**Introduction to Software Development**

**Module code: SWE4101**

**June 9, 2021**

Table of Contents

[Task 1: 2](#_Toc74160050)

[1.1 Introduction: 2](#_Toc74160051)

[1.2 List: 3](#_Toc74160052)

[1.3 Tuples: 4](#_Toc74160053)

[1.4 Sets: 4](#_Toc74160054)

[1.5 Dictionary: 6](#_Toc74160055)

[Task 2: 8](#_Toc74160056)

[2.1 Brief synopsis of the problem: 8](#_Toc74160057)

[2.2 How we solve the problem: 8](#_Toc74160058)

[Screenshots and testing result: 11](#_Toc74160059)

[3.1 Run the code: 11](#_Toc74160060)

[3.2 Test with a valid number: 11](#_Toc74160061)

[3.3 Test with a 3 digit invalid number: 12](#_Toc74160062)

[3.4 Test with other input: 13](#_Toc74160063)

[References 14](#_Toc74160064)

[Appendix 15](#_Toc74160065)

# **Task 1:**

## **1.1 Introduction:**

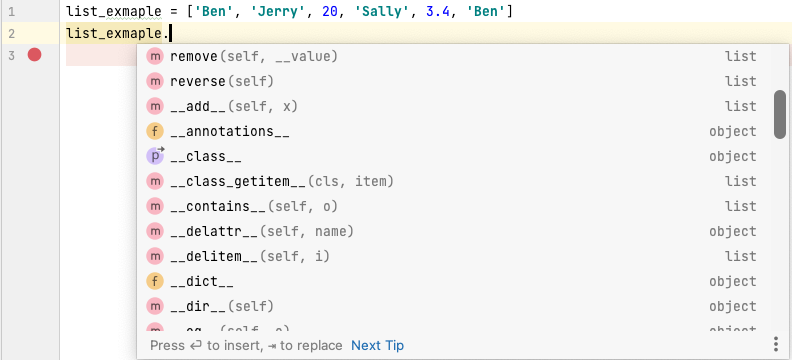
There are two types of variables in any programming language namely primitive and non-primitive. In our discussion we will attempt to discuss the core data structure in python at the beginner level. Our discussion might refer to our previous learning of programming language in C# in visual studio to contextualise our study. For IDE we are using PyCharm 2021.1.1 (Community Edition) Build #PC-211.7142.13, built on April 21, 2021 version: 11.0.10+9-b1341.41 x86\_64 macOS 11.2.3.

Overall user experience on PyCharm IDE is very satisfactory. Pycharm is extremely type friendly and very light editor to work with. PyCharm execute the code much faster than it did with visual studio. One of main reason Python is popular for its high-level built-in data structure. We shall discuss the core data structure of python. Python is also very intelligent in terms of recognising data type as we type. Unlike C# Python recognises the non primitive data structure without being declared first. With the minimal simplicity in syntax and interpreted python has a very easy learning curve for the beginner. In our learning we already been familiar with the int, float, string, varchar, bool etc. At this point our discussion will be centre around the non-primitive data structure.

## **1.2 List:**

List can be easily identified inside the [ ] square bracket with (,) coma separated values. As the name suggest List are more common and widely used data structure than any other data structure that we going to discuss later part of this topic. List follows the rule of indexing and it starts with 0,1,2,3.. so on. We can also store various data type in a list; Once we create a List; it gives us the choice of built in method to search, access, append, insert, extend, reverse, sort, delete etc; that makes List as a obvious choice in the workflow. One of the disadvantages that comes with flexibility is that it takes more memory than other data structure therefore sometimes depending on the requirement list is not the efficient choice.

Example of a list in python:



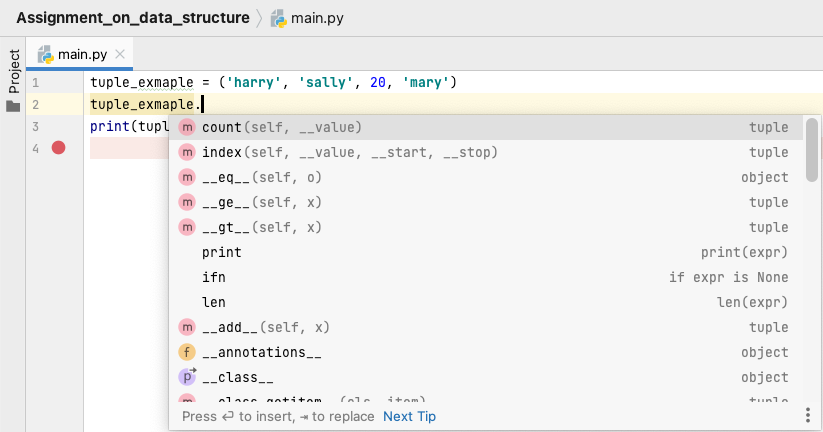
## **1.3 Tuples:**

## 

Tuple are very similar to the list with the major difference is we can’t modify a tuples. Tuples are immutable, however it is possible to create tuples which contain mutable objects, such as lists.

This is efficient choice in a program where we don’t want to change the data value. One of the classic example is a location coordinates. Now how do we identify a tuple? Unlike list we write the value of the tuple inside the round brackets or parenthesis with comma separated value. Inside the tuple we can create a list and mix the data types; keeping in mind that its immutable. Due to the rigidness nature of data structure python only has two built in method we can use in tuple. Count and index. These two method returns the information what we have in our tuple.

Example of a Tuples in python:

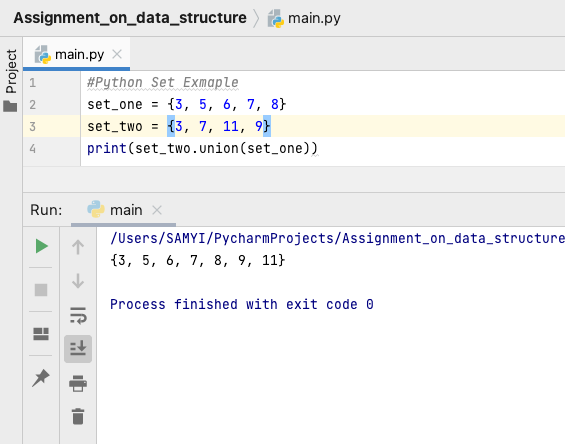


Another powerful feature is unpacking in relation with tuple and list whereby we can simply store a value in multiple of variable by separated them in comma.

## **1.4 Sets:**

Sets can be easily identified with { } curly brackets. Sets are mutable. Set has the distinct feature of unordered with no duplicate value. Meaning that Using set in our programming is extremely efficient to throw away the duplicate value. In addition to regular method like copy, clear, add, remove, update; python set built in method returns the value of union, intersection, subset, disjoint, symmetric etc.

Example of a set in python:

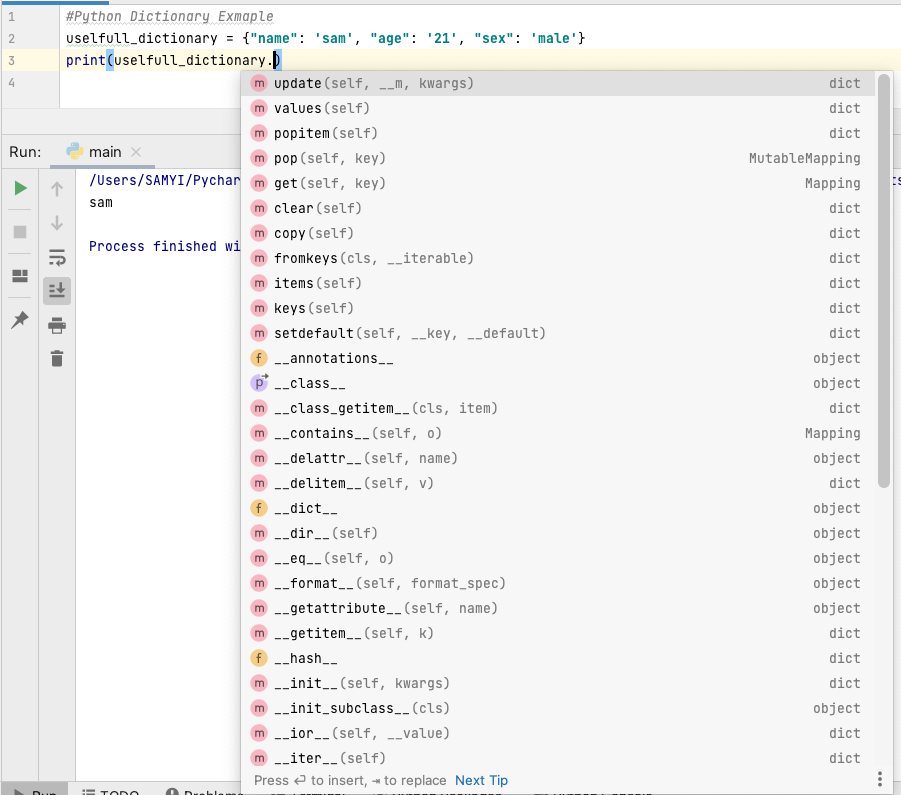
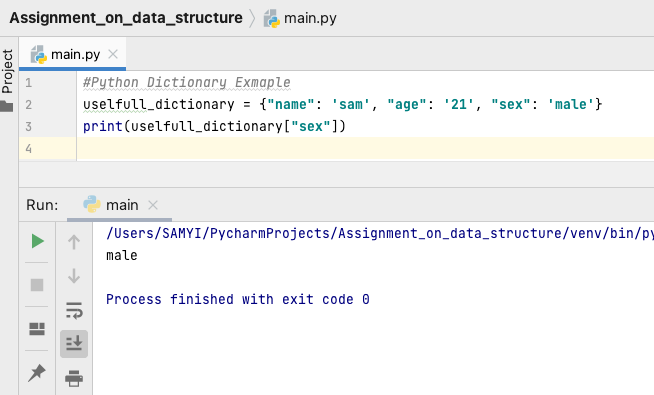




# **1.5 Dictionary:**

Dictionary is a special type of data structure which has many implication in real world. A dictionary does what it says; it is consist of unordered collection of key value pair. Each key value pair maps the key to associate value. In python we express a dictionary within curly braces { } colon : is used to separate each key form its value we separate each item as usual with comma (,). Each key in a dictionary has to be unique if we repeat the key that will overwrite the old value with the new value. We access the value int the dictionary using its key instead of index.

Dictionaries example in python:



**1.6 Core data structure at a glance:**

We can easily identify the our core data structure in the following picture.



[ ]

{ Toy: 2}

{ }

( )

# **Task 2:**

## **2.1 Brief synopsis of the problem:**

Our Second task is a programming task whereby we were asked to program one of the puzzle that Mr. Albert Einstein used to baffle his friend. User is asked to entre the three digit number with one condition the first number must be differ by at least two from the last no. Whatever the number combination it is; the answer will always be 1098. The original game is played on piece of paper and player were asked to reversed the number and subtract the number with the original guessed no. In our program we will automate the steps so that we get our desired result 1098. Throughout our code we try to use the comment tool to explain our code. In addition to the problem solving we attempt to do some error handing. In programming we keep into consideration what are the possible interaction with the user and program. In essence we do the error handing in case any wrong data type entered.

## **2.2 How we solve the problem:**

We start encounter our problem by giving user a welcome message to give a brief about the game. Since we are building a puzzle game; we have used some emoji in our instruction, just to give our player a little bit of more attention to our instruction. Emoji can be access by pressing clrt + cmd + shift on a mac keyboard. Python supports the emoji character in the script which can be useful in a case like this. There is wide speared use of emoji in the title and subtitle in social media content.

Our player will be asked to entre the 3 digit number. In the program we start by saving the input in a variable called original\_number and we convert it with integer. Next we reverse the number. Python has a built in function reversed perhaps that would give us a faster result and shorter code; unfortunately reversed function only returns

Iterator not a string. Instead we will be using the slicing notation of the list function. The slice notation have three parts; start, stop, step. We will be using reversed\_number variable to store the value; so inside the parenthesis we convert our initial original\_number to string and use the slicing function. In our scenario we use [::-1]; that means end to it should to all the sequence. End to beginning counting down by 1.

Next up we will be using if statement to compare the original number and the reversed number we just produced. We should have a two scenario whereby the original\_number may be greater of smaller than the reversed\_number. That means we needed have two if statement. Instead of using the same argument we can simply use a function at the start of the code and later on we can call the function to if statement. We start writing function by using the keyword def and give our function a name with a parameter. Inside our function first we create a list called my\_list. We convert the my\_result variable as string and as a list. Following the previous steps now we will store the value by creating a list that is reverse of using [::-1] means end to beginning counting down by 1. Next step is to convert the integer data type we use .join method here. This method provides a easy way to make strings from iterable objects such as list, string, and tuple. by a string separator (the string on which the join() method is called) and returns the concatenated string. The syntax of the join() method is: string.join(iterable).

If we continue this way we will be repeating the code in two if statement. We can take the advantage of using the same code in a function and later on we call and in two if statements. Let’s name the function “puzzel”. Inside the function we created a list and then we reverse our list with the slicing method and finally we created another variable to converted as integer. At the end of the function we use return keyword to call the final\_result variable. We need to use key word def above our code to make this function work downwards.

Between our function and rest of the code we will still need to make some amendments. Starting with the while loop with True value; this will enable user to come back to the initial stage of the game.

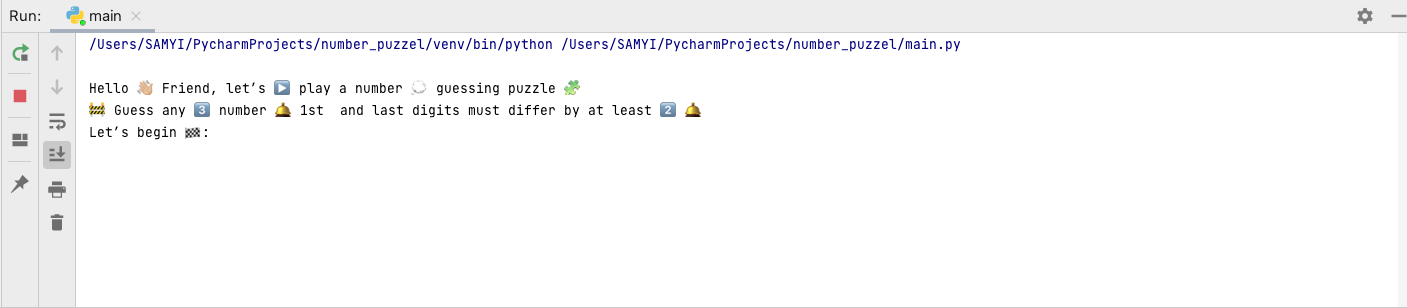
Going back to checklist of the assignment we have our welcome message and instruction all set for the user.

Next step we are going to continue our work with error handing and number validation. If user enters any other character than a positive number we going to show a custom error massage. If the student input a number that doesn’t meet condition of 1st digit differ at least two more or enter less than three digit we will display an error message too. We are using isdigit() built in method to validate our string if it is digit. We use the len() function to validate the length of the string. To compare our digit we created a nested if statement where we compare three digits. That will take our original if statement inside our validating if statement. That implies that the program will only be executed once the conditions are meet. We need to call in the function and print each stage as per the requirement.

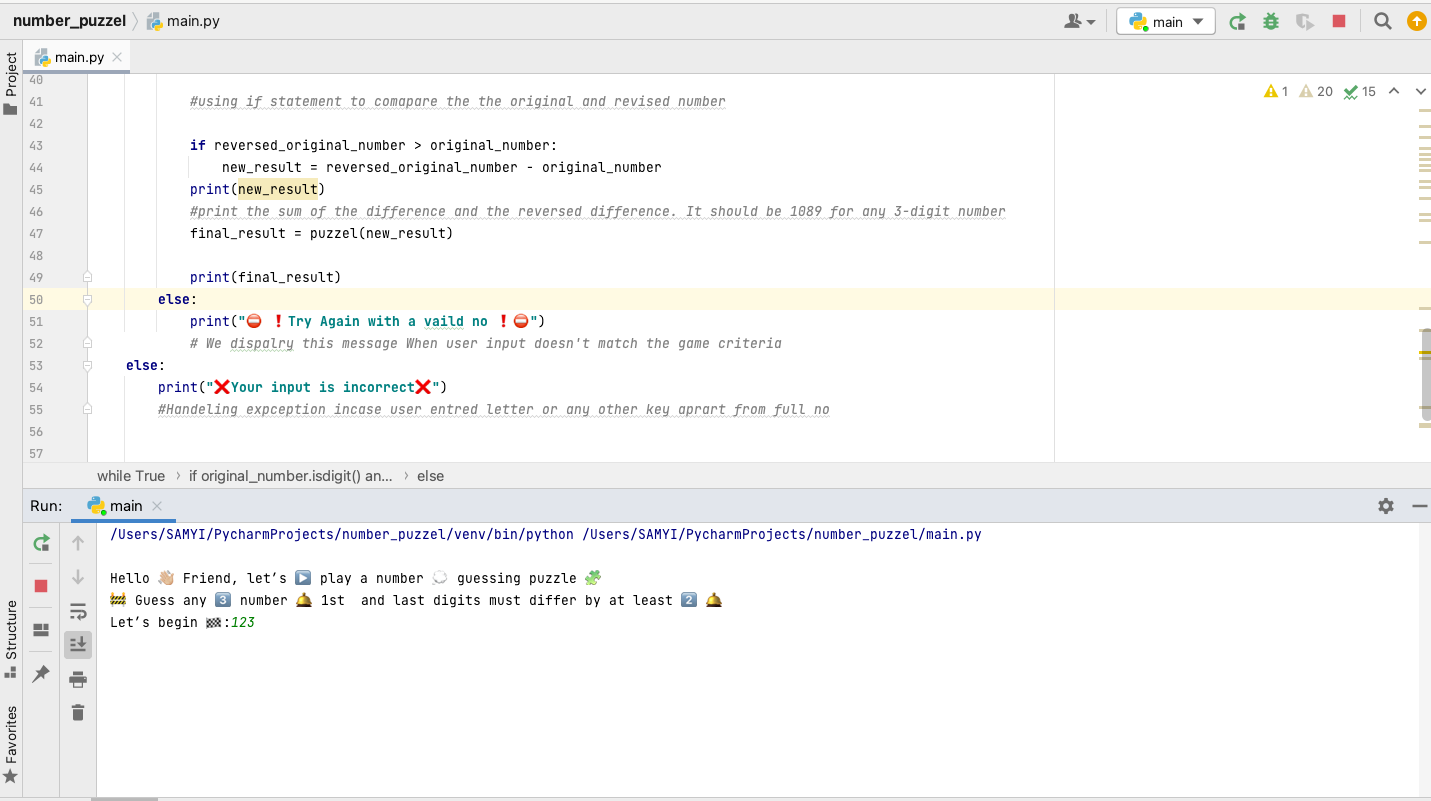
# **Screenshots and testing result:**

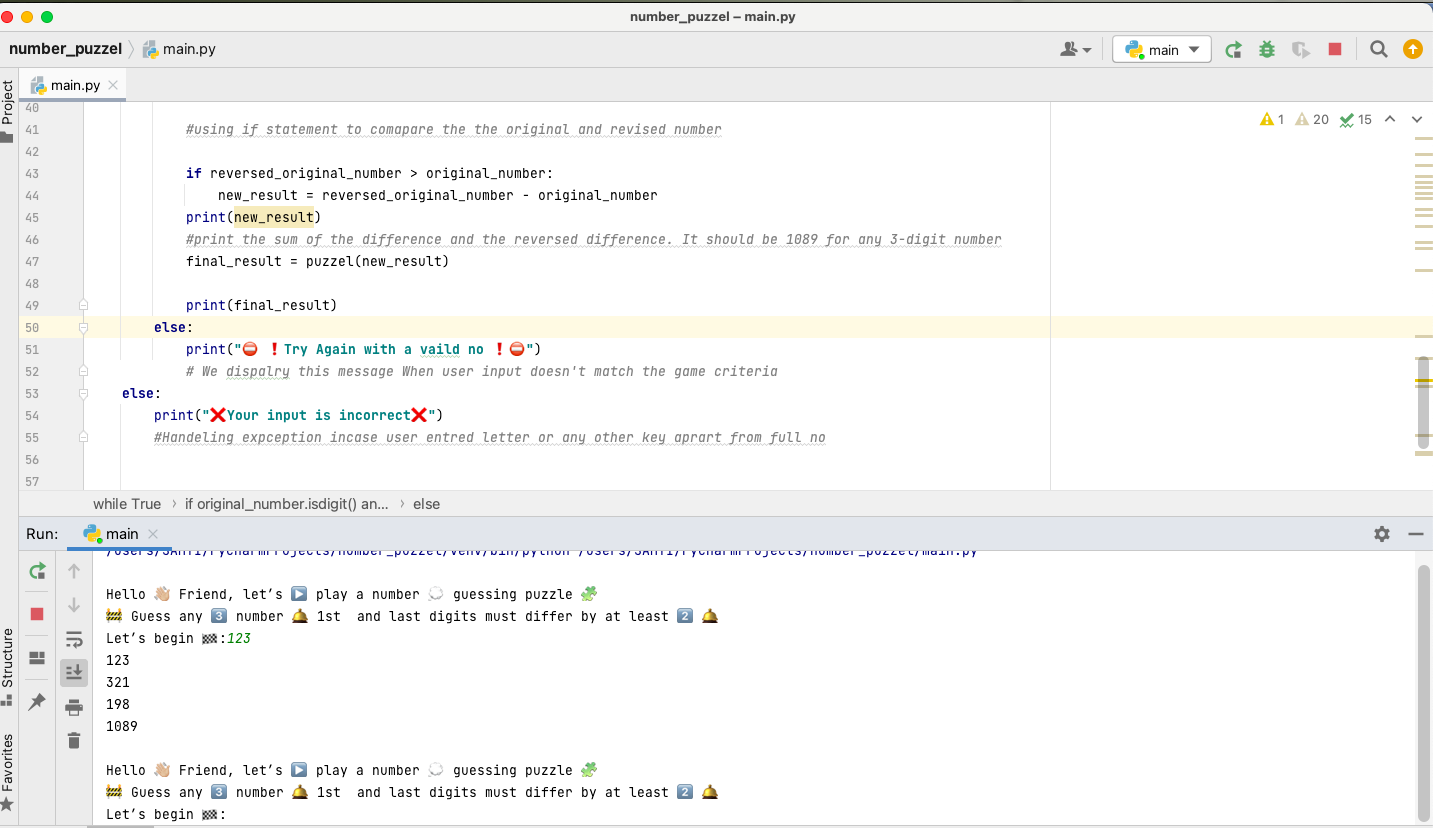
## **3.1 Run the code:**

When executing the code we see the following message in the console:



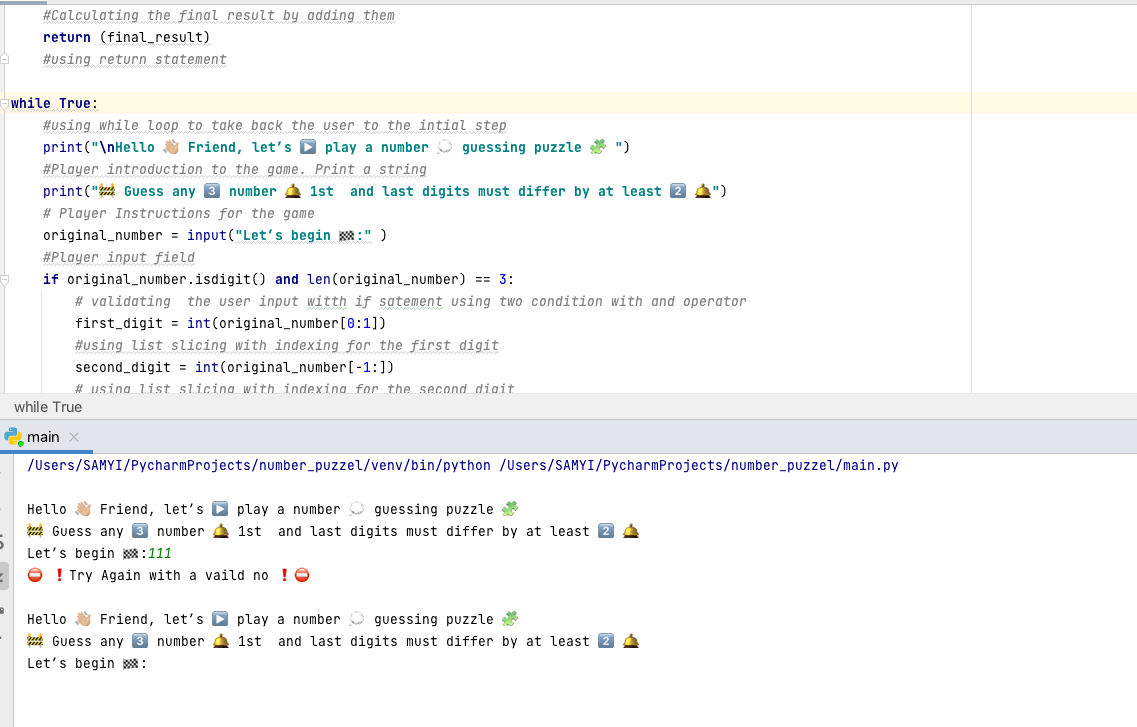
## **3.2 Test with a valid number:**

We start by entering a valid number 123 ;that meet our condition to check if we achieve the result we wanted: 



## **3.3 Test with a 3 digit invalid number:**

Now let's entre an invalid number 111; that does not meet our condition of the difference between 1st and the 3rd number. This time we should get the error message for wrong



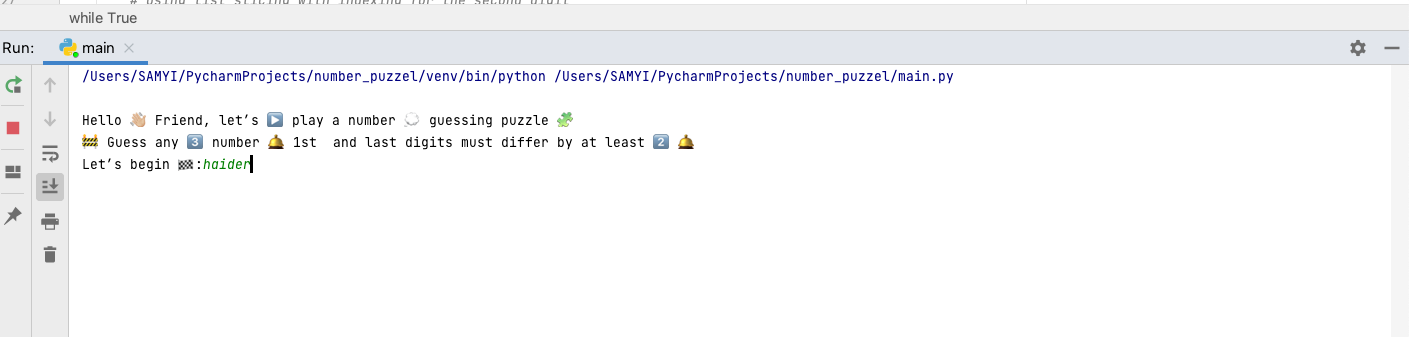


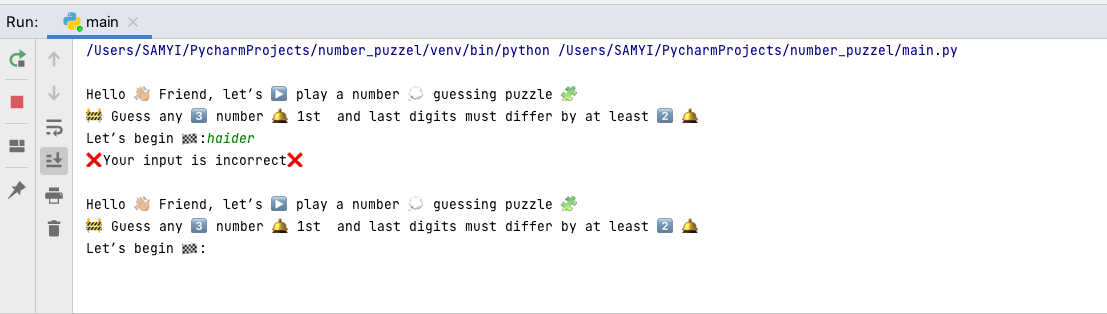
## **3.4 Test with other input:**

Next we test with two digit number 11 and see what happens.



The final test is to check if we type a letter to check if our program returns our custom error message. We type haider instead of the number.





# **References**

1. Schafer, C. (2017). *Python Tutorial for Beginners 4: Lists, Tuples, and Sets*. *YouTube*. Available at: https://www.youtube.com/watch?v=W8KRzm-HUcc [Accessed 9 Jun. 2021].
2. Python.org. (2021). *Data Types — Python 3.9.5 documentation*. [online] Available at: https://docs.python.org/3/library/datatypes.html [Accessed 30 May. 2021].
3. ‌Programiz.com. (2015). *Python List (With Examples)*. [online] Available at: https://www.programiz.com/python-programming/list [Accessed 5 Jun. 2021].
4. Rungta, K. (2020). *Home*. [online] Guru99.com. Available at: https://www.guru99.com/ [Accessed 15 April. 2021].
5. W3schools.com. (2021). *Python String join() Method*. [online] Available at: https://www.w3schools.com/python/ref\_string\_join.asp [Accessed 7 Jun. 2021].
6. Freecodecamp.org. (2021). *Scientific Computing with Python Certification | freeCodeCamp.org*. [online] Available at: https://www.freecodecamp.org/learn/scientific-computing-with-python/ [Accessed 1 Jun. 2021].

‌

‌

‌

# **Appendix**

**def** puzzel (new\_result):  
 *# function created named puzzel* new\_result\_list = list(str(new\_result))  
 *#creating a list and convert it to string* reversed\_new\_result\_list = new\_result\_list[::-1]  
 *#creating a reverse list slicing function used* reversed\_new\_result\_list\_int = int(**''**.join(reversed\_new\_result\_list))  
 *#Converting the reverse list into interger type new variable* final\_result = new\_result + reversed\_new\_result\_list\_int  
 *#Calculating the final result by adding them* **return** (final\_result)  
 *#using return statement***while True**:  
 *#using while loop to take back the user to the intial step* print(**"\nHello 👋🏼 Friend, let’s ▶️ play a number 💭 guessing puzzle 🧩 "**)  
 *#Player introduction to the game. Print a string* print(**"🚧 Guess any 3️⃣ number 🛎 1st and last digits must differ by at least 2️⃣ 🛎"**)  
 *# Player Instructions for the game* original\_number = input(**"Let’s begin 🏁:"** )  
 *#Player input field* **if** original\_number.isdigit() **and** len(original\_number) == 3:  
 *# validating the user input witth if satement using two condition with and operator* first\_digit = int(original\_number[0:1])  
 *#using list slicing with indexing for the first digit* second\_digit = int(original\_number[-1:])  
 *# using list slicing with indexing for the second digit* **if** first\_digit - second\_digit >= 2 **or** second\_digit - first\_digit >= 2:  
 original\_number = int(original\_number)  
 *# Player input field converted to integer data type* print(original\_number)  
 *# Printing the original number that user entred* reversed\_original\_number = int(str(original\_number)[::-1])  
 *# slicing notation used to reverse the stirng first then convert it int again* print(reversed\_original\_number)  
 *#Printing the reverse of the original number* **if** original\_number > reversed\_original\_number:  
 new\_result = original\_number - reversed\_original\_number  
  
 *#using if statement to comapare the the original and revised number* **if** reversed\_original\_number > original\_number:  
 new\_result = reversed\_original\_number - original\_number  
 print(new\_result)  
 *#print the sum of the difference and the reversed difference. It should be 1089 for any 3-digit number* final\_result = puzzel(new\_result)  
  
 print(final\_result)  
 **else**:  
 print(**"⛔️ ❗️Try Again with a vaild no ❗️⛔️"**)  
 *# We dispalry this message When user input doesn't match the game criteria* **else**:  
 print(**"❌Your input is incorrect❌"**)  
 *#Handeling expception incase user entered letter or any other key aprart from full no*