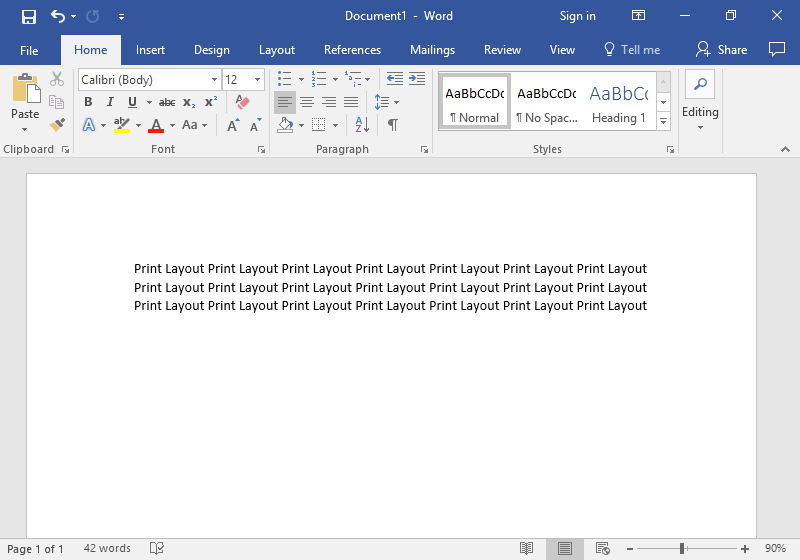
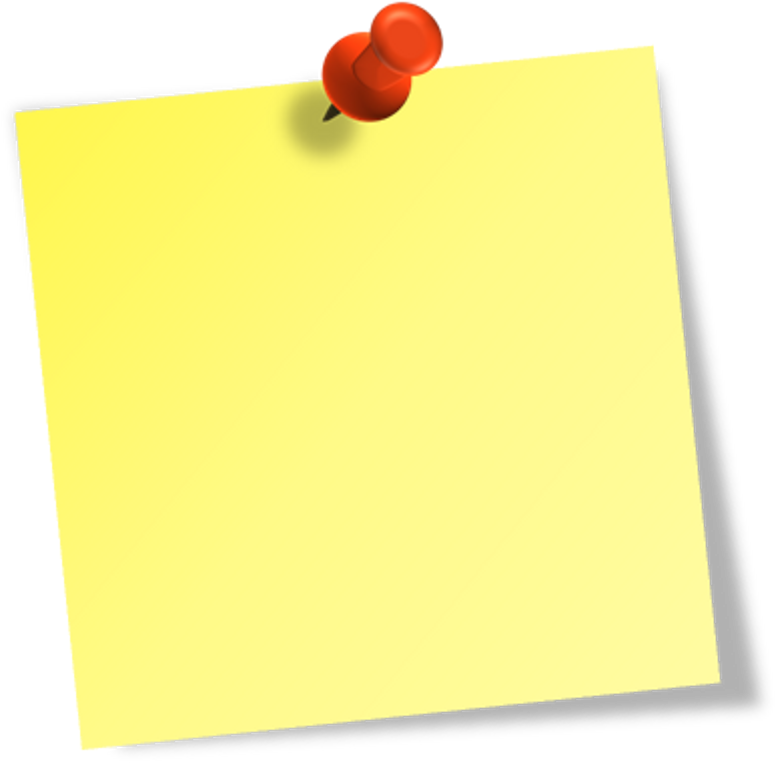
**FOURTH GRADING LEARNING MODULE IN COMPUTER 10**

**PYTHON BASICS**

****



**What to Learn in this Module**

**Lesson 1: Python Flow Control**

1. **Assignment Operator**
2. **Logical Operator**
3. **While Loop**
4. **For Loop**

**Lesson 2: Debugging**

1. **Debugging**
2. **Syntax Error**
3. **Logical Error**

**Programming itself is a work of art: it’s the product of a creative mind. Programming is science: it’s innovation. Programming is math: math is the basic of programming. Programming is a language that has its own grammar: programming needs to be syntactically and logically correct.**

**Programming help builds the society for what it is today. Imagine not having gadgets and applications. Our online class would not be possible without programming.**

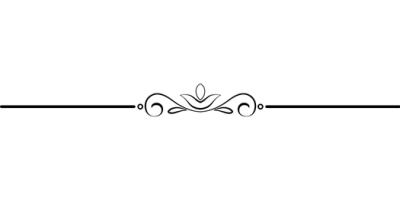
**This module will help you discover more than just the concept of programming.**

**So dear students, keep on coding!**

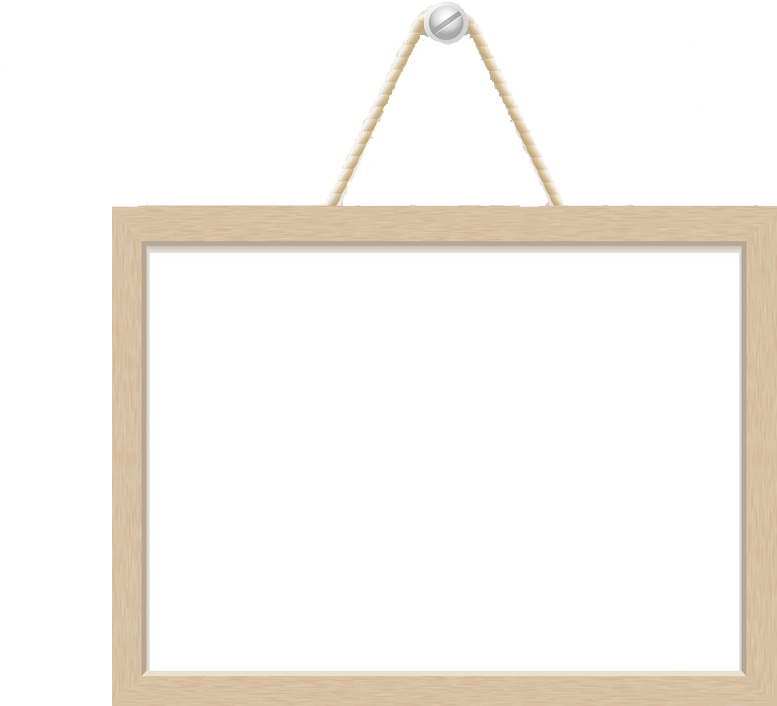


**CONFIDENTIALITY NOTICE:** This module and any of its attachments are meant for the use of the learners who are enrolled in this subject. This module is for academic/ school purposes only and is not meant for sharing, publishing, copying, and disseminating. Hence,

the distribution of this module and any of its content/s is strictly prohibited.

**Lesson 1: Python Flow Control**





* Learning Objectives

**Checklist of Activities**

* Word Sight
* Acronym

After completing this module, you should be able to:

* Understand and explain underlying concept of programming;
* Understand and explain flow control of Python;
* Create block of codes free from errors;
* Debug codes;
* Create simple program using Python; and
* Display confidence and commitment in accomplishing all given tasks

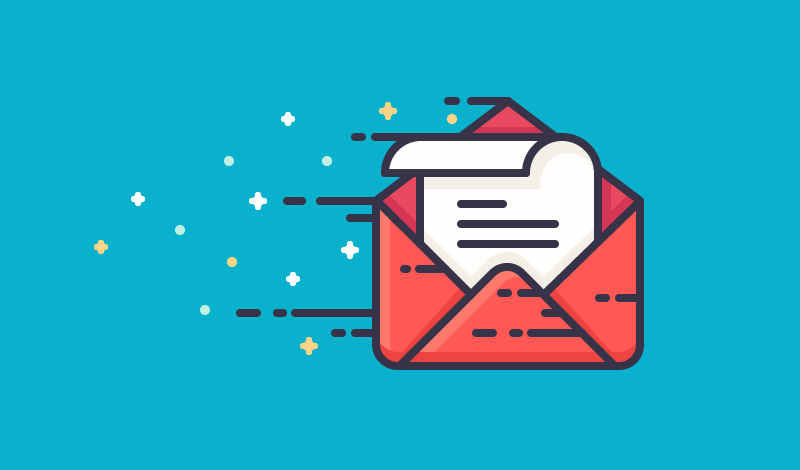


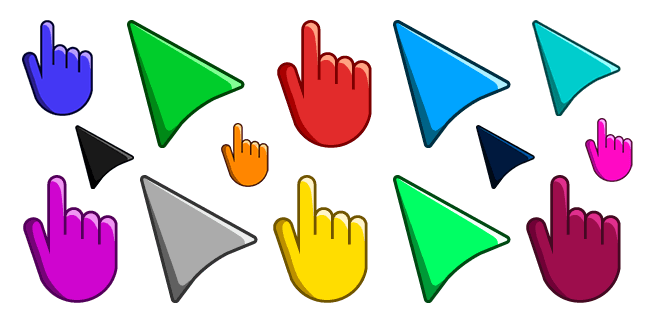


* Time Frame

Here is the third grading time frame to help you manage your time. Give yourself some rest in between. You can do it!

|  |  |  |
| --- | --- | --- |
| Activities | Completion Time | Date |
| Lesson: Let’s Recall | 40 Minutes | ----------------- |
| Activity: Word Sight | 15 Minutes | ----------------- |
| Lesson: Assignment Operator | 30 Minutes | ----------------- |
| Lesson: Logical Operator | 1 Hour | ----------------- |
| Practice Coding: Logical Operator | 40 Minutes | ----------------- |
| Lesson: While Loop | 1 Hour | ----------------- |
| Practice Coding: While Loop | 40 Minutes | ----------------- |
| Lesson: For Loop | 1 Hour | ----------------- |
| Practice Coding: While Loop | 40 Minutes | ----------------- |
| Lesson: Key Points to Remember | 10 Minutes | ----------------- |
| Activity: Acronym | 30 Minutes | ----------------- |
| Deadline of All Lesson 1 Activities | -------------- | April 5, 2021 |



* Submission Guidelines
* **Deadline** for all lesson 1 activities: **April 5**, 2021 (Monday) at 4 PM
* Encode your answer on a blank Microsoft Word document
* Convert your file to **PDF** or **picture** format
* Send your output through the **Assignment** **Tab** of **Microsoft Teams**
* Submit the two activities at the same time.
* Submit your answer only. DO NOT send the module
* Save your files as:

**Grade&Section.Subsection – Last Name, First Name – Title of Activity**

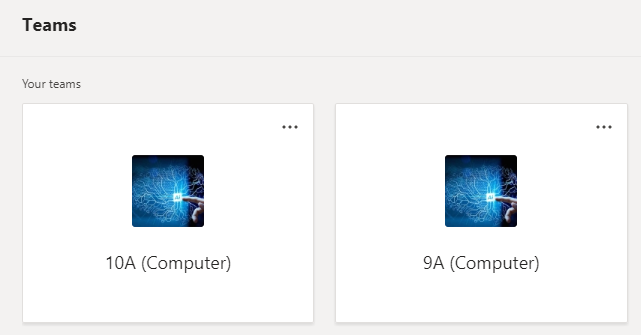
Examples:

10A**.**1 – Gonzales, Faye – Word Sight

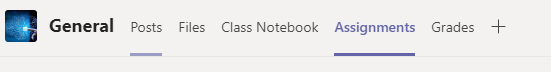
10A**.**1 – Gonzales, Faye – Acronym

* Steps in submitting your output through Assignment Tab of Microsoft Teams

1. Log in to your MS Teams account
2. Click **Teams**
3. Click your Computer Team. Example: 10A (Computer)



1. Click **Assignments**



1. Click **Fourth Grading Module and Lesson 1 Submission**
2. Click **Add work**
3. Upload your file
4. Click the **Turn in** button to submit



**Note:** A checklist of activities has been provided for you.



* Let’s Recall

Lesson 1: Matching Type Answer

Column A Column B

D 1. Program A. Describes a value

J 2. Database B. Creating applications

F 3. Programming Language C. Notes

K 4. Code D. Set of finished functional codes

L 5. Variable E. Seen in website

H 6. Programmer F. Python

A 7. Data Type G. Improper Indentation

G 8. Error H. Computer memory

C 9. Comment I. Guido van Rossum

B 10. Programming J. Virtual storage

K. Can contain errors

L. number

Lesson 2: Wordscapes Answer



U

T

N

O

I

N

F

C

**C**

**N**

**T**

**U**

**F**

**O**

**N**

**O**

**C**

**T**

**I**

**F**

**N**

**O**

**O**

**I**

**F**

**U**

**C**

**N**

**T**

**I**

**O**

**N**

**N**

**U**

**I**

**T**

**C**

**O**

**N**

**U**

**N**

**I**

**N**

**N**

**C**

**T**

**I**



* Let’s Spice It Up!

**WORD SIGHT:** Analyze the picture and guess the wordthat the picture depicts. Complete the word box below to form the quote using the pictures presented as a hint. Follow the arrow to know the arrangement of the words.

CN: B8 Name: Josh Espanola Score: \_\_\_\_\_\_\_\_\_\_\_

Grade and Section: 10A.2 Teacher: Miss Faye Cyd A. Gonzales Date: March 25, 2021

A picture containing diagram

Description automatically generated

**D**

T H E



F

**F I**

**R S T THVVHVHVHVHVVHTTTT T**

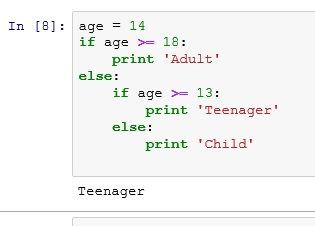
**W**

**R**

**I**

**T**

**E**







**S**

**O**

**E**

**D**

**O**

**C**

**T**

**X**

**E**

**N**

**E**

**V**

**L**

Icon

Description automatically generated

**D**

**H**

**P**

**B**

**R**

**O**

**L**

**E**

**M**

**E**

**T**

Quote: **“ First solve the problem, next write the code.”**

*Always remember the quote! ☺*



* Python Flow Control

*Let us discuss and code know how to control the flow of codes in Python. Are we going to make a loop or just a continuous flow?*

**Assignment Operator**

* **Operators** are used to perform mathematical operations on values and variables

*In the previous grading period, we have discussed some of the basic operators used in Python. Let us widen our knowledge with the other operators used in Python.*

* **Assignment operator** is an operator used to assign values to certain variables

|  |  |  |  |
| --- | --- | --- | --- |
| **Comparison Operator** | **Description** | **The Same As** | **Example** |
| **=** | Assigning value to a certain variable | num = 6 | num = 6 |
| **+=** | Adding a certain number to itself | num = num + 6 | num = 5  num += 2 *Same as 5+2*  print(num)  output: 7 |
| **-=** | Subtracting a certain number to itself | num = num - 6 | num = 8  num -= 4 *Same as 8-4*  print(num)  output: 4 |
| **\*=** | Multiplying a certain number to itself | num = num \* 6 | num = 3  num \*= 7 *Same as 3\*7*  print(num)  output: 21 |
| **/=** | Dividing a certain number to itself | num = num / 6 | num = 9  num /= 2 *Same as 9/2*  print(num)  output: 4.5 |
| **%=** | Multiply its remainder part to a certain number | num = num % 6 | num = 20  num %= 13 *Same as (20%13)\*13*  print(num)  output: 7 |
| **\*\*=** | Exponentiate itself to a certain number | num = num \*\* 6 | num = 11  num \*\*= 3 *Same as 11\*\*3*  print(num)  output: 1331 |

**Logical Operator**

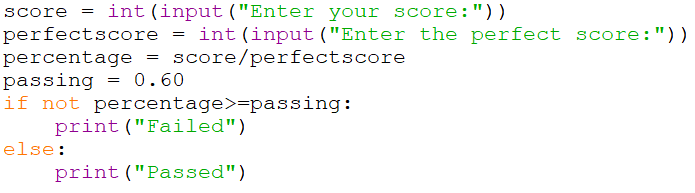
* **Logical operator** is an operator used to combine conditional statements.
* **Conditional statements** are used to express conditions. Most of the time logical operators are used in for loop, while loop and if function.
* **and** and **or** logical operators may have more than two conditions.
* You may use open and close parenthesis **( )** to group conditions.

|  |  |  |
| --- | --- | --- |
| **Logical Operator** | **Description** | **Syntax** |
| **and** | Returns true if both conditions are met | x and y  *where x and y are conditional statements* |
| **or** | Returns true if one of the conditions are met | x or y  *where x and y are conditional statements* |
| **not** | Returns true if the condition is contradicted. | not x  *where x is a conditional statement* |

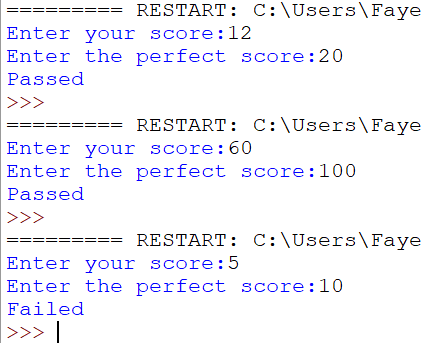
* Examples:

1. Situation: Code that will compute for the passing score. The passing score is 60% of the perfect score. Display “Passed” if the user obtained at least 60% of the passing score, otherwise display “Failed”.

Code:



Sample Output:



Explanation:

score = int(input("Enter your score:"))

*Let the user enter a score in whole number.*

perfectscore = int(input("Enter the perfect score:"))

*Let the user enter a perfect score in whole number.*

percentage = score/perfectscore

*Divides the score by the perfectscore to compute for the percentage.*

passing = 0.60

*Assign a constant value of 0.60 to indicate the passing percentage which is 60%. This value will be used as a comparison for the passing percentage and the percentage of the score that the user got.*

if not percentage>=passing:

***percentage>=passing*** *is the condition. The word* ***not*** *contradicts this condition. So once the condition is contradicted, then the computer will execute the return statement.*

print("Failed")

*Display “Failed” if the percentage of the score over the perfect score is lesser than 0.60.*

else:

print("Passed")

*Display “Passed” if the percentage of the score over the perfect score is lesser than 0.60.*

Another Valid Code:

score = int(input("Enter your score:"))

perfectscore = int(input("Enter the perfect score:"))

percentage = (score/perfectscore)\*100

passing = 60

if not percentage<passing:

print("Passed")

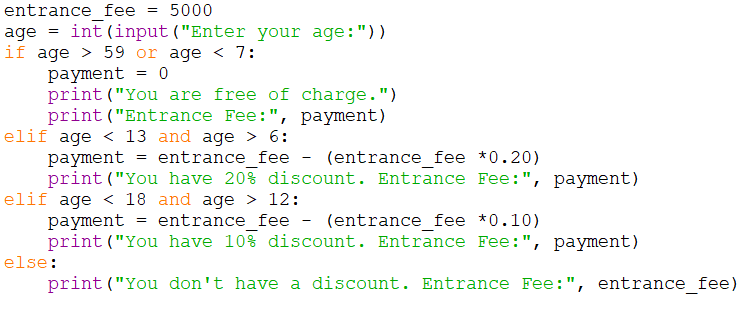
else:

print("Failed")

1. Situation: Code that will compute for the entrance fee base on the age of the client. The entrance fee is 5000. A certain discount is given to clients base on their certain age:

* Under 7 years old: free entrance fee
* 7 to 12 years old: 20% discount on the entrance fee
* 13 to 17 years old: 10% discount on the entrance fee
* 18 to 59 years old: no entrance fee discount
* 60 years old and above: free entrance fee

Code:



Sample Output:



Explanation:

entrance\_fee = 5000

*5000 is the value of entrance\_fee variable*

age = int(input("Enter your age:"))

*let the user enter his/her age*

if age > 59 or age < 7:

*If either one of the conditions is met, then the computer will execute the return statements. Creates a condition to check if the age is either greater than 59 or lesser than 7. The age should range from 60 and above or 0 to 7 (since the condition will check if one of the conditions is met so that is either the age is greater than 59 or lesser than 7) for the computer to execute the following return statements:*

payment = 0

print("You are free of charge.")

print("Entrance Fee:", payment)

elif age < 13 and age > 6:

*If both conditions are met, then the computer will execute the return statements. Creates a condition to check that age should be lesser than 13 and greater than 6. The age should range from 7 to 12 (since 7 to 12 is both greater than 6 and lesser than 13) for the computer to execute the following return statements:*

payment = entrance\_fee - (entrance\_fee \*0.20)

print("You have 20% discount. Entrance Fee:", payment)

elif age < 18 and age > 12:

*If both conditions are met, then the computer will execute the return statements. Creates a condition to check that age should be lesser than 18 and greater than 12. The age should range from 13 to 17 (since 13 to 17 is both greater than 12 and lesser than 18) for the computer to execute the following return statements:*

payment = entrance\_fee - (entrance\_fee \*0.10)

print("You have 10% discount. Entrance Fee:", payment)

else:

*Else statement doesn’t need a condition since it will automatically catch the other conditions not indicated on the if and elif conditions. The condition not indicated is if the age ranges from age 18 to 59. If the age ranges from 18 to 59, then the computer will execute the following return statements:*

print("You don't have a discount. Entrance Fee:", entrance\_fee)

Another Valid Code:

entrance\_fee = 5000

age = int(input("Enter your age:"))

if age >=13 and age <= 17:

payment = entrance\_fee - (entrance\_fee \*0.10)

print("You have 10% discount. Entrance Fee:", payment)

elif age >= 18 and age <=59:

print("You don't have a discount. Entrance Fee:", entrance\_fee)

elif age >= 60 or age <=6:

payment=0

print("You are free of charge. Entrance Fee:", payment)

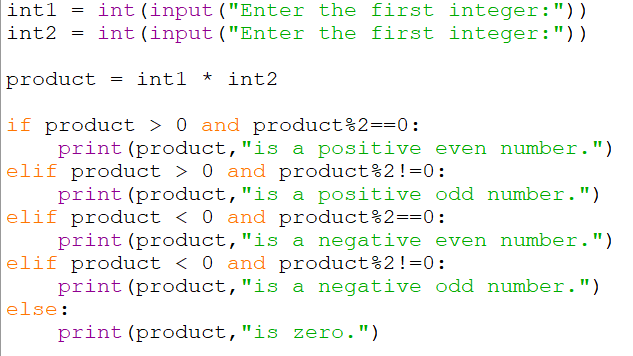
else:

payment = entrance\_fee - (entrance\_fee \*0.20)

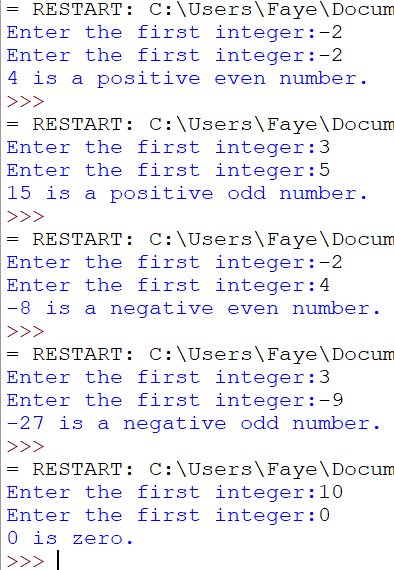
print("You have 20% discount. Entrance Fee:", payment)

1. Situation: Code that will display the product of two entered integer numbers. Then display whether the product is one of the following remarks: positive even number, positive odd number, negative even number, negative odd number, or zero.

Code:



Sample Output:



Explanation:

int1 = int(input("Enter the first integer:"))

*Let the user enter an integer.*

int2 = int(input("Enter the first integer:"))

*Let the user enter another integer.*

product = int1 \* int2

*Computes for the product of the value of int1 and int2.*

if product > 0 and product%2==0:

*If both conditions are met, then the computer will execute the return statements. If the value of the product is both positive (greater than 0) and even (if the remainder is equal to zero, then the integer is even), then the computer will execute the return statement:*

print(product, "is a positive even number.")

elif product > 0 and product%2!=0:

*If both conditions are met, then the computer will execute the return statements. If the value of the product is both negative (lesser than 0) and even (if the remainder is not equal to zero, then the integer is even), then the computer will execute the return statement:*

print(product, "is a positive odd number.")

elif product < 0 and product%2==0:

print(product,"is a negative even number.")

elif product < 0 and product%2!=0:

print(product,"is a negative odd number.")

else:

*If the entered integer is not positive, not negative, not even, and not odd, then the integer is zero.*

print(product, "is zero.")

**While Loop**

* Loop pertains to something being repeatedly done.
* **While loop** executes the while loop body statement as long as a given condition is true. The while loop body statement will be executed repeatedly unless the loop is terminated by a certain code.
* While loop syntax 1:

**while condition:**

**while loop body statement**

* The **condition** is any expression that can be evaluated as either true or false.
* The **while loop body statement** is the expression that will be executed once the given condition in the while loop is true. The while loop body statement is determined by the indentation. Anything included inside the indentation of the while loop is part of the while loop body statement.
* An iteration might be included in the while loop body statement. The **iteration** is used to repeatedly go over a sequence. The sequence in the while loop is the while loop body statement. To go over a sequence and display different values, the value of a certain variable needs to increment or decrement. Most of the time, the iteration is done at the end of the while loop body statement.
* While loop syntax 2:

**while condition:**

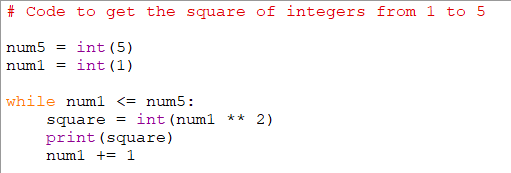
**while loop body statement**

**else:**

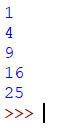
**else body statement**

* Oftentimes, the while loop has a counterpart which is the else statement. The **else statement** will automatically catch other conditions not indicated in the while loop condition. The else statement does not include a condition. The else statement is **optional** but if needed, it must be included. Else statement is a separate statement from the while loop.
* The **else body statement** is the expression that will be executed once the given condition in the while loop is false. The else body statement is determined by the indentation. Anything included inside the indentation of else statement is part of the else body statement.
* Give a limit to the loop, or else the loop will continue forever. The limit can be set in the condition and/or the iteration.
* You may use open and close parenthesis **( )** to group conditions.
* While loop example 1:

Situation: Code to compute for the square of integers from 1 to 5



Output:



Explanation:

num5 = 5 *Initialize the value of num5 which is 5.*

num1 = 1 *Initialize the value of num1 which is 1.*

while num1 <= num5: *Condition: “As long as the value of num1 is lesser than or equal to num5”*

square = num1 \*\* 2 *The first value of num1 is 1 since that is the initialized value. The value of*

*the square variable is the square of num1. \*\* is exponentiation.*

print(square) *The computer will display the square value of num1 starting from 1*

num1 += 1 *The value of num1 will increase by 1, indicated by* ***+=1*** *. Meaning, the first*

*value of num1 is 1 and it will continue to increment. But since it is indicated in the condition that if num1 is already greater than 5, the loop will be terminated.* ***num1 +=1*** *is the iteration.*

*The code goes like this:*

*The value of num5 is 1. The value of num1 is 1.*

*First loop: Check if the value of num1 is lesser than or equal to num5. Since num1 is 1 and is lesser than 5, square 1, then display 1 on the screen. Increment the value of num1 by 1. Now, the value of num1 is 2.*

*Second loop: Check if the value of num1 is lesser than or equal to num5. Since num1 is 2 and is lesser than 5, square 2, then display 4 on the screen. Increment the value of num1 by 1. Now, the value of num1 is 3.*

*Third loop: Check if the value of num1 is lesser than or equal to num5. Since num1 is 3 and is lesser than 5, square 3, then display 9 on the screen. Increment the value of num1 by 1. Now, the value of num1 is 4.*

*Fourth loop: Check if the value of num1 is lesser than or equal to num5. Since num1 is 4 and is lesser than 5, square 4, then display 16 on the screen. Increment the value of num1 by 1. Now, the value of num1 is 5.*

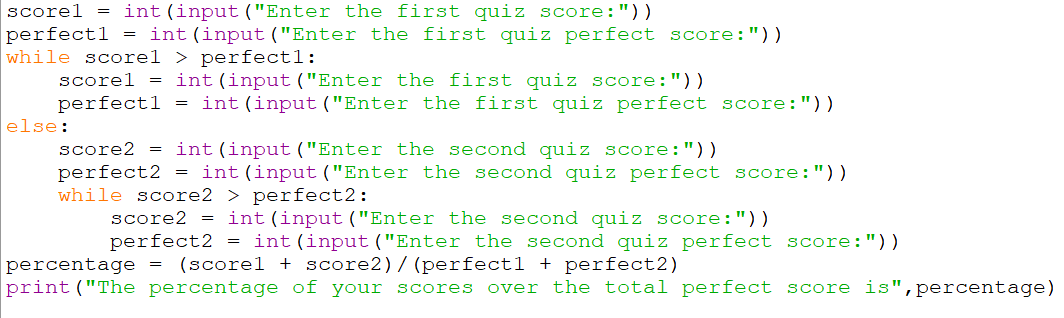
*Fifth loop: Check if the value of num1 is lesser than or equal to num5. Since num1 is 5 and is equal to 5, square 5, then display 25 on the screen. Increment the value of num1 by 1. Now, the value of num1 is 6.*

*Sixth loop: Check if the value of num1 is lesser than or equal to num5. Since num1 is 6 and is not lesser than nor equal to 5 therefore, terminate the loop.*

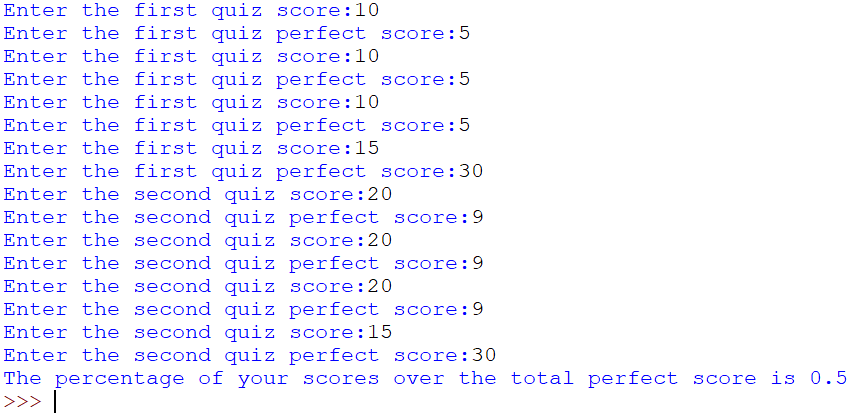
* In the while loop, the condition is checked first. If the condition is met, the while loop body statement is executed. If there is an iteration, the iteration is executed then the condition is checked again. The process continues unless the condition is evaluated to be false.
* While loop example 2:

Situation: Code that will let the user enter three quiz scores and three quiz perfect scores; prevent the user from entering a score greater than the perfect score; display the percentage of the total score over the total perfect score.

Code:



Sample Output:



Explanation:

score1 = int(input("Enter the first quiz score:"))

*Lets the user enter a score*

perfect1 = int(input("Enter the first quiz perfect score:"))

*Lets the user enter a perfect score*

while score1 > perfect1:

*As long as the user will enter a score that is greater than the perfect score, the computer will continue to ask the user to re-enter a score and a perfect score.*

score1 = int(input("Enter the first quiz score:"))

perfect1 = int(input("Enter the first quiz perfect score:"))

else:

*If the score is lesser than or equal to the perfect score, then the computer will ask for the user to enter the second score and second perfect score.*

score2 = int(input("Enter the second quiz score:"))

perfect2 = int(input("Enter the second quiz perfect score:"))

while score2 > perfect2:

*The computer needs to check that the second score should be lesser or equal to the second perfect score*

*for it to do the computation. As long as the user will enter a score that is greater than the*

*perfect score, the computer will continue to ask the user to re-enter a score and a perfect score.*

score2 = int(input("Enter the second quiz score:"))

perfect2 = int(input("Enter the second quiz perfect score:"))

percentage = (score1 + score2)/(perfect1 + perfect2)

*If the user has entered a score lesser than or equal to the perfect score, then the computer will perform the computation to get the percentage of the total score over the perfect score.*

print("The percentage of your scores over the total perfect score is", percentage)

*The computer will display a certain message and the value of the percentage variable on the screen.*

**For Loop**

* **For loop** is used for iterating over a sequence.
* **For loop** executes the for loop body statement in every iteration. The for loop body statement will be executed over and over again unless the loop is terminated by a certain code.
* **Iterate** means to repeatedly go over a sequence. On every iteration it takes the next value from the sequence until the end of a sequence is reached.
* For loop syntax:

**for variable in sequence:**

**for loop body statement**

* For loop syntax with else statement:

**for variable in sequence:**

**for loop body statement**

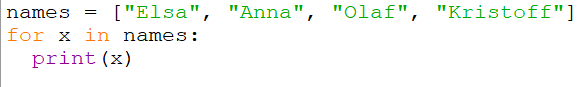
**else:**

**else body statement**

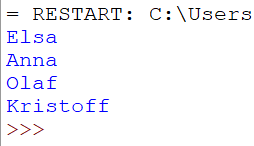
* The **variable** in the for loop syntax is used to iterate over the sequence.
* The **sequence** that the variable will iterate over is the set of values.
* The word **in** is used to check the values of a sequence or the value/s of a variable.
* **For loop body statement** is the return statement that the computer will execute every time that the variable will iterate over a certain sequence. The for loop body statement is determined by the indentation. Anything included inside the indentation of for loop is part of the for loop body statement.
* Oftentimes, the for loop has a counterpart which is the else statement. The **else body statement** will be executed once there is no sequence to be iterated in the for loop. Anything included inside the indentation of else statement is part of the else body statement.
* **break** is used to terminate the current loop and resume to the next statement.
* You may use open and close parenthesis **( )** to group conditions.
* For loop example 1:

Situation: Code that will display each name in the names list.

Code:



Output:



Explanation:

names = ["Elsa", "Anna", "Olaf", "Kristoff"]

*We created a set of values using the list data type.*

for x in names:

*x is the variable that will iterate over the set of values stored in names. We say that iterate means to repeatedly go over a sequence. Since the sequence in our example is the values of names, then x will go over the names until it reaches the last value which is Kristoff.*

print(x)

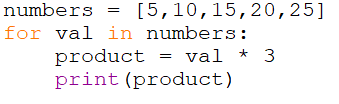
*Used to display the value of the iteration. The first iteration is Elsa, followed by Anna, then Olaf, and lastly*

*Kristoff.*

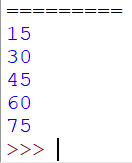
* For loop example 2:

Situation: Code that will compute for the product of the values multiplied by 3 and display the product.

Code:



Output:



Explanation:

numbers = [5,10,15,20,25]

*We created a set of values using the list data type.*

for val in numbers:

*val is the variable that will iterate over the set of values stored in numbers.*

product = val \* 3

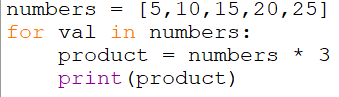
*Will multiply the iterated value by 3.*

print(product)

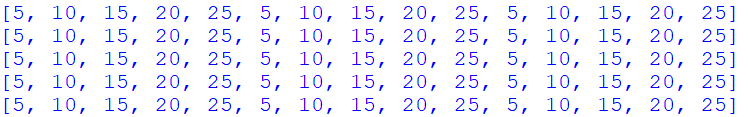
*Used to display the product.*

**Note:** We did not use the variable numbers to multiply the values by 3, because this will only result in displaying the values of numbers 15 times. Iterated 5 times (since there are five values which are 5,10,15,20 and 25) and multiplied by 3 resulting in 15 displayed values. If this is the case, we are not displaying the desired output.

*Code that is logically wrong:*



*Not the Desired Output:*

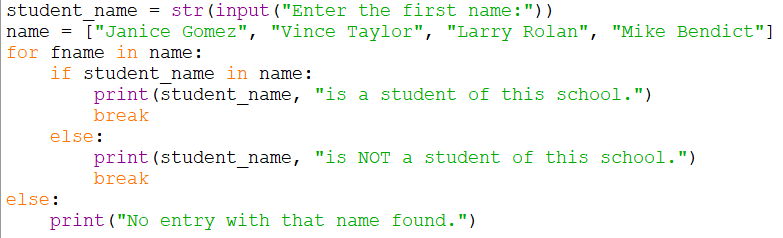


* For loop example 3:

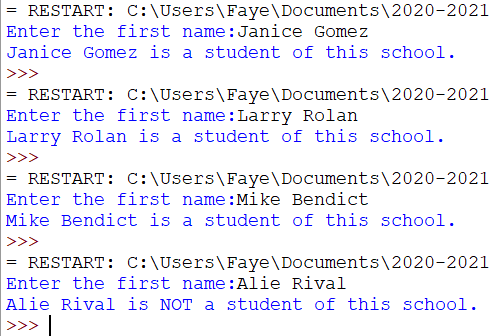
Situation: Code that will check if the entered name matches the record of the school.

*For this example, we assume that there are only four students.*

Code:



Sample Output:



Explanation:

student\_name = str(input("Enter the first name:"))

*Lets the user enter a name.*

name = ["Janice Gomez", "Vince Taylor", "Larry Rolan", "Mike Bendict"]

*Create a list of names.*

for fname in name:

*fname is the variable that will iterate over the set of values stored in the name.*

if student\_name in name:

*checks if the entered student\_name exists in the values stored in the variable name. If it exists, then the*

*computer will execute the following return statement:*

print(student\_name, "is a student of this school.")

break

***break*** *is used to terminate the current loop and proceed to the next statement. In our example, there is*

*no next statement.*

else:

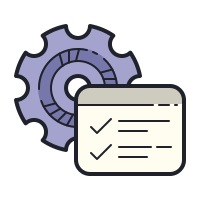
print(student\_name, "is NOT a student of this school.")

break

else:

*This else statement is optional. The code will still work accordingly even without it.*

print("No entry with that name found.")



* Key Points to Remember
* Online Python compiler: https://www.programiz.com/python-programming/online-compiler/
* Offline Python compiler: go to https://www.python.org/downloads/

Download the offline application by clicking on “Download Python 3.9.1” for Windows OS.

Download the offline application by clicking on “Mac OS X” for Mac OS.

* and logical operator will execute the return statement if all conditions indicated are met.
* or logical operator will execute the return statement if one of the conditions indicated is met.
* not logical operator will execute the return statement if the condition is contradicted.
* In the while loop, the condition is checked first. If the condition is met, the while loop body statement is executed. If there is iteration, the iteration is executed then the condition is checked again. The process continues unless the condition is evaluated to be false.
* While loop will execute the while loop body statement as long as the condition is met.
* For loop will execute the for loop body statement upon iteration.
* While loop function is like saying, “as long as…then do this”. While for loop function is like saying, “in every iteration…do this”
* In while loop, the never-ending cycle depends on the condition. If the condition is not met, then the cycle ends. On the other hand, the cycle in for loop depends on the sequence. Once the end of the sequence is reached, then the cycle ends.
* break is used to terminate the current loop.
* You may use open and close parenthesis **( )** to group conditions.



* Reflection

**ACRONYM**: Using the acronym PROGRAMMING, create a quote, trivia, definition, points to remember, phrase, or sentence related to programming. Each acronym should contain a different phrase or sentence. Do not copy the given example.

*Example:*

*P Python is a high-level programming language use to create applications.*

*R Running your code will let you test if your code is working accordingly.*

*O Online games are one of the products of programming.*

*G Greater than or equal to symbol is indicated by >=.*

*R Rules in naming variables should always be remembered.*

*A Always follow proper indentation.*

*M Multi-line comment should start and end with a triple-double quotation.*

*M Multiplication in programming is indicated by asterisk \**

*I Iterate means to go over a certain sequence in the code.*

*N Not equal operator is indicated by !=.*

*G Go over your code to check for correct syntax.*

CN: B8 Name: Josh Espanola Score: \_\_\_\_\_\_\_\_

Grade and Section: 10A.2 Teacher: Miss Faye Cyd A. Gonzales Date: April 4, 2021

P – Programming encourages us in becoming adaptable and more knowledgeable individual by enabling us to step outside of our comfort zone.

R – Retry on developing your codes when it’s not working, don’t lose hope and keep on trying!

O – One of the examples of Programming Languages is Python that is beneficial in our entertainment nowadays.

G - Games like Valorant and Mobile Legends keep us entertained this quarantine because this is a finished product of coding.

R - Raw products of programming is called a “Code”.

A – Always put in your best effort in honing your programming skills because it will benefit both ourselves and our society. It may be complicated, but bear in mind that it is well worth the effort.

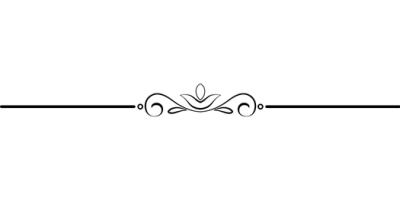
M – Memorize your skills in developing coding because this is our foundation to be a successful person in the future generation.

M – Mike Muuss developed a computer software called Ping that is a computer software that enables us to measure the reachability of a host on an Internet Protocol network. Thanks to coding, because it is extremely beneficial to all of us netizens.

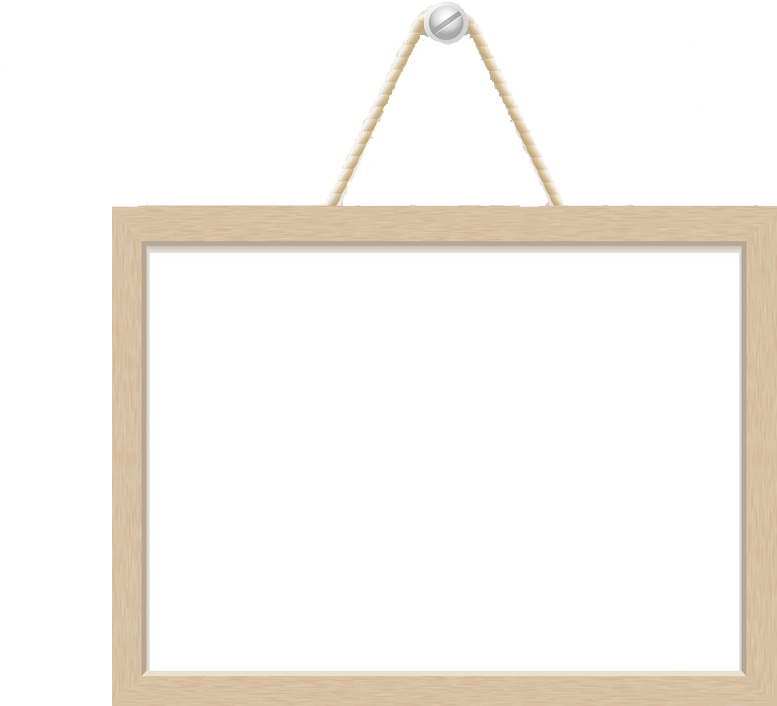
I – Invest your skills in programming as much as possible because it helps you to make innovative and brilliant things since everything are revolving through technologies.

N – Naming codes in an innovative and comprehensible manner makes the outcome of our coding better.

G – Google is the most widely used search engine on the planet. Thanks to programming since it helps our daily tasks, such as productivity and schoolwork more efficient through the finished products of programming.

**Lesson 2: Debugging**





* Learning Objectives

**Checklist of Activities**

* Odd Choice Out
* Exercise (through MS Teams)
* Performance Task
* Reflect n’ Sketch

After completing this module, you should be able to:

* Understand and explain underlying concept of programming;
* Understand and explain flow control of Python;
* Create block of codes free from errors;
* Debug codes;
* Create simple program using Python; and
* Display confidence and commitment in accomplishing all given tasks

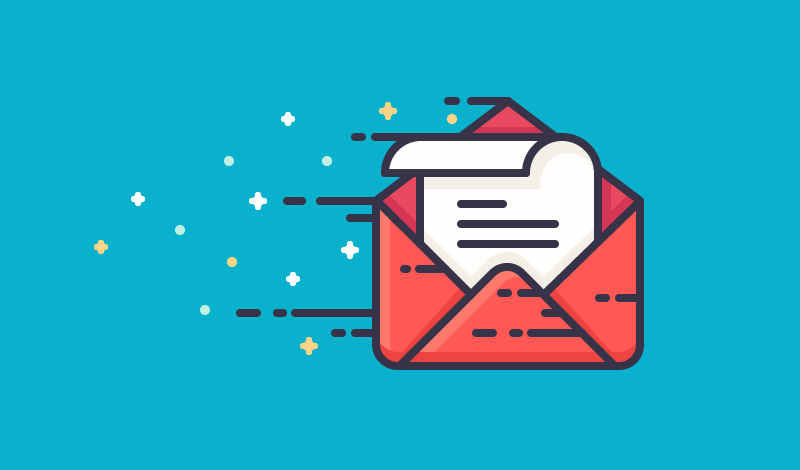


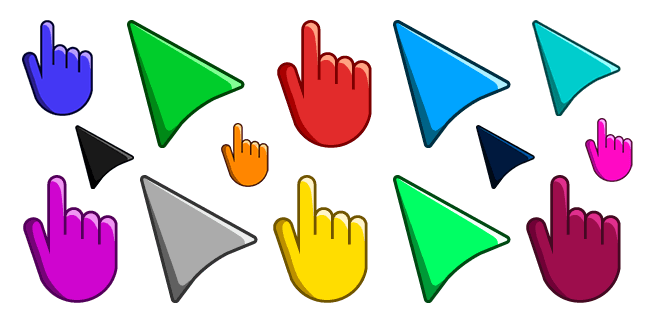


* Time Frame

Here is the third grading time frame to help you manage your time. Give yourself some rest in between. You can do it!

|  |  |  |
| --- | --- | --- |
| Activities | Completion Time | Date |
| Lesson: Let’s Recall | 10 Minutes | ----------------- |
| Activity: Odd Choice Out | 15 Minutes | ----------------- |
| Lesson: Debugging | 30 Minutes | ----------------- |
| Lesson: Syntax Error | 1 Hour | ----------------- |
| Lesson: Logical Error | 1 Hour | ----------------- |
| Practice Coding: Flow Control and If Function | 40 Minutes | ----------------- |
| Practice Coding: Debugging | 40 Minutes | ----------------- |
| Lesson: Key Points to Remember | 10 Minutes | ----------------- |
| Activity: Exercise (through MS Teams Assignment Tab) | 20 Minutes | Until April 19, 2021 |
| Activity: Start of Performance Task | 40 Minutes | ----------------- |
| Activity: Performance Task Continuation | 40 Minutes | ----------------- |
| Activity: Performance Task Editing | 40 Minutes | ----------------- |
| Activity: Performance Task Finalization | 40 Minutes | ----------------- |
| Activity: Reflect n’ Sketch | 40 Minutes | ----------------- |
| Deadline of Lesson 2 activities | -------------- | April 19, 2021 |



* Submission Guidelines
* **Deadline** of Lesson 2 activities: **April 19**, 2021 (Monday) at 4 PM
* Encode your answer on a blank Microsoft Word document
* Convert your file to **PDF** or **picture** format
* Send your output through the **Assignment** **Tab** of **Microsoft Teams**
* Submit all three activities at the same time: Odd One Out, Performance Task, and Reflect n’ Sketch. The exercise is done through MS Teams Assignment Tab.
* Submit your answer only. DO NOT send the module
* Save your files as:

**Grade&Section.Subsection – Last Name, First Name – Title of Activity**

Examples:

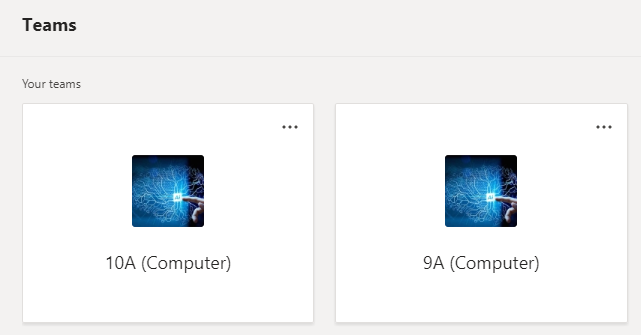
10A**.**1 – Gonzales, Faye – Odd Choice Out

10A**.**1 – Gonzales, Faye – Performance Task

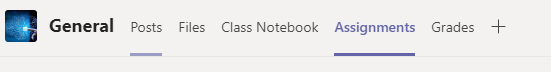
10A**.**1 – Gonzales, Faye – ReflectnSketch

* Steps in submitting your output through Assignment Tab of Microsoft Teams

1. Log in to your MS Teams account
2. Click **Teams**
3. Click your Computer Team. Example: 10A (Computer)



1. Click **Assignments**



1. Click **Fourth Grading Lesson 2 Submission**
2. Click **Add work**
3. Upload your file
4. Click the **Turn in** button to submit



**Note:** A checklist of activities has been provided for you.



* Let’s Recall
* Assignment operator is used to assign values to certain variables.
* Logical operator is an operator used to combine conditional statements.
* Logical operator syntax:

**x and y**

**x or y**

**not x**

*where x and y are both conditional statements*

* while loop syntax:

**while condition:**

**while loop body statement**

**else:**

**else body statement**

* for loop syntax:

**for variable in sequence:**

**for loop body statement**

**else:**

**else body statement**



* Let’s Spice It Up!

**ODD CHOICE OUT:** Read and analyze the connection between the given words. Encircle the word that does not belong to the group. Briefly explain the reason why that encircled word is not part of the group.

CN: \_\_\_\_\_ Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Score: \_\_\_\_\_\_\_\_\_\_\_

Grade and Section: \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Teacher: Miss Faye Cyd A. Gonzales Date: \_\_\_\_\_\_\_\_\_\_\_\_

1. %= \*\*= == =

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. and if not or

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. else for while if

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. condition data type value variable

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. ( ) \*\* ! $

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

*Let’s use those given words in coding.*



* Debugging

*Error is part of the process of a successful program. Errors will also make programmers realize the logic behind the program. Like in a real-life situation, mistakes and failures are part of triumph.*

**Debugging**

* **Debugging** pertains to the process of identifying and removing errors in the code.
* The code should be free from errors to display the desired output.
* Two types of Errors:

1. Syntax Error
2. Logical Error

**Syntax Error**

* **Syntax error** pertains to errors in the code that the computer can detect.
* Syntax error has something to do with the error in the syntax.
* Since the computer can detect syntax errors, it will give you the type of syntax error after running the code. Then simply locate and correct it.
* Some examples of syntax error:

1. Missing code in the syntax

Example: sample = float( ("Enter your allowance:"))

*To correct this, simply supply the missing code. The word input is missing.*

*Correct code:*

sample = float(input("Enter your allowance:"))

1. Additional code in the syntax

Example: sample = float(input("Enter your allowance:")))

*In order to correct this, simply remove the additional code which is the third closing parenthesis. Correct code:*

sample = float(input("Enter your allowance:"))

1. Wrong syntax

Example: sample = input(float("Enter your allowance:"))

*To correct this, simply correct the syntax of your code. In this example, the data type should go first before the word input.*

*Correct code:*

sample = float(input("Enter your allowance:"))

1. Wrong spelling of variable or data type or other functions

Example: sample = input(float("Enter your allowance:"))

print(smple)

*To correct this, simply correct the spelling of your variable.*

*Correct code:*

sample = input(float("Enter your allowance:"))

print(sample)

1. Wrong indentation

Example: sample = input(float("Enter your allowance:"))

print(smple)

*You are not supposed to indent if there is no colon. In addition, indentation is only applicable to codes that belong inside the return statement or body of a loop function.*

*Correct code:*

sample = input(float("Enter your allowance:"))

print(sample)

1. A combination of the said syntax error

*Your code might have missing and additional syntax, some of your variables might have wrong spelling, etc.*

**Logical Error**

* **Logical error** pertains to errors in the code that the computer cannot detect.
* Logical error has something to do with the error in the logical flow of the code.
* Your code might have correct syntax but incorrect logic which will result in the code not displaying the desired output.
* Since the computer can’t detect this kind of error, you must analyze your code by yourself then detect where the error is located and think of the solution. Some logical errors are difficult to debug so you have to think beyond the box.
* There are no types of logical errors because this kind of error occurs due to mistakes beyond the computer’s ability to interpret. Meaning, you created that type of logical error.
* Some examples of logical error:

Example 1: *We want to display the square of the values stored inside the sample variable.*

sample = [1,2,3]

for y in sample:

square = y\*sample

print(square)

*If we want to square the values, then we must square the iterated value.*

*Correct code:*

sample = [1,2,3]

for y in sample:

square = y\*2

print(square)

Example 2: *We want to determine whether an entered integer is odd or even.*

integer = int(input("Enter an integer:"))

if integer%2==0:

print(integer,"is odd")

else:

print(integer, "is even.")

*Focus on the print statement. If the user will enter a value of 1, the computer will display “1 is even”. This code is syntactically correct but it will not produce the desired output.*

*Correct code:*

integer = int(input("Enter an integer:"))

if integer%2==0:

print(integer,"is even")

else:

print(integer,"is odd.")

Example 3: *We want to determine the entrance fee discount base on age. Above 59 years old and below 18 have a 20% discount.*

age = int(input(“Enter your age:”))

if age>59 and age<18:

print(“You have 20% discount.”)

else:

print(“You don’t have a discount.”)

*Focus on the logical operator used. Using the and operator means that both conditions should be true for the computer to execute the return statement. There is only one age. A person can’t be 60 years old and 15 years old at the same time. Meaning, you have to choose between the two conditions. or logical operator should be used instead of and.*

*Correct code:*

age = int(input(“Enter your age:”))

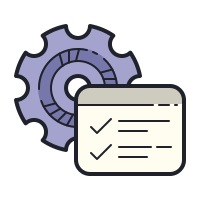
if age>59 or age<18:

print(“You have 20% discount.”)

else:

print(“You don’t have a discount.”)

*The longer the code, the more error you might encounter. You will be having more errors as you do more coding. Error is already part of the process in programming. Errors will help you arrive to the correct algorithm and logic of the code. Be thankful to errors because it helps you think.*



* Key Points to Remember
* Syntax error is not following the rules of the syntax in coding resulting in an error; while logical error is not understanding the flow of logic in creating a functional code that is beyond the computer’s ability to detect resulting in an error.
* Your code might have correct syntax but incorrect logic which will result in the code not displaying the desired output.
* To correct the error, you must think about what, where, and how the error occurred. A logical error is more difficult to detect and correct compared to a syntax error.
* Your code must be error-free for it to be syntactically working and logically correct.

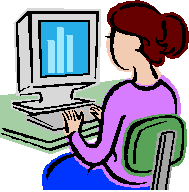


* Exercise

**Exercise**: Python Flow Control and Debugging

Instructions: Answer Fourth Grading Lesson 2 Exercise through MS Teams Assignment Tab. Analyze each question or statement.

1. Log in to your MS Teams account.
2. Click on Computer subject team.
3. Click on the “Assignment” Tab.
4. Click on “Fourth Grading Lesson 2 Exercise”.
5. The exercise is a total of 20 points.
6. The exercise is open until April 19, 2021, at 4 pm.



* Challenge!

**Performance Task**: (R) You are a famous programmer in a prestigious Information Technology firm. (S) You are working on programming an application. (G) You are tasked to create the (P) code of an application that will let the student create his/her log-in account and take the general quiz. (S) Your output will be evaluated in terms of its organization, completeness, function and computation, syntax, logical flow, and output. You will be presenting your code to your (A) team leader for review and approval.

Guidelines:

1. You are tasked to create the code of an application that will let the student create his/her log-in account and take the general quiz. After taking the general quiz, compute and display the score of the student together with its remarks.
2. There are only 10 students in the class. Only these 10 students can create their accounts and can access the general quiz. Each student has a different student ID number as follows:

studentID = 000124524, 524178741, 789451784, 634781222, 965874012, 524178357, 741258964, 963547536, 741635784, 12339955257

1. Let the user enter the student ID number. Only students with the listed studentID can proceed to the next step. This will prevent unauthorized people to have access to the system. If the entered student ID number matches the stored studentID, the application will then let the user create his/her account:

* Let the user enter his/her name; username; password; and password confirmation

1. Do not accept null or blank name, username, and password. If the password confirmation does not match the password, let the user re-enter his/her name, username, password, and confirmation password. This cycle continues until the password matches the password confirmation.
2. Once the password and confirmation password matched, let the student log in to his/her registered account. Check if the username and password match the registered account otherwise, let the user re-enter his/her username and password.
3. Upon successful login, the student can already access and take the quiz. There are only 3 questions and each question has different points. Provide your instructions for the quiz. The student has only one chance to answer each question. Let the student enter his/her answer.

Question 1 (2 points): Who is considered the first computer programmer?

Valid Answers: Ada Lovelace, ada lovelace, Augusta Ada King, augusta ada king

Question 2 (4 points): ENIAC was the first invented computer. (True or False)

Valid Answers: True, true, TRUE

str data type can’t hold space as its value. (True or False)

Valid Answers: False, false, FALSE

*There are two questions for number 2. If one is incorrect, the student will garner only 2 points. If both questions are answered correctly then the student will garner 4 points.*

Question 3: Which of the following are correct examples of object-oriented programming language. (Type the letter only)

1. (2 points) C++, Java
2. (minus 1 point) Python, Java, SQL
3. (4 points) Python, C++, Java
4. (minus 2 points) PHP, SQL, CSS

*For question number 3, each choice can give either points or deductions. Let the user enter the letter of the answer. Do not display the points in the choices. This will give an obvious answer.*

1. Null or blank answer is not accepted. If the answer is blank, let the student answer that certain question again. All questions must be answered.
2. Display the question together with the student’s answer after answering all the questions.
3. Compute and display the score of the student. The perfect score is 15. The passing score is 60% of the perfect score. Display the percentage of the score over the perfect score.
4. Display the following remarks:

If the percentage is 100, display “You are Excellent!”

If the percentage ranges from 90 to 99, display “Very Good!”

If the percentage ranges from 80 to 89, display “Good.”

If the percentage ranges from 60 to 79, display “Not Bad.”

If the percentage is below 60, display “Don’t be disappointed.”

1. Run your code to check it.
2. Copy your code in MS Word.
3. Convert your file to PDF or picture format
4. Save your file as:

Grade&Section.Subsection – Last Name, First Name – Title of Activity

Example: 10A.1 – Gonzales, Faye – Performance Task

1. Submit your output through MS Teams “Assignment” Tab

CN: \_\_\_\_\_ Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Score: \_\_\_\_\_\_\_\_\_\_\_

Grade and Section: \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Teacher: Miss Faye Cyd A. Gonzales Date: \_\_\_\_\_\_\_\_\_\_\_\_

Raw Code *(Copy and paste your code. You may decrease the font size to fit the code within the desired line.)*:

Raw Code in Python Command Line *(Snip your code in Python Command Line.)*:

Snip of sample output *(If the user keeps on entering wrong log-in information.)*:

Snip of sample output *(If the user keeps on entering null values.)*:

Snip of sample output *(If the user skip a question or has a blank answer.)*:

Snip of sample output *(If the user enters incorrect log in information; if question 1 is incorrect; if only 1 question in question 2 is correct; if letter a is the answer of the user in question 3.)*:

Snip of sample output *(If the user enters incorrect log in information; if answer to all questions is incorrect.):*

Snip of sample output *(If the user enters the correct log in information and correct answer to all questions.)*:

**Rubrics**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **CRITERIA** | **Outstanding (10)** | **Satisfactory (8)** | **Developing (6)** | **Beginning (5)** | **RATING** |
| **Organization**  **(How you organize your code)** | The code is exceptionally well organized, on point, very easy to read, and has chronological order. | The code is organized and has a chronological order. The code is somewhat complicated which can be further edited. | The code is organized and has a chronological order. The code is very complicated making some of its content redundant and/or some of its contents are unnecessary. | The code is not organized, not on point, not easy to read, and doesn’t follow chronological order. The code is very complicated and/or some of its contents are unnecessary. |  |
| **Completeness**  **(Format and guidelines)** | Followed the guidelines and formats presented in the module. The complete raw code is present. Complete snips of sample outputs are present. | Some of the guidelines and formats presented in the module were not followed. The complete raw code is present. Complete snips of sample outputs are present. | Some of the guidelines and formats presented in the module were not followed. The raw code has missing parts. Incomplete snips of sample outputs. | Did not follow the guidelines and format presented in the module. The raw code has missing parts. Incomplete snips of sample outputs. |  |
| **Function and Computation**  **(Functions used and correct computation)** | Excellently used the correct syntax of different functions. Shows mastery in declaring variables, using loop functions, and combining Python operators. Perfectly created correct formula for varied computations. | Used correct syntax of different functions. Correctly used variables, loop functions, and Python operators. Created correct formula for varied computations. | Incorrect or inappropriate usage of the following: declaring variables, loop function, if function, input function, print function, operators, and computations. | Functions, operators, and computations are incorrect. Needs great improvement in coding. |  |
| **Syntax**  **(Mastery)** | Exceptional coding skills which show in terms of string messages, spelling, spacing, indentation, variables, equations, functions, and operators. Showed an advanced understanding of coding by applying additional programming techniques. | Correctly used string messages, spelling, spacing, indentation, variables, equations, functions, and operators. | Used string messages, spelling, spacing, indentation, variables, equations, functions, and operators. Code needs further improvement or the length of the code can be narrowed down. Missing some parts of the code. | Completely lost in coding. Doesn’t show evidence of understanding the concept of basic programming using Python. |  |
| **Logical Flow**  **(Logical reasoning)** | The code is  extremely efficient, fully functional, and free from errors displaying the desired output. | The code is complete and functional but contains a minimal error.  Or  The code is complete but was not able to display the desired output due to minimal error. (*Note that even one error can make everything wrong or completely different.)* | The code is present but was not able to display the desired output due to multiple errors. | The code is present but was not able to display the desired output due to multiple errors.  Or  The code is halfway done. |  |
| **Output**  **(Desired Output)** | Created a login account. Only when the username and password match that the user can have access to the questions. Answer and score to all the questions are logically and syntactically correct. All possible answers are correctly included. Prevented the user from entering a null value. Computed and displayed the score and the percentage of the score. Displayed the desired remark. | Created a login account. Only when the username and password match that the user can have access to the questions. Answer and score to all the questions are logically and syntactically correct. Some possible answers are not included. Computed and displayed the score and the percentage of the score. Displayed a certain remark. | Created a login account but can’t check the validity of username and password entered. Created questions but can only verify one answer. Null values are accepted by the computer. The score and percentage of the score are present but incorrect. Displayed a certain remark. | Has completely different output other than the output discussed and presented in the module. |  |



* Reflection

**Reflect n’ Sketch**: Interpret programming through a drawing. Draw a scene that depicts the importance of programming in the community.

Guidelines:

1. Use long or short bond paper.
2. Landscape or portrait.
3. With or without margin.
4. Design your output.
5. Use this format:

|  |
| --- |
| Name:  Grade&Section: |
| *<Your drawing here>* |

1. Take a clear picture of your drawing.
2. Save your file as:

Grade&Section.Subsection – Last Name, First Name – Title of Activity

Example: 10A.1 – Gonzales, Faye – ReflectnSketch

1. Submit your output through MS Teams “Assignment” Tab.

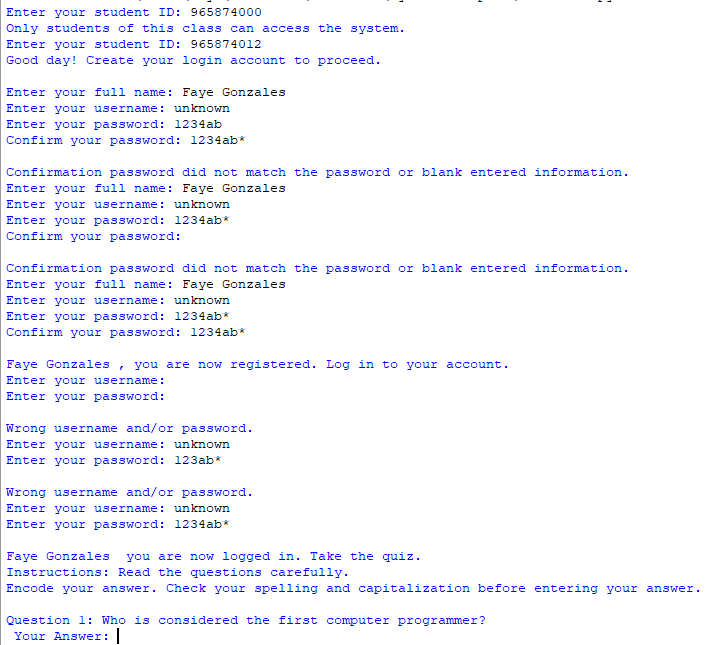


* Appendix

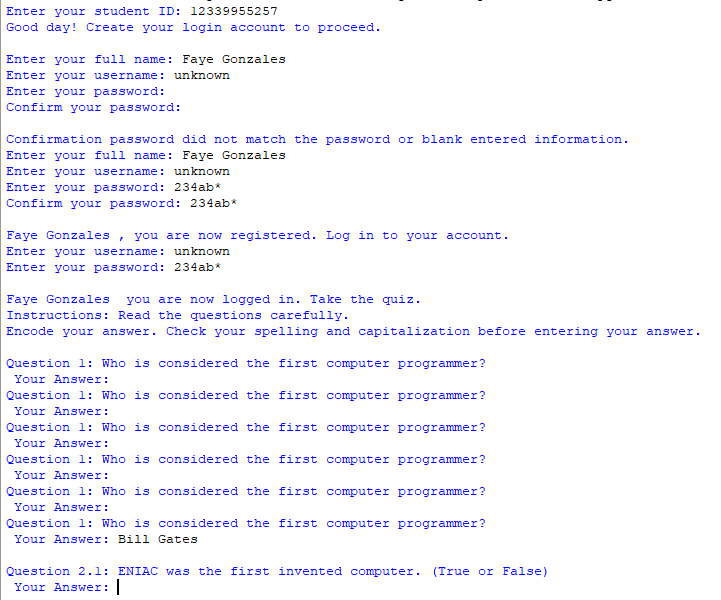
Lesson 2: Performance Task Sample Answer

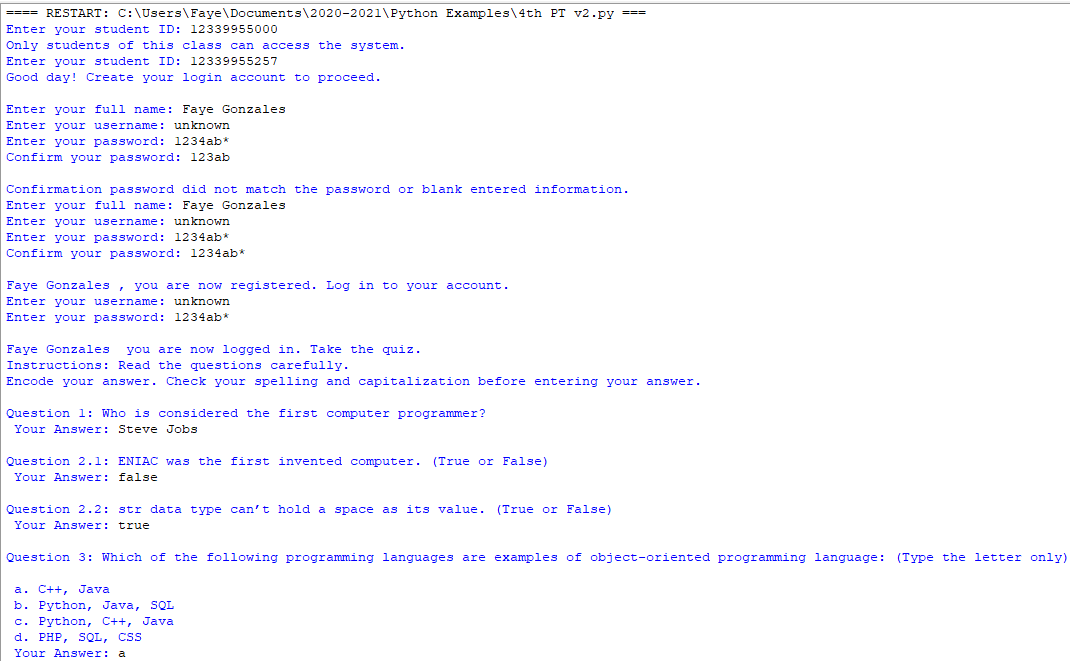
Note: Run your code. Complete the snip. The one indicated here is just a sample.

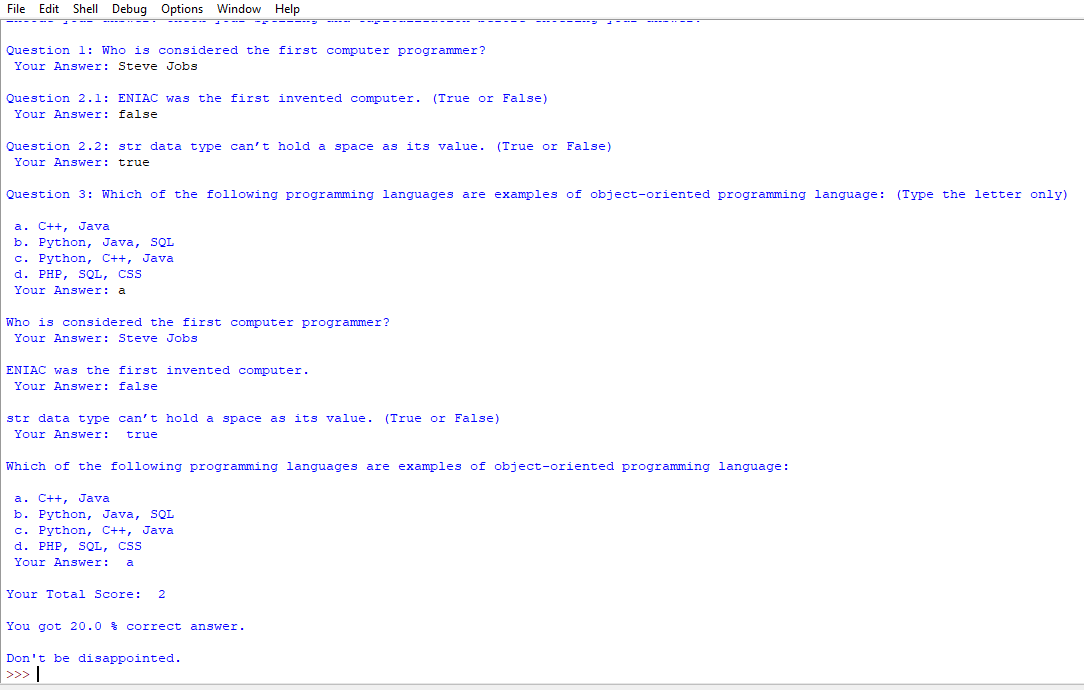
Snip of sample output *(If the user keeps on entering wrong log-in information.)*:



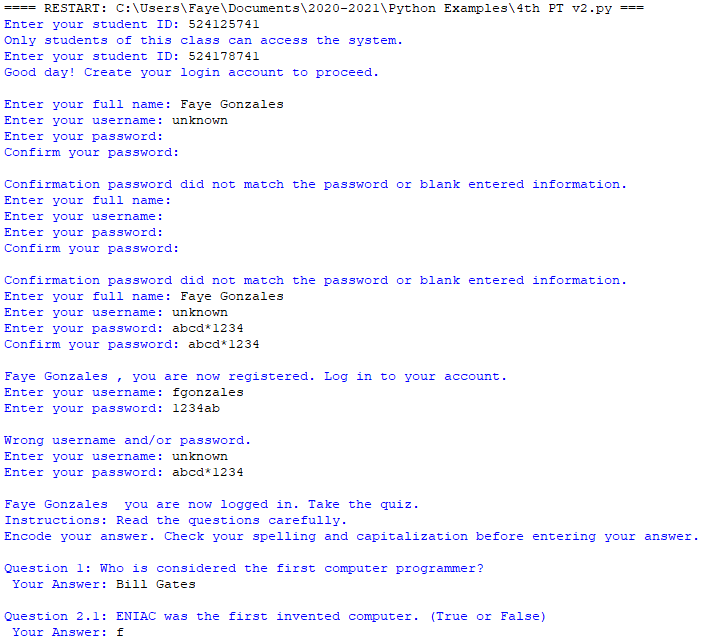
Snip of sample output *(If the user keeps on entering null values in the log in account and questions.)*:

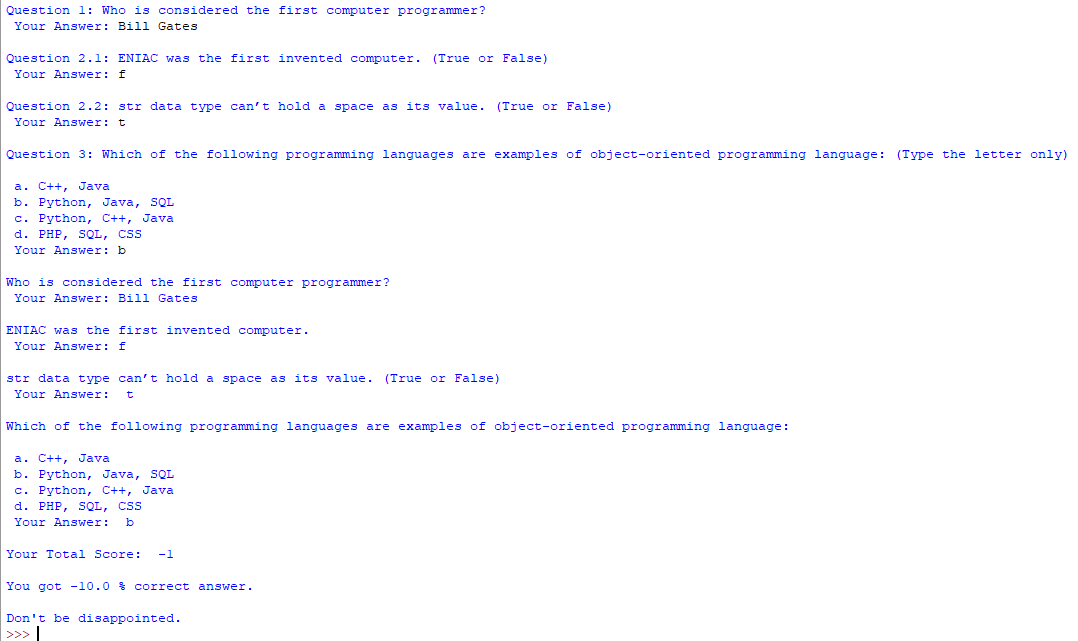


Snip of sample output *(If the user enters incorrect log in information; if question 1 is incorrect; if only 1 question in question 2 is correct; if letter a is the answer of the user in question 3.)*:

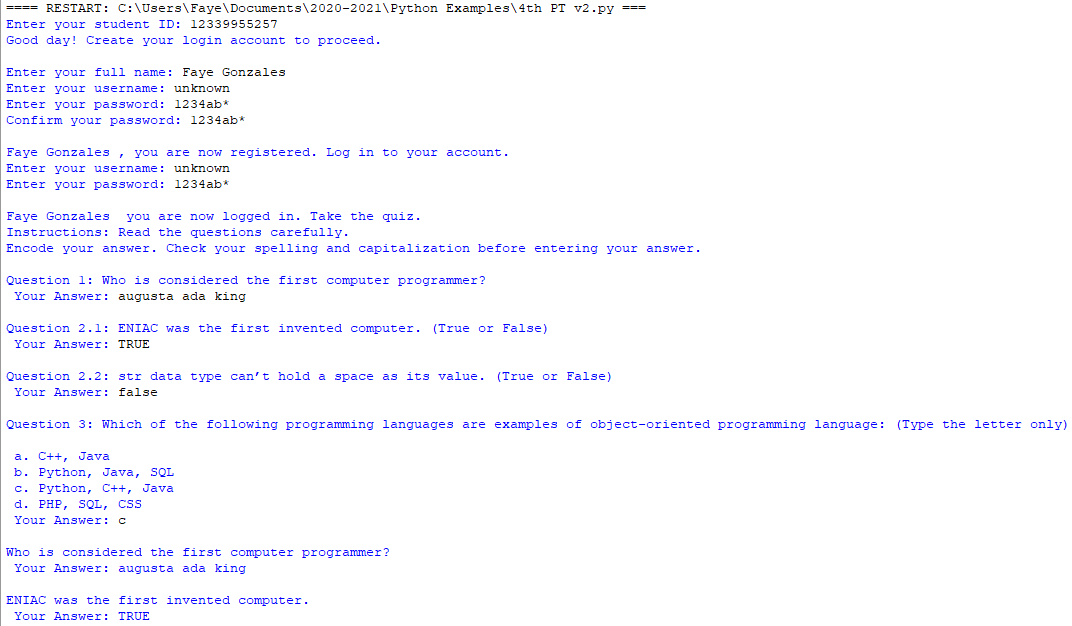


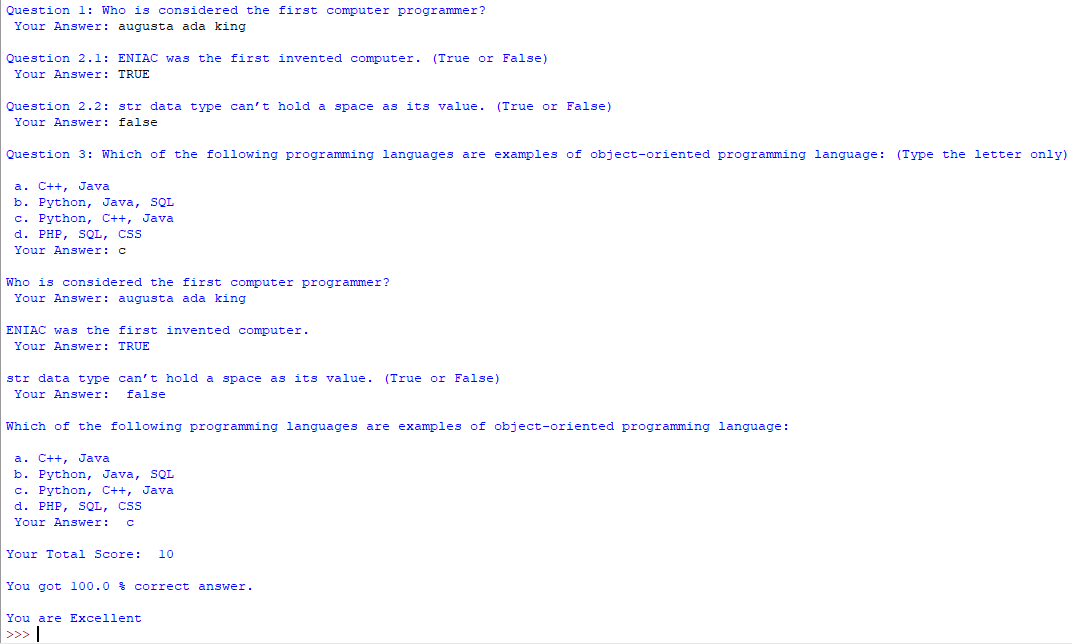
Snip of sample output *(If the user enters incorrect log in information; if answer to all questions is incorrect.):*





Snip of sample output *(If the user enters the correct log in information and correct answer to all questions.)*:







* References

Book References:

Cobre, E.J., et al (2019). ICT Tools Today High School Series Computers for Digital Learners. Quezon City,

Philippines: The Phoenix Publishing House Inc.

Rex Book Store, Inc. (2014). D Whiz in Basic Programming. Quezon City, Philippines: Rex Book Store, Inc.

Website References:

Artyom, P. (2018). Email Marketing Blog. Retrieved from http://blog.notisend.ru/struktura-email-pisma.

Boldsave. (2020). Take Control of Your Finances Today. Retrieved from https://boldsave.co.uk/.

Coolen, M., Vermeulen, W., & Wuytens, G. (n.d.). Gevo Taiji and Qigong. Retrieved from https://gevo-

meeuwen.weebly.com/.

Cryptonix. (n.d.). Security. Retrieved from https://cryptonix.weebly.com/our-services.html.

Custom Cursor. (2020). Cursor Collections. Retrieved from https://custom-

cursor.com/es/collections?page=5&limit=18.

Doncker, E. (n.d.). High Performance Bayesian Applications in Medical, Economics and Climate Sciences.

Retrieved from https://scholarworks.wmich.edu/cgi/viewcontent.cgi?article=

1182&context=grad\_research\_posters.

Emojiisland. (2021). Writing Hand Emoji. Retrieved from https://emojiisland.com/products/

writing-hand-iphone-emoji-jpg .

Icons8. (n.d.). Business Communication. Retrieved from https://icons8.com/icon/pack/business/bubbles.

Larson, B. (2016). Python Conditional Logic. Retrieved from https://analytics4all.org/tag/

python-if-statements-examples/ .

Lumenlearning. (n.d.). Creating a New Blank Document and Finding Your Way Around. Retrieved from

https://courses.lumenlearning.com/wm-compapp/chapter/creating-and-saving-a-file/.

Marsel, M. (2019). Rate my Node. Retrieved from https://medium.com/@blacktangledtongue/battle-of-nodes-

rate-my-node-bb355edd7f7b.

Nicepng. (n.d.). White Board Hanging Png. Retrieved from https://www.nicepng.com/maxp/u2t4t4o0y3q8w7r5/.

Pngio. (2017). Png Warning. Retrieved from https://pngio.com/images/png-a177673.html.

Pngwing. (n.d.). Worried Man. Retrieved from https://www.pngwing.com/en/free-png-ysnhk.

Programiz. (n.d.). Python Online Compiler. Retrieved from https://www.programiz.com/python-

programming/online-compiler/.

Royal. (2019). Solution PNG Transparent Images. Retrieved from http://www.pngall.com/solution-png.

Tom. (2020). Faces. Retrieved from https://www.pinterest.ph/tom9677/faces/.

Tonteo. (2017). Suitable Time to Bag Grapefruit. Retrieved from https://nongdan.pro/tag/trai-buoi/.

Tsu. (n.d.). Master's program "Siberia: resources and modern practices of regional development" / Russian

Studies: Siberia is. Retrieved from http://ib.tsu.ru/ru/siberia/.

Vexels. (2016). First Rank Bandge Olympic. Retrieved from https://www.vexels.com/png-

svg/preview/127647/first-rank-badge-olympic.

Vexels. (2016). Orange round next button. Retrieved from https://www.vexels.com/png-

svg/preview/140916/orange-round-next-button.

Webstockreview. (n.d.). Clipart arrow printable. Retrieved from https://webstockreview.net/explore/clipart-

arrow-printable/.

Webstockreview. (n.d.). Computer Clipart Computer Monitor 773577. Retrieved from

https://webstockreview.net/image/computer-clipart-computer-monitor/773577.html.

Woodard Wood Works. (n.d.). Powerpoint Animation. Retrieved from

https://www.pinterest.co.uk/pin/428264245804772389/.