Welcome to Python Fundamentals

Joaquin, Marc Christopher C., ©2021 58015

In this module, we are going to establish or review our skills in Python programming. In this notebook we are going to cover:

- Variables and Data Types
- Operations
- Input and Output Operations
- Logic Control
- Iterables
- Functions

Variable and Data Types

```
x = 1
a,b = 0, -1
```

```
type(x)
```

```
int
```

```
y = 1.0
type(y)
```

```
float
```

By using *type* function, it will identify the data types of the given variable.

```
x = float(x)
type(x)
```

```
float
```

By using *float* function, it wll make an integer variable into float variable.

```
s,t,u = "0", '1', 'one'
type(s)
```

```
str
```

```
s_int = int(s)
s_int
```

```
0
```

By using *int* function, it will show what the given variable is.

Operations

Arithmetic

```
a,b,c,d = 2.0, -0.5, 0, -32
```

```
### Addition
S = a+b
S
```

```
1.5
```

To add variables, *plus* sign can be used to get the sum of it.

```
### Subtraction
D = b-d
D
```

```
31.5
```

To subtract variables, *minus* sign can be used to get the sum of it.

```
### Multiplication
P = a*d
P
```

```
-64.0
```

To multiply variables, *times* sign can be used to get the sum of it.

```
### Division
Q = c/a
Q
```

```
0.0
```

To divide variables, *slash* sign can be used to get the sum of it.To add up, the programmers can use *alpa* to avoid errors.

```
### Floor Division
Fq = a//b
Fq
```

```
-4.0
```

To divide and round it automatically in the nearest integer, double slash sign can be used.

```
### Exponentiation
E = a**b
E
```

```
0.7071067811865476
```

For exponentiation, double asterisk can be used to get the answer.

```
### Modulo
mod = d%a
mod
```

```
0.0
```

To get the remainder of the variable, *percent* sign can be used.

Assignment Operations

```
G, H, J, K = 0, 100, 2, 2
```

```
G = 0 ## in order for not adding the values continuously
a = 2.0
G += a
G
```

```
2.0
```

```
H -= d
H
```

```
196
```

```
J *= 2
J
```

```
8
```

```
K **= 2
K
```

```
16
```

In every operation with equal sign can give you a different answers continuously.

Comparators

```
res_1, res_2, res_3 = 1, 2.0, "1"
true_val = 1.0
```

```
## Equality
res_1 == true_val
```

```
True
```

Using double *equal* sign can help to determine if the two variables are match or literally equal to each other.

```
## Non-equality
res_2 != true_val
```

```
True
```

WIth exclamation point and an *equal* sign can distinguish if the two variables are not equal to each other.

```
## Inequality
t1 = res_1 > res_2
t2 = res_1 < res_2/2
t3 = res_1 >= res_2/2
t4 = res_1 <= res_2
```

```
t4 = res_1 <= res_2
t1
```

```
False
```

With *greater* and *less than* sign or with *equal sign can define if one of the variable are greater, less than, or equal to each other.

Logical

```
res_1 == true_val
```

```
True
```

It is the same as the equality given above under comparators

```
res_1 is true_val
```

```
False
```

is function used to determine if the two given are the same value and data types.

```
res_1 is not true_val
```

```
True
```

is not function is the opposite function of *is* function.

```
p, q = True, False
conj = p and q
conj
```

```
False
```

```
p, q = True, False
disj = p or q
disj
```

```
True
```

```
p, q = True, False
nand = not(p and q)
nand
```

```
True
```

```
p, q = True, False
xor = (not p and q) or (p and not q)
xor
```

```
True
```

conj, *disj*,*nand* and *xor* are can be helpful to determine easily if it's true or false instead of finding it in the truth table. In discrete math it will clearly explain how come all that happened.

I/O

```
print("Hello World")
```

```
Hello World
```

Using *print* statement, it can give you an output which the programmers dictate inside of it.

```
cnt = 1
```

```
string = "Hello World"
print(string, ", Current run count is:", cnt)
cnt += 1
```

```
Hello World , Current run count is: 3
```

for a continuous counting, *cnt* is a must to achieve that output.

```
print(f"{string}, Current count is: {cnt}")
```

```
Hello World, Current count is: 4
```

This is *fstring* which is also the same as *print* statement above.

```
sem_grade = 82.243564657461234
name = ""
print("Hello {}, your semestral grade is: {}".format(name, sem_grade))
```

```
Hello , your semestral grade is: 82.24356465746123
```

```
w_pg, w_mg, w_fg = 0.3, 0.3, 0.4
print("The weights of your semestral grades are:\
\n\t{:.2%} for Prelims\
\n\t{:.2%} for Midterms, and\
\n\t{:.2%} for Finals.".format(w_pg, w_mg, w_fg))
```

```
The weights of your semestral grades are:
        30.00% for Prelims
        30.00% for Midterms, and
        40.00% for Finals.
```

```
x = input("enter a number: ")
x
```

```
enter a number: ada
'ada'
```

In order to let the user input values, use *input* statement.

```
name = input("Kimi no nawa: ")
pg = float(input("Enter prelim grade: "))
mg = float(input("Enter midterm grade: "))
fg = float(input("Enter finals grade: "))
sem_grade = pg*w_pg + mg*w_mg + fg*w_fg
print("Hello {}, your semestral grade is: {}".format(name, sem_grade))
```

```
Kimi no nawa: Kwis
Enter prelim grade: 65
Enter midterm grade: 65
Enter finals grade: 80
Hello Kwis, your semestral grade is: 71.0
```

Looping Statements

While

```
## while loops
i, j = 0, 10
while(i<=j):
    print(f"{i}\t|\t{j}")
    i+=1
```

```
0       |       10
1       |       10
```

```
2       |       10
3       |       10
4       |       10
5       |       10
6       |       10
7       |       10
8       |       10
9       |       10
10      |       10
```

while looping statement is stating a condition what value it will start and when it will end. In this statement, "i" is 0 and "j" is 10, it stated inside the *while* looping statement that it will start from zero up until it will be equal to ten which is "j".

For

```
# for(int i=0; i<10; i++){
# printf(i)
# }

i=0
for i in range(10):
    print(i)
```

```
0
1
2
3
4
5
6
7
8
9
```

```
playlist = []
print('Now Playing:\n')
for song in playlist:
    print(song)
```

```
Now Playing:
```

"For" looping statement, it helps the programmer to print a ton of datas like for an instance, from the value of "i" and the value inside the range is 10, it will print from 0 to 9 because 0 is consider counting as 1.

Flow Control

Condition Statements

```
numeral1, numeral2 = 12, 12
if(numeral1 == numeral2):
    print("Yey")
elif(numeral1>numeral2):
    print("Hoho")
else:
    print("Aww")
print("Hip hip")
```

```
Yey
Hip hip
```

By using *if* and *elif* statement, it shows conditions in which the output will agree on. if the output and condition are match, then it will print the given statement under that condition.

Functions

```
# void DeleteUser(int userid){
#     delete(userid);
# }

def delete_user (userid):
    print("Successfully deleted user: {}".format(userid))

def delete_all_users ():
    print("Successfully deleted all users")
```

```
userid = 0
delete_user(0)
delete_all_users()
```

```
Successfully deleted user: 0
Successfully deleted all users
```

```
def add(addend1, addend2):
    return addend1 + addend2

def power_of_base2(exponent):
```

```
def power_of_base2(exponent):
    return 2**exponent
```

Functions is the compilation of needed functions for a ton of datas

Lambda Functions

```
x = 4
```

```
g = lambda x: 2*(x*x)-1
print(g(x))
```

```
31
```

```
'''
Create a grade calculator that computes for the semestral grade of a course.
Students could type their names, the name of the course, then their prelim,
midterm, and final grade.
The program should print the semestral grade in 2 decimal points and should
display the following emojis depending on the situation:
happy - when grade is greater than 70.00
laughing - when grade is exactly 70.00
sad - when grade is below 70.00
'''
happy, lol, sad = "\U0001F600","\U0001F923","\U0001F619"
```

```
def grade_calculator():
  studentname = input("Student Name: ")
  sutdentcourse = input("Student Course: ")
  pg = float(input("Enter prelim grade: "))
  mg = float(input("Enter midterm grade: "))
  fg = float(input("Enter finals grade: "))
  sem_grade = pg*w_pg + mg*w_mg + fg*w_fg
  if sem_grade == 70:
    print("Hello {}, your semestral grade is: {}.{}".format(studentname, sem_grade, lol))
  elif sem_grade > 70:
    print("Hello {}, your semestral grade is: {}.{}".format(studentname, sem_grade, happy))
  else:
    print("Hello {}, your semestral grade is: {}.{}".format(studentname, sem_grade, sad))
```

```
grade_calculator()
```

```
Student Name: Kwis
Student Course: CpE
Enter prelim grade: 70
```

```
Enter midterm grade: 79
Enter finals grade: 80
Hello Kwis, your semestral grade is: 76.7.😀
```

In this activity, the programmer used *input* statement so the user can fill up the following details. In addition, he used *elif* statement so that once the program determine the final grade, it will look for the perfect match of the conditions given by the programmer.

Github Repo:[chrisjoaquin29]