Today's Topics:

- Using Math module
- Types of errors

LEE WILKINS

OFFICE : 6C.9

CONTACT: MIO

INTRODUCTION TO COMPUTER PROGRAMMING IN ENGINEERING AND SCIENCE

Comp Sci assignments

Assignment 1: week 4 (10%)

Test 1 week 7 (15%)

Assignment 2 week 8 (10%)

Assignment 3 week 11 (10%)

Test 2 week 13 (15%)

Physics assignments

Assignments (4 x 2%) 8% Date communicated by the Physics teacher

Project 1: Solving differential equations 10% Week 11

Project 2: Applying programming in science 22% Week 15

GRADE BREAKDOWN REVIEW

Next week: non graded practice quiz on Moodle!

GRADE BREAKDOWN REVIEW

Logical: a mistake made in the logic of the program that prevents it from running. Programs with logic errors still run, but potentially might not function as you imagine.

Syntax: a mistake in the writing of the code itself (missing punctuation, brackets, etc)

Run time: an error that occurs as the program runs that causes it to crash. For example, trying to perform mathematical calculations on input strings.

TYPES OF ERRORS

```
if 5+5 == 11:
    print("not true")
else:
    print("this is true")
```

The condition is untrue and will never be true

LOGICAL ERROR

```
if 5+5 == 11
    print("not true")
else
    print("this is true")
```

The program won’t compile because there are syntax elements missing

SYNTAX ERROR

```
myVar = int(input("input a number “))
print(myVar + 4)
```

The program runs, but if the user inputs something that isn’t a number there will be an error casting it to an integer

RUN TIME ERROR

Addition: + myVariable = 10+5

>> 15

Subtraction: - myVariable = 10-5

>> 5

Multiply: * myVariable = 10*5

>> 50

Division: / myVariable = 10/5

>> 2

Modulus: % Modulus is the remainder of division myVariable = 10%5

>> 0

Exponent: ** myVariable = 10**5

>> 100000

ARITHMETIC OPERATORS

```
# Each bag of fruit has 4 apples and 2 oranges.
# I want to make 5 bags. How many fruit do I need in total?
myVar = 2 + 4 * 5
myOtherVar = (2+4) * 5

print(f'I have {myOtherVar} pieces of fruit not {myVar}')
```

REVIEW: PEDMAS

```
import math
```

This gives access to all kinds of things to help you do more advanced calculations. You can see all of them here.

You need to put this at the top of your document or above all of the math module methods you use.

https://www.w3schools.com/python/module_math.asp

MATH MODULE

```
print(math.pi) # pi
print(math.e) # Euler's number
print(math.tau) # tau
```

By importing math we have access to a number of different methods and constants. These are constants that don't have () because they are not performing tasks, they are representing numbers.

CONSTANTS

```
my_num = math.sqrt(4)
print(my_num)
```

You can pass numbers through a method as a parameter (or param). In this example, we are passing 4 through the square root function (math.sqrt())

You can alway look up how it works

https://www.w3schools.com/python/ref_math_sqrt.asp

METHODS

You can convert degrees and radians by passing them THROUGH the appropriate methods. You can pas a method a parameter by putting it in the round brackets like this:

```
print(math.degrees(1))
# this takes radians, prints degrees
print(math.radians(90))
# this takes degrees, prints radians
```

You can also pass variables through a function

```
my_angle_degrees = 90
my_angle_radians = math.radians(my_angle_degrees)
print(f'{my_angle_degrees} degrees is {my_angle_radians} radians’)

another_angle_radians = 1.1
another_angle_degrees = math.degrees(another_angle_radians)
print(f'{another_angle_degrees} degrees is {another_angle_radians} radians')
```

USING MATH.

```
my_num = math.pow(2,2)
print(my_num)
```

Some methods need multiple parameters, such as pow. This needs to know the base and the exponent

https://www.w3schools.com/python/ref_math_pow.asp

USING MATH.

You can pass methods through other methods. Like this:

```
myNum = 2.0

print(math.sqrt(math.pow(myNum, 2)))
> 2.0

print(math.log(math.exp(myNum)))
> 2.0

print(math.log10(math.pow(10, myNum)))
> 2.0
```

USING MATH.

```
opposite_side = 5
adjacent_side = 3

These two lines are the same
hypotenuse_side = opposite_side**2 + adjacent_side**2
This is the same line using math.pow
hypotenuse_side = math.pow(opposite_side, 2) + math.pow(adjacent_side, 2)


I can apply the square root function after
hypotenuse_side = math.sqrt(hypotenuse_side)


You can also do it all in one expression like this
hypotenuse_side = math.sqrt(math.pow(opposite_side, 2) + math.pow(adjacent_side,
2))
print(f'The triangle hypotenuse is {hypotenuse_side}’)

Remember, it is not faster to do all the calculations in one line. Don’t hesitate
to spread it out and use multiple lines. Make it easy to read!
```

USING MATH.

Calculate the adjacent angle with the two sides, your result is in radians.

```
angle_adjacent = math.atan(opposite_side/adjacent_side) Returns the value in radians
```

Convert it to degrees like this

```
angle_adjacent = math.degrees(angle_adjacent)
```

Or all in one expression where degrees holds the entire expression

```
angle_adjacent = math.degrees(math.atan(opposite_side/adjacent_side))
```

USING MATH.

LAB TIME

WRITING YOUR OWN FUNCTIONS

NEXT WEEK:

IN CLASS ASSIGNMENT, GRADED THROUGH MOODLE
BUT YOU CAN USE YOUR CODE EDITOR TO WRITE

NEXT CLASS: