Lab 2 Manual

Lab 2 Outline

Part 1: Experimenting with Python [35 mins]

- celsius_to_fahrenheit.py
- division.py
- circle_property.py
- hello_returning.py
- hello_printing.py
- scholarship_determination.py

Part 2: Quiz [15 mins]

Now that you have completed the lab, you will take a short mandatory quiz on the lab material. To take the quiz, **your lab leader will provide you with the link and passcode to access the Lab Quiz on your Ed account**. This will be a **timed (15 mins)**, multiple choice quiz on the lab material. If you completed the lab, you will be able to answer all of the questions on the quiz.

You are required to take the quiz during your synchronous lab session. Make sure to leave enough time to take the quiz before the end of the lab session.

celsius_to_fahrenheit.py

Create a file called celsius_to_fahrenheit.py, and copy the function definition in the code block below and paste to the file.

```
def celsius_to_fahrenheit(degree_in_celsius):
    return degree_in_celsius*9/5+32
```

This function takes in the degree in Celsius, as a number(int or float), and returns the degree in Fahrenheit as a number.

In the terminal, run `python -i celsius_to_fahrenheit.py`. Do you see any error messages? Fix any errors or typos in your program before moving on! Ask your labmates if you're stuck.

Then at the Python prompt, run `celsius_to_fahrenheit(100)`. This is a function call that is calling the `celsius_to_fahrenheit` function with an argument, the value 100. What do you see as the result? Is that what you expected?

Write some more function calls of the `celsius_to_fahrenheit` function with different arguments and see if the results match your expection.

Recap: You have now ***written a function definition***, and then ***loaded it into interactive Python***, and then ***called it with arguments*** to see the values it produced.

division.py

Create a file called `division.py` and copy and paste the function definition from the code block below into the file.

```
def division(dividend,divisor):
    return dividend/divisor
```

Write a function call of the `division` function with different arguments and swap the first and second number that you passed in(e.g. `division(1,2)` and `division(2,1)`). Do you see different result? What does this tell you about the order in which the arguments are passed in?

circle_property.py

Create a file called `circle_property.py` and copy and paste the function definition from the code block below into the file.

```
def circle_property(radius):
    area = 3.14*(radius**2)
    circumference = 2*3.14*radius
    return circumference,area
```

Write a function call of the `circle_property`. In Python, functions can return multiple values. How many values are being returned by this function?

hello_returning.py

Create a file called `hello_returning.py` and copy and paste the function definition from the code block below into the file.

```
def hello_returning():
    return "hello"
```

Write a function call of `hello_returning`. In the previous functions we have done in this lab, you can call a function with different parameters, is there another way you can call this `hello_returning` function?

hello_printing.py

Create a file called `hello_printing.py` and copy and paste the function definition from the code block below into the file.

```
def hello_printing():
    print("hello")
```

Write a function call to the `hello_printing` function under the function definition within the `hello_printing.py` file, so now the code looks like this:

```
def hello_printing():
    print("hello")

hello_printing()
```

Is there any difference between the result of function call of `hello_printing` and the function call of `hello_returning`?

scholarship_determination.py

Create a file called `scholarship_determination.py` and copy and paste the function definition from the code block below into the file.

```
def scholarship_determination(gpa, on_academic_probation, in_state, major):
    if on_academic_probation:
        return 0
    scholarship_amount = 0
    if in_state:
        scholarship_amount = scholarship_amount + 500
    if gpa >= 3.75:
        scholarship_amount = scholarship_amount + 1000
    elif gpa >= 3.5:
        scholarship_amount = scholarship_amount + 500
    else:
        scholarship_amount = scholarship_amount + 100
    if major == "CSE":
        scholarship_amount = scholarship_amount + 100
    if major == "MATH":
        scholarship_amount = scholarship_amount + 50
    return scholarship_amount
```

This function takes in a student's GPA, whether they are on academic probation or not, whether they are in state or not, and their major, and returns the total scholarship amount.

What do you think the scholarship would be for a student with a 4.0 GPA, not on academic probation, in state, and a CSE major. Try running the function with these arguments, `scholarship_determination(4.0, False, True, "CSE")`. What do you see as the result? Is that what's expected?

Write some more function calls of `scholarship_determination` function with different arguments and see if the results match your expectation.