**Python Tutorial**

Set up Python

* Print(“Text”) | Print out the text in “ ”
* Python execute code line by line | Top to bottom

Variables & Data Types

* Variable are good because it can store values (numbers, list, tuple, strings, dictionary, etc.) | If you have long story and you want to change the character name, you have to manually make changes, that’s not programming, but we have variables to store the values, and you just have to make change in one place then everything that contain variable will change.
* Strings | Plain text
* Numbers | Decimal numbers or whole numbers
* Boolean | True or False (Important)

For example:

Code: king = "key" (Variable)  
 practice = "perform" (Variable)  
 print("Thing don't come easy, work hard for it, and " + practice)  
 print("People don't succeed overnight, it takes years of " + practice)  
 print("Work is " + king)

# You can also change variable half way through code   
 king = "Jan" (Change variable)  
 print("Kindness is " + king)  
 print("Truth always win")  
 print("Patient is " + king)

Output: Thing don't come easy, work hard for it, and perform

People don't succeed overnight, it takes years of perform

Work is key

Kindness is Jan

Truth always win

Patient is Jan

Working With Strings

* Strings are plain text
* \n is newline in a string
* concatenate is combine strings (line = "Jan Poonthong" | print(line + " is kind")
* Built-in method used for string handling | .lower(), .upper(), etc.
  + You can also check if text are uppercase | isupper() (True or False)
  + You can use one after the other | (line = "Jan Poonthong" | print(line.upper().isupper())) | True
* Len() | Check how many character are in string
* Variable[0] | Give first letter of string
  + Jan Poontho n g
  + Python character start with 0123456789 -2 -1
* .index(“ ”) | is a method that finds the given element in a list and returns its position
  + If I put “Poon” | It will show number 4 because the letter P start on number 4
  + If you put the character that’s not in a string | Error
* .replace(“ “, “ “) | is a method that can replace text in a string

For example:

Code: print("Jan\nPoonthong")

print("Jan\"Poonthong")

line = "Jan Poonthong"  
 print(line)

line = "Jan Poonthong"  
 print(line + " is kind")

line = "Jan Poonthong"  
 print(line.lower())

line = "Jan Poonthong"  
 print(line.isupper())

# It makes line uppercase first then check it

line = "Jan Poonthong"  
 print(line.upper().isupper())

# Counts how many character are in string

line = "Jan Poonthong"  
 print(len(line))

line = "Jan Poonthong"  
 print(line[0])

line = "Jan Poonthong"  
 print(line.index("J"))

line = "Jan Poonthong"  
 print(line.index("Poon"))

line = "Jan Poonthong"  
 print(line.index("z"))

line = "Jan Poonthong"  
 print(line.replace("Jan", "Baw"))

Output: Jan

Poonthong

Jan"Poonthong

Jan Poonthong

Jan Poonthong is kind

jan poonthong

False

True

J

0

4

Baw Poonthong

Working With Numbers

* print(numbers) | Positive, Negative, Decimals number, whole number, etc.
* print(3 \* (4 + 5)) | You can use parentheses to change mathematical order
* print(10 % 3) | This mean 10 / 3 but % will only give the remainder of the division
  + 10 / 3 = 3 but remainder is 1
* You can also change number to string by print(str(variable))
  + print(str(variable) + “ my favorite number”) | We use this because in some case you want to print some text after the number but you can’t join int with string, you can only join string + string, so that’s the point of converting int to string
* You can get absolute number by | print(abs(variable))
* You can use pow | To power the number 3^2 = 9
* print(max(number, number)) | To get max of number
* print(min(number, number)) | To get min of number
* print(round(number)) | You can also round numbers, 3.5 = 4, 3.4 = 3
* There are many .method but in order to access we need to import
  + from math import \* | Importing
  + print(floor(number)) = Lowest number or cut off decimal point
  + print(ceil(number)) = No matter number 3.7, ceil around lowest 3.7 = 3
  + print(sqrt(number)) = Square root the number | 36 = 6.0

Code: my\_num = 5  
 print(str(my\_num))

my\_num = 5  
 print(str(my\_num) + " my favorite number")

my\_num = -5  
 print(abs(my\_num))

my\_num = -5  
 print(pow(3, 2))

my\_num = -5  
 print(pow(3, 2))

my\_num = -5  
 print(max(3, 2))

print(round(3.7))

print(floor(3.7))

print(ceil(3.7))

print(sqrt(36))

Output: 5

5 my favorite number

5

9

3

4

4

3

6.0

Getting Input From Users

* variable = input(“ “) | You have store what user type in a variable to do next
  + name = input("What is your name: ")  
    print("Hi "+ name + " nice to meet you")

Code: name = input("What is your name: ")  
 print("Hi "+ name + " nice to meet you")

name = input("What is your name: ")  
 age = input("What is your age: ")  
 print("Hi "+ name + " nice to meet you, you are " + age + " years old")

Output: What is your name: Jan # User can type here

Hi Jan nice to meet you

What is your name: Some

What is your age: 71

Hi Some nice to meet you, you are 71 years old

Building a Basic Calculator

* int only allow whole number, if you put decimal number it will give an error
* float allow whole number and decimal number, so user can put any number can the software will not break
  + Code: number\_one = input("Enter a number: ")  
     number\_two = input("Enter a number: ")  
     print(float(number\_two) + float(number\_two))
  + Output: number\_one = input("Enter a number: ")  
     number\_two = input("Enter a number: ")  
     print(float(number\_two) + float(number\_two))

Lists ( Arrays)

* List is a collection which is ordered and changeable. Allows duplicate members.
* A list is a collection which is ordered and changeable. In Python lists are written with square brackets.
* Python start the character by 0123456789
* You can also put string, number, boolean inside of list, Python will just fine
  + list = [“Jim”, 2, True]
  + You can access the list by putting index
  + print(friends[-1] | You can put negative to access back of the list
  + print(friends[1:] | This mean start from character one and all the way till end
  + print(friends[1:3] | This will only grab character one and character two, it will not grab character three
  + You can also update in the middle or change name
    - Code: friends = ["John", "Bob", "Roh"]  
       friends[1] = "Johny"  
       print(friends[1])
    - Ouput: Johny

Code: friends = ["John", "Bob", "Roh"]  
 print(friends[0])

Output: John

List Functions

* The extend() extends the list by adding all items of a list (passed as an argument) to the end.
  + Code: numbers = [1, 4, 5, 76, 23, 25]  
     friends = ["John", "Tom", "Toby", "Saw"]  
     friends.extend(numbers)  
     print(friends)
  + Output: ['John', 'Tom', 'Toby', 'Saw', 1, 4, 5, 76, 23, 25]
* The append() method in python adds a single item to the existing list. but will modify the original list by adding the item to the end of the list.
  + Code: numbers = [1, 4, 5, 76, 23, 25]  
     friends = ["John", "Tom", "Toby", "Saw"]  
     friends.append("Creed")  
     print(friends)
  + Output: ['John', 'Tom', 'Toby', 'Saw', 'Creed']
* The insert() method insert an element to the list at a given index
  + list.insert(index, “element”)
  + It will not replace it but push other element to right hand-side an add it to list
* The remove() method just remove element from list
  + remove(“element”)
* The clear() method clear all element on the list
  + clear()
* The pop() method pop an element of the list (Last element)
  + pop()
* The index(“ “) method give an element position or check element is there in a list or not
  + list.index(“ “)
  + But if you put element that is not in the list, it will give an error
* The count(“ “) method count how many time did the element have in a list
* The sort() method sort the element of a list in specific order (Ascending or Descending)
  + sort(reverse=True) or list.reverse() | reverse to (Descending to Ascending)
* Variable1 = Variable.copy() | The copy() method copes all the element on other variable

Tuples

* A tuple is a collection which is ordered and unchangeable. In Python tuples are written with round brackets
* You can’t change anything in tuple, what you see is what you get
* Use tuples when you want to fix element or you don’t want anything to changeable

Function

* A function is a block of code which only runs when it is called. You can pass data, known as parameters, into a function. A function can return data as a result.
* Arguments | Information can be passed into functions as arguments. Arguments are specified after the function name, inside the parentheses. You can add as many arguments as you want, just separate them with a comma.

Return Statement

* A return statement is used to end the execution of the function call and “returns” the result (value of the expression following the return keyword) to the caller. The statements after the return statements are not executed. If the return statement is without any expression, then the special value None is returned.
* Return will break out of def (functions) | If you want to write more code after return, you will not be able to

If Statement

* **If Statement** is used for decision making. It will run the body of code only when **IF statement** is true. When you want to justify one condition while the other condition is not true, then you use "**if statement**"
  + Code: is\_male = True  
      
     if is\_male:  
     print("You are a male")  
     else:  
     print("You are not a male")
  + Output: You are a male
* Or | At least one of the condition is True
  + Code: is\_male = False  
     is\_tall = False  
      
     if is\_male or is\_tall:  
     print("You are a male or tall or both")  
     else:  
     print("You are neither male nor tall")
  + Output: You are neither male nor tall
* And | Both of the condition must be True to execute the like in if statement
  + Code: is\_male = True  
     is\_tall = True  
      
     if is\_male and is\_tall:  
     print("You are a male tall")  
     else:  
     print("You are either not male or not tall or both")
  + Output: You are a male tall
* Not() | The '**not**' is a Logical operator in **Python** that will return True **if** the expression is False.
  + Code: is\_male = False  
     is\_tall = True  
      
     if is\_male and is\_tall:  
     print("You are a male tall")  
     elif is\_male and not (is\_tall):  
     print("You are a short male")  
     elif is\_tall and not (is\_male):  
     print("You are tall female")  
     else:  
     print("You are either not male or not tall or both")
  + Output: You are tall female

If Statement & Comparisons

* Code: def max\_num(num1, num2, num3):  
   if num1 >= num2 and num1 >= num3:  
   return num1  
   elif num2 >= num1 and num2 >= num3:  
   return num2  
   else:  
   return num3  
    
    
  print(max(3,40,5))
* Output: 40

Dictionaries

* Dictionaries are in {} is fixed and unchangeable, what you see is what you get

While Loop

* While loop is used to execute a block of statement repeatedly until a given condition is satisfied
  + i = 1  
    while i <= 10:  
     print(i)  
     i += 1  
    print("Done with loop")
    - First variable is give I
      * while loop is given to check if condition is true, if it true the while will read code under it
      * print(i) then I = I + 1, again it check the condition can do it again and again, until the condition is False

For Loop

* A for loop is used for iterating over a sequence (that is either a list, a tuple, a dictionary, a set, or a string).
* This is less like the for keyword in other programming languages, and works more like an iterator method as found in other object-orientated programming languages.
* With the for loop we can execute a set of statements, once for each item in a list, tuple, set etc.
* Code: friends = ["Jim", "Jan", "Poon"]  
   for i in range(3):  
   if i == 0:  
   print(friends[i])  
   else:  
   print("Not friends")
* Output: Jim

Not friends

Not friends

Exponent Function

* def raise\_to\_power (base\_num, pow\_num):  
   result = 1  
   for i in range(pow\_num):  
   result = result \* base\_num  
   return result  
    
  print(raise\_to\_power(2, 3))

2D Lists & Nested Loops

* Check in Google or your files name 2D lists

Try & Except

* Help Python to not break the program
* Code: try:

number = int(input(“Enter a number: “))

print(number)

except:

print(“Invalid Input”)

* You can also put except ZeroDivisionError as err:

print(err)

* Have file in system name: try\_except

Reading Files

* open(“filename”, “r”)
* “r” = Read
* “w” = Write
* “a” = Append (Can’t change only add)
* Check file in system