it is even or odd! **45**

Sample output:

45 is an ODD number my friend.

Possible solution below…**don't peek yet see if you can figure it out first.**

Possible solution (no peeking try on your own first please):

```
num = int(input("Give me a number! I will tell you if it is even or odd!: "))
mod = num % 2
if mod == 0:
    print("This is an even number Bro.")
else:
    print("This is an odd number Bro.")
```

Floor division

// - double slash is a special kind of division in computer programming that will give you **only the whole number portion** of the answer to a division problem. Example: 9**/**2=4.5, 9**//**2=4. Predict what the following code will do, type into trinket and include at the end of exercise#2.

```
print(7//2)
print(13//2)
print(8//3)
print(1//4)
print(1//2)
```

Integers vs. Floats (variable types)

Remember, **whole numbers** in computer programming are called: **integers** (int).

Decimal numbers are called: **floats** or floating decimals.

Exercise#3

When you use **floats** it is **important to know few things**:

1. You can use **integers** and **floats** together, but **the result will be a float PLEASE TYPE-IN All the examples to trinket to see what they do**. (save as Exercise#3):

Examples:

```
print(3*3.8134)
print(4.7567 + 56)
print(79/4.23)
```

2. **floats** can displayed so that only a precise number decimal points show:

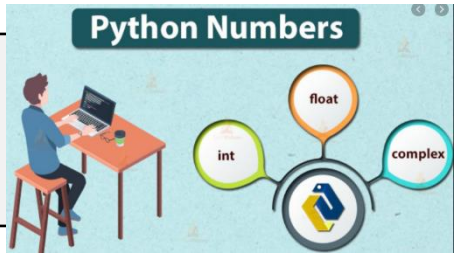
Example:

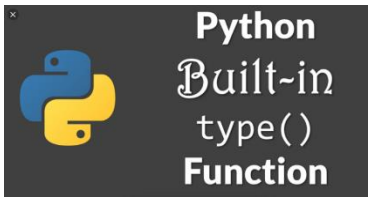
```
a=2.12544566
print("{:.2f}".format(a))  #this will cut-off the decimal after 2 places
b=4.5678972
print("{:.4f}".format(b))  #cut-off after 4 decimal placed NO rounding
```

3. The **round()** function

   Unlike the previous formatting example, the round() function returns a float that is **rounded** to a specified number of decimals.

```
x = round(5.76543, 2)
print(x)
x = round(2.18149, 4)
print(x)
```





The `type()` function

ints , **floats**, **strings**, **characters**, **complex numbers** are all different **data types** in python.

We can use the **type**() **function** to identify which data type a **variable** or a value belongs to.

Exercise#4

Use the **`type()`** function with the examples below and see the output to see what it does.

```
print(type(4.54456))
print(type(4))
print(type("Python is the Best"))
print(type("a"))
```

Random Functions:

This is a **super important** function in programming. Frequently, you will need to input a random values into a program to create a realistic simulation. In programming anytime we model situations that involve chance like:

- rolling of dice,
- coin flipping,
- the shuffling of playing cards,
- motion of an opponent in a video game,

There are a number random functions in python. Here are just a few:

The `randint()` method returns an integer selected from the specified range.

```
import random
x=random.randint(1, 14)
print(x)
```

⇦ Run this code a few times to see it generate random integers and save it with **exercise#4**

The `choice()` method returns a randomly selected element from the specified list of items.

```
import random
mylist = ["apple", "banana", "cherry"]
print(random.choice(mylist))
```

Run this code a few times to see it generate random integers and save it with **exercise#4**

Exercise#5

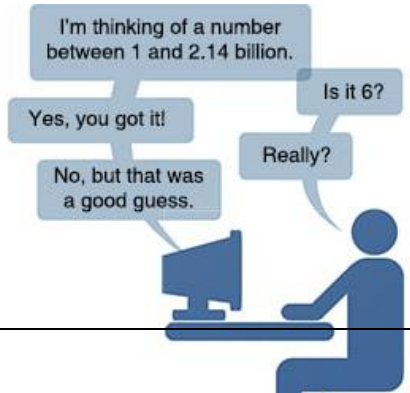
Create a game in python where two players compete in a game of dice. In the game there should be two separate dice values that are added together to get the player total
It might look like the following:

```
Welcome to the dice game! player 1 press the letter r to roll the dice:
Player 1 rolls an 11!
Player 2 press r to roll the dice:
Player 2 roles a 3!
Player 1 is the winner!
```

Exercise#6

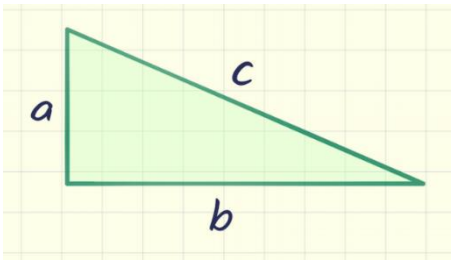
- Create a number guessing game in python where a randomly generated number is created by your program between 1 and 15.
- A player is asked by the computer to guess a number between 1 and 15.
- The computer will let the player know whether the guess is correct, higher, or lower than the generated number and allow the player to guess again if necessary.
- The program will keep track of the number of guesses and will report the number of guesses at the end of the game.
- The game might look like the following:

```
Welcome to the the number guessing game! Guess a number between 1 and 15: 4
Sorry your guess was too low. Guess again: 10
Sorry your guess was too high. Guess again: 7
Excellent! 7 was correct it only took you 3 guesses!
Player 2 roles a 3!
Player 1 is the winner!
```



Exercise#7

Create a python program that when given the two right angled sides of a triangle, it will be able to determine the size of the hypotenuse.
(**Pythagorean Theorem!)**



$$c = \sqrt{a^2 + b^2}$$

Exercise#8

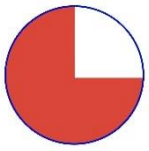
Explore the math module in python and use turtle to do the following:

- draw a right angled triangle that that has 90˚,50˚,40˚ angles

Exercise#9

Explore the math module in python and use turtle to do the following:

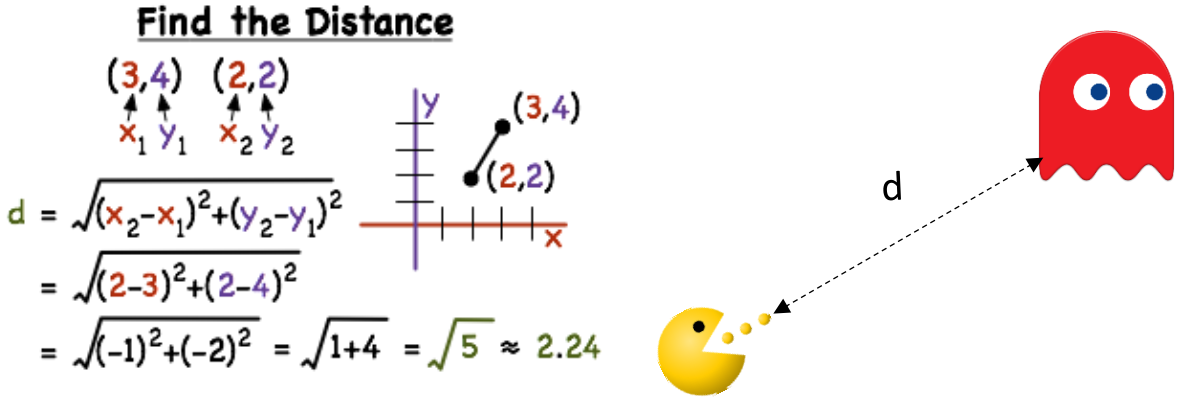
Draw the following two shapes exactly:



Exercise#10

A computer screen is commonly mapped out in software using an **(x,y) coordinate system**. Keeping track of how close to objects on a screen is an essential task in programming.

In mathematics you can find the distance between two points in the following way:



The Formula you need to know is:

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

- Write a Python program that will plot two dots the the screen using turtle.
- then use the formula to compute the distance between the points (x_1, y_1) and (x_2, y_2).
- Then draw a line between the two points and display the distance value you have calculated
- You can check your anwer using the turtle.distance() function. Look it up.
- Partial sample solution on the next page **if** you are stuck.
- You must use: `math.sqrt`

Sample solution on next page. Try first then peek...

Sample solution:

```
import math

p1 = [4, 0]
p2 = [6, 6]
distance = math.sqrt( ((p1[0]-p2[0])**2)+((p1[1]-p2[1])**2) )

print(distance)
```

Exercise#11

Create a python program that can help you solve a **math problem based on what you are doing in math class right now**. Maybe something in your math class that would impress your teacher! Use a turtle drawing/animation to drawing to spice up your program and the user experience.

Super Bonus:

Create a calculator user interface in turtle that operates by the user clicking on your calculator buttons.