Computer Science 1 (../index.html) CPSC 1301K (../../index.html)

Python Intro Practice Assignments

Next (../modules/modulesPracticeAssignments.html)

Overview

Submission Instructions

For each of the following practice assignments, save your solution in a `.py` file, with the name being `intro` and the number of the assignment. For example, for "Intro01: Go Cougars!", save your solution in a file named `intro01.py` (notice the lowercase "i"). Then, submit that file the respective assignment on codePost.io (https://codePost.io).
To register for a free account, go to https://codepost.io/signup/join?code=I5039NQNWJ (https://codepost.io/signup/join?code=I5039NQNWJ). Register with your CSU email address. Sometimes it takes more than one try, so please work on this well before the deadline. Additionally, some students have had better luck with using the “Forgot password” link.

Practice Assignments

Intro01: Go Cougars!

Create a file named `intro01.py` and write Python code that displays:

```
Go Cougars!
```

Upload `intro01.py` to codepost.io (codepost.io) for the "Intro01: Go Cougars!" assignment.

Intro02: Quotes

Write Python code so that the following text is displayed (exactly as shown):

```
Cody the Cougar yelled, "Let's go!"
```

Intro03: CSU Fight Song

Using a single call to the print`()` function, display the beginning of the CSU fight song (on 4 lines):

```
On! On! Ye Cougars!
We will fight for victory!
On! On! Ye Cougars!
We have the pride for all to see!
```

Note: Do not include a space after any of the "`!`"s. A space is a character.

Intro04: Assigning a Variable

Replace the (red) string literal '''Put your code here''' with Python code that creates a variable named **`letterGrade`**. Have **`letterGrade`** reference the value `"A+"`.
Note: Submit your code with the print(letterGrade) statement.

Provided starter code (`intro04.py`):

```
'''Put your code here'''
print(letterGrade)
```

Intro05: Number of Classes

Replace the (red) string literal '''Put your code here''' with Python code that creates a variable named **`numberOfClasses`**. Assign the variable **`numberOfClasses`** with the number of classes that you are taking this semester.

Provided starter code (`intro05.py`):

```
'''Put your code here'''
print(numberOfClasses)
```

Intro06: 1958 (int)

Replace the (red) string literal '''Put your code here''' with Python code that creates a variable named **`year`**. Assign `1958` to the variable **`year`** so that your code displays the following:

```
<class 'int'>
```

Provided code:

```
'''Put your code here'''
print( type( year ) )
```

Intro07: 1958 (str)

Replace the (red) string literal '''Put your code here''' with Python code that creates a variable named **`year`**. Assign `1958` to the variable **`year`** so that your code displays the following:

```
<class 'str'>
```

Provided code:

```
'''Put your code here'''
print( type( year ) )
```

Intro08: Pi Approximation

The fraction 22/7 is sometimes used as a rough approximation for Pi (3.14159...). Write a Python expression that divides 22 by 7 and assigns the result to a variable named **`piApproximate`**.

Provided code:

```
'''Put your code here'''
print( piApproximate )
```

Intro09: Friends and Pizza Slices

Assume that you have 22 slices of pizza and 7 friends that are going to share it (you've already eaten). There's been some arguments among your friends, so you've decided to only give people whole slices. Write a Python expression with the values 22 and 7 that calculates the number of whole slices each person would receive and assigns the result to **`numberOfWholeSlices`**.

Provided code:

```
'''Put your code here'''
print( numberOfWholeSlices )
```

Intro10: Pizza for Fido

Assume that you have 22 slices of pizza and 7 people that are going to share it. There's been some arguments among your friends, so you've decided to only give people whole slices. Your pet dog Fido loves pizza. Write a Python expression with the remainder operator **that calculates how many pizza slices will be left over for your dog after serving just whole slices to 7 people. Assign the result of that expression to `fidos`.**

Provided code:

```
'''Put your code here'''
print( fidos )
```

Intro11: Hello

Python allows you to get information from a user with the built-in function named `input()`. In codepost.io, any input values are already entered, it's just waiting for a Python script to request it.
For this problem, write a Python script that prompts the user (in this case, the codepost.io system) with exactly the following:

```
Please enter your name:
```

codepost.io will then enter a name. Assign that value to **`username`**.
Example 1:

```
Please enter your name: Pat
Hello Pat
```

Example 2:

```
Please enter your name: codepost
Hello codepost
```

Provided code:

```
'''Put your prompt here'''
print( 'Hello', username)
```

Intro12: Years as Columbus College

For this problem, we're going to have python calculate the number of years that CSU was known as Columbus College. First, prompt the user to enter the year that Columbus College was renamed to be Columbus State University and assign that to a variable named **`csuYear`**. Then, prompt the user to enter the year that Columbus College was founded and assign that value to **`ccBirthYear`**.
Note, if after submitting your solution it says something like:

```
TypeError: unsupported operand type(s) for -: 'str' and 'str'
```

then that means that the Python interpreter does not know how to *subtract* one string from another. Hint: It needs those variables to be `int`s and not `str`s.
Note, if after submitting your solution the solutions says:

```
.+You entered 1996
.+You entered 1958
```

This means that you are missing a prompt. Add one and re-submit. By the way, "`.+`" is a regular expression. It means it was expecting one or more characters.

Provided code:

```
'''Prompt the user and get the year Columbus College was renamed to CSU'''
print('You entered', csuYear)

'''Prompt the user and get the year Columbus College was founded from the user'''
print('You entered', ccBirthYear)

print( 'CSU was known as Columbus College for its first', csuYear - ccBirthYear, 'years')
```

Intro13: Lowest to Highest Precedence

In math and Python, the following three statements are equivalent:

```
3 * 4 + 5
```

```
(3 * 4) + 5
```

```
((3 * 4) + 5)
```

The set of parentheses only makes the precedence order explicit.
For this problem, add sets of parentheses so that the Python expression is evaluated with the lowest order of precedence first. (See https://runestone.academy/runestone/static/thinkcspy/Appendices/PrecedenceTable.html (https://runestone.academy/runestone/static/thinkcspy/Appendices/PrecedenceTable.html) for a Python operator precedence table.) Hint, for the first expression, the operator with the lowest precedence is subtraction, so the

first set of parentheses to add would be around `9 - 2`.

Provided code:

```
'''Add parentheses so that the following expression
is evaluated from the lowest to the highest
precedence order'''

result1 = 8 / 2 ** 9 - 2
print( 'result1:', result1 )

result2 = 1 - 2 * 3 + 4 / 5
print( 'result2:', result2 )

result3 = 1 * 2 - 3 / 4
print( 'result3:', result3 )

result4 = 66 // 7 ** 8 % 9 + 10)
print( 'result4:', result4 )
```

Intro14: Making Change

Implement a Python program that directs a cashier how to give change. The program has two inputs: the amount due and the amount received from the customer. Display the dollars, quarters, dimes, nickels, and pennies that the customer should receive in return. In order to avoid round-off errors, the user should supply both amounts in pennies. For example, for the amount $5 and 26 cents, the user will enter 526 (instead of 5.26).
For this assignment you may need find it helpful to use `//` (the integer division operator) and `%` (the remainder operator).
We usually write computer scripts to work for all valid inputs. Your script should work for any valid input, including the examples below:

Example 1:

```
Enter the amount due in pennies: 828
Enter the amount received from the customer in pennies: 1000
Give the following change to the customer:
1 dollars, 2 quarters, 2 dimes, 0 nickels and 2 pennies
```

Example 2:

```
Enter the amount due in pennies: 456
Enter the amount received from the customer in pennies: 2000
Give the following change to the customer:
15 dollars, 1 quarters, 1 dimes, 1 nickels and 4 pennies
```

Example 3:

```
Enter the amount due in pennies: 401
Enter the amount received from the customer in pennies: 500
Give the following change to the customer:
0 dollars, 3 quarters, 2 dimes, 0 nickels and 4 pennies
```

Hint

Stuck on where to start? Perform the calculations "by-hand" or "on paper". This will allow you to work through the math and see the pattern that will work for all valid cases. Then, code part of it and put in temporary `print()` statements to see what's going on. For example, display the amount of change due after getting both inputs.
Still stuck? If the change is 76 cents, how do you know how many quarters are needed? You would divide the change by the amount of a quarter.

Provided code:

```
# Retrieve inputs

# Calculate dollars to return
# Calculate quarters to return
# Calculate dimes to return
# Calculate nickels to return
# Calculate pennies to return

# Display change due
print("Give the following change to the customer:")
print( dollars, "dollars,", quarters, "quarters,", dimes, "dimes,", nickels, "nickels and", pen
nies, "pennies")
```