**COLLEGE OF COMPUTER STUDIES AND MULTIMEDIA ARTS**

## 

**IT0011**

**(INTEGRATIVE PROGRAMMING TECHNOLOGIES)**

**EXERCISE**

**1**

**RUNNING STATEMENTS**

|  |  |
| --- | --- |
| **Student Name / Group Name:** | Carl Allen L. Torno |
| **Members (if Group):** | |  |  | | --- | --- | | **Name** | **Role** | |  |  | |  |  | |  |  | |
| **Section:** | TW21 |
| **Professor:** | Dr. Jay-ar Lalata |

1. **PROGRAM OUTCOME/S (PO) ADDRESSED BY THE LABORATORY EXERCISE**

* Analyze a problem and identify and define the computing requirements appropriate to its solution. [PO: B]

1. **COURSE LEARNING OUTCOME/S (CLO)ADDRESSED BY THE LABORATORY EXERCISE**

* Understand the Python syntax and semantics. To run program in Python and understand some basic program error. [CLO: 1,2]

1. **INTENDED LEARNING OUTCOME/S (ILO) OF THE LABORATORY EXERCISE**

At the end of this exercise, students must be able to:

* Apply the kinds of formatting outputs
* Create a Python program that accepts input and produces outputs.
* Utilize operators and built-in functions in solving a computing problem.

1. **BACKGROUND INFORMATION**
2. **Steps to Create and Run a Python Program in VS Code:**
3. **Installation and Setup:**

* Ensure you have Python installed on your machine. If not, download and install it from the official Python website.
* Install Visual Studio Code from the official website.
* Launch VS Code.

1. **Setting up Python Extension:**

* Go to the Extensions view by clicking on the square icon on the sidebar or pressing Ctrl + Shift + X.
* Search for the "Python" extension by Microsoft and install it.
* Once installed, reload VS Code.

1. **Creating a New Python File:**

* Click on the Explorer icon (folder icon) on the sidebar.
* Create a new folder or select an existing one.
* Right-click inside the folder, select New File, and name it with a .py extension, e.g., hello.py.

1. **Writing Python Code:**

* Open the newly created .py file.
* Write your Python code in the editor.

For example:

print("Hello, World!")

1. **Running the Python Program:**

* Right-click anywhere in the editor and choose Run Python File in Terminal. This action will open the integrated terminal at the bottom of VS Code and execute the Python program, displaying the output.

1. **Using the Integrated Terminal:**

* Alternatively, you can use the integrated terminal directly:
  + Open the integrated terminal in VS Code by selecting View > Terminal or pressing Ctrl + ~.
  + Navigate to the directory containing your Python file using the cd command.
  + Run the Python script using the command: python hello.py (assuming the script's name is hello.py).

1. **Operators in Python:**
2. Arithmetic Operators: These include + (addition), - (subtraction), \* (multiplication), / (division), % (modulus), \*\* (exponentiation), and // (floor division).
3. Comparison Operators: These operators compare two values and return either True or False. Examples include == (equal to), != (not equal to), < (less than), > (greater than), <= (less than or equal to), and >= (greater than or equal to).
4. Logical Operators: These operators are used to combine conditional statements. They include and, or, and not.
5. Assignment Operators: Operators like =, +=, -=, \*=, /=, %= etc., are used to assign values to variables.
6. Identity Operators: is and is not are used to compare objects, checking if they are the same object in memory.
7. Membership Operators: in and not in are used to test if a value or variable is found in a sequence (like lists, tuples, strings, etc.).
8. **Formatting Outputs in Python:**
9. String Formatting with % Operator: Allows formatting using the % operator.

For example:

print("Hello, %s!" % "Python")

1. Formatted String Literals (f-strings): Introduced in Python 3.6, f-strings provide a concise and convenient way to embed expressions inside string literals:

For example:

name = "Python"

print(f"Hello, {name}!")

1. String format() Method: Another way to format strings:

For example:

* print("Hello, {}!".format("Python"))
* print("Hello, {0}! My name is {1}.".format("Python", "Jay-ar"))

1. **LABORATORY ACTIVITY**

**Instructions:**

Read and analyze the problems then create a Python program that satisfies the requirement of each problem.

**ACTIVITY 1.1: Formatting Output**

Write a Python program that accepts two numbers and computes their sum, difference, product, and quotient. The output should be displayed using the three ways to format outputs.

**Expected Output:**

The sum of 10 and 5 is 15.

The difference of 10 and 5 is 5.

The product of 10 and 5 is 50.

The quotient of 10 and 5 is 2.

***Copy the source code directly from the IDE so that colors of the codes are preserved for readability. Snip the output as seen in the sample and paste it below. Use one set of inputs for all kinds of formatting outputs. Screenshot the output and paste it under the Output label.***

1. Using modulo operator

**Source Code:**

*# Formatting Output using modulo operator*

*# Getting user input*

num1 = int(input("Enter num 1: "))

num2 = int(input("Enter num 2: "))

*#solving for each operation*

sum = num1 + num2

diff = num1 - num2

qou = num1 // num2

pro = num1 \* num2

*# Displaying output for each operation*

print("\nThe sum of %d and %d is %d." % (num1, num2, sum))

print("The difference of %d and %d is %d." % (num1, num2, diff))

print("The product of %d and %d is %d." % (num1, num2, pro))

print("The qoutient of %d and %d is %d." % (num1, num2, qou))

**Output:**

**A computer screen with blue text

Description automatically generated**

1. Using f or F

**Source Code:**

*# Formatting Output using F*

*# Getting user input*

num1 = int(input("Enter num 1: "))

num2 = int(input("Enter num 2: "))

*#solving for each operation*

sum = num1 + num2

diff = num1 - num2

qou = num1 // num2

pro = num1 \* num2

*# Displaying output for each operation*

print(f"\nThe sum of {num1} and {num2} is {sum}.")

print(f"The difference of {num1} and {num2} is {diff}.")

print(f"The product of {num1} and {num2} is {pro}.")

print(f"The qoutient of {num1} and {num2} is {qou}.")

**Output:**

**A computer screen with blue text

Description automatically generated**

1. Using format method

**Source Code:**

*# Formatting Output using String format (Empty placeholder)*

*# Getting user input*

num1 = int(input("Enter num 1: "))

num2 = int(input("Enter num 2: "))

*#solving for each operation*

sum = num1 + num2

diff = num1 - num2

qou = num1 // num2

pro = num1 \* num2

*# Displaying output for each operation*

print("\nThe sum of {} and {} is {}.".format(num1,num2,sum))

print("The difference of {} and {} is {}.".format(num1,num2,diff))

print("The product of {} and {} is {}.".format(num1,num2,pro))

print("The qoutient of {} and {} is {}.".format(num1,num2,qou))

**Output:**

**A computer screen with blue text

Description automatically generated**

**ACTIVITY 1.2: There’s a Heron**

Write a Python program that computes the area of a triangle using Heron's formula. Data type should be float. Note: You are not allowed to user-defined functions.

A math equation with black text

Description automatically generated

**Sample Output:**

A black text on a white background

Description automatically generated

***Copy the source code directly from the IDE so that colors of the codes are preserved for readability. Snip the output as seen in the sample and paste it below.***

**Source Code:**

import math

*# Solving for Heron*

*# Getting user input*

sideA = float(input("Enter the length side a: "))

sideB = float(input("Enter the length side b: "))

sideC = float(input("Enter the length side c: "))

*# Solving for semi-perimeter*

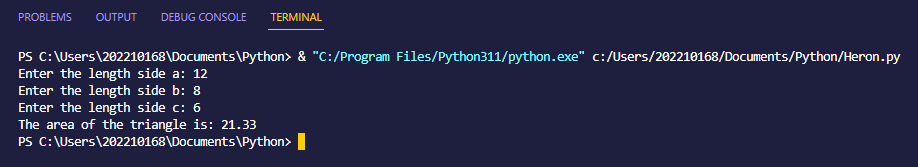
semi = (sideA + sideB + sideC) / 2.0

*# Soling for the Area*

area = math.sqrt(semi \* (semi-sideA) \* (semi-sideB) \* (semi-sideC))

print("The area of the triangle is: %.2f" % area)

**Output: (Use three sets of inputs and screenshot all the outputs and paste them here)**

****

**A blue and white text on a dark background

Description automatically generated**

**A blue screen with white text

Description automatically generated**

**ACTIVITY 1.3: Read between the Lines**

Write a Python program that accepts two points then computes the slope of the line using the two points. The program should output the equation of the line. Data type should be float. Note: You are not allowed to user-defined functions.

**Sample Output:**

A black text on a white background

Description automatically generated

***Copy the source code directly from the IDE so that colors of the codes are preserved for readability. Snip the output as seen in the sample and paste it below.***

**Source Code:**

*# Read between the lines*

*# Getting user input*

x1 = float(input("Enter x1: "))

y1 = float(input("Enter y1: "))

x2 = float(input("Enter x2: "))

y2 = float(input("Enter y2: "))

*# Solving the slope*

slope = (y2-y1) / (x2-x1)

intercept = y1 - slope \* x1

if(intercept < 0):

    print("The equation of the line is: y = %.2fx %.2f" % (slope,intercept))

else:

    print("The equation of the line is: y = %.2fx + %.2f" % (slope,intercept))

**Output: (Use three sets of inputs and screenshot all the outputs and paste them here)**

**A blue screen with white text

Description automatically generated**

**A computer screen with blue text

Description automatically generated**

**A computer screen with blue and white text

Description automatically generated**

1. **QUESTION AND ANSWER**

Briefly answer the questions below.

|  |
| --- |
| * + What is the advantage of using Python Programming? * It is easy to understand because it forms an English like structure also you can compare it to a pseudocode. It has a shorter syntax code. |
| * + What is Python interpreter? * As we all know an interpreter compared to compiler is that in interprets the code line by line inside of as a whole. Also, A python interpreter is a computer program that converts each high-level program statement into machine code. |
| * + What is importance of having different ways in formatting outputs? * We can use the 3 methods the modulo, using f, and the format method using the can easily combine the printing instead of separating the strings or phrase to a variable by giving it a placeholder, also you can modify such as decimal places or number of characters to be displayed. |

1. **GRADING SYSTEM/ RUBRIC**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Trait** | **(Excellent)** | **(Good)** | **(Fair)** | **(Poor)** |
| **Correct Identification of Inputs and Outputs(30pts)** | Able to identify correctly all input and output and provide alternative.  **(30-26pts)** | Able to identify correctly all input and output  **(25-17pts)** | Able to identify only one input or output  **(16-9pts)** | Unable to identify any input and output  **(8-1pts)** |
| **Requirement Specification(40pts)** | The program works and meets all specifications. Does exception al checking for errors and out-of- range data **(40-31pts)** | The program works and meets all specifications. Does some checking for errors and out of range data **(30-21pts)** | The program produces correct results but does not display correctly Does not check for errors and out of range data **(20-11pts)** | The program produce s incorrect results **(10-1pts)** |
| **Free from syntax, logic, and runtime errors (30pts)** | Unable to run program **(30-26pts)** | Able to run program but have logic error **(25-17pts)** | Able to run program correctly without any logic error and display inappropriate output **(16-9pts)** | Able to run program correctly without any logic error and display appropriate output **(8-1pts)** |

1. **REFERENCES**

* Muller, A. C. & Guido, S. (2017). Introduction to Machine Learning with Python. O’Reilly Media, Inc
* Eric Matthes (2016). Python Crash Course. No Starch Press, Inc.
* http://onefeu.instructure.com