HW 1: Basic Python Programming

CPE232 Data Models

Description : For this homework, please write your code in the section we prepared for you. If you need to leave a comment or message for the TA, use the # symbol in the code section. You may also create a text cell in the Jupyter Notebook to provide clarification.

Good luck with Python!

PYTHON
PROGRAMMING
A PRACTICAL INTRODUCTION TO PYTHON PROGRAMMING FOR TOTAL BEGINNERS

1. Basic usage

> John Doe is a 35 years-old software engineer who earns ฿35000.00 a month.

Create and assign variables to store this person's information (name, age, position and salary).

In [4]:

```
# Write your code here
name = "John Doe"
age = 35
position = "software engineer"
salary = 35000.00
```

What is the type of each variables?

In [5]:

```
# Write your code here
print(type(name))
print(type(age))
print(type(position))
print(type(salary))
```

```
<class 'str'>
<class 'int'>
<class 'str'>
<class 'float'>
```

The manager decides to give John a 7% raise. Update his salary.

In [6]:

```
# Write your code here
salary = (salary*107)/100
print(salary)
```

```
37450.0
```

Prints his information again with his new salary.

In [7]:

```
print(name)
print(age)
print(position)
print(salary)
```

```
John Doe
35
software engineer
37450.0
```

2. Variable and Expression

2.1 Write a code to convert temperature unit from celcius to other units and then prints out

In [ ]:

```
C = 34.5
```

Fahrenheit

$$\frac{C}{5} = \frac{F-32}{9}$$

In [11]:

```
F = (C*9)/5 + 32
print(F)
```

```
94.1
```

Kelvin

$K = C + 273.15$

In [12]:

```
K = C + 273.15
print(K)
```

```
307.65
```

Rømer

$$Ro = \frac{C \times 21}{40} + 7.5$$

In [13]:

```
Ro = (C*21)/40 + 7.5
print(Ro)
```

```
25.6125
```

2.2 Write code to read the input for weight (kg) and height (cm), then print out the BMI (Body Mass Index).

$$BMI = \frac{kg}{m^2}$$

In [15]:

```
weight = int(input("Weight (kg):"))
height = int(input("Height (cm):"))
BMI = weight/((height/100)**2)
# Write your code here
print(BMI)
```

```
19.883853048857468
```

3. Multi-item variables

List

In [16]:

```
names = ['Thomas', 'Kate', 'Mike', 'Amelia', 'James', 'Megan']
```

Create new variable call `new_name` which takes input name of the user.

In [18]:

```
new_name = input('Enter your name: ')
print(new_name)
```

```
thanaboon
```

Insert `new_name` into `names` list.

In [19]:

```
names.append(new_name)
```

Print your name from the list `by index`.

In [22]:

```
print(names.index("thanaboon"))
print(names[6])
```

```
6
thanaboon
```

Merge `another_names` into `names`.

In [24]:

```
another_names = ['Peter', 'Steve', 'Sam', 'Charlotte']
```

In [25]:

```python
names.extend(another_names)
print(names)
```

```
['Thomas', 'Kate', 'Mike', 'Amelia', 'James', 'Megan', 'thanaboon', 'Peter', 'Stev
e', 'Sam', 'Charlotte']
```

Change `Amelia` 's name to `Amy`

In [27]:

```python
names[3] = "Amy"
print(names)
```

```
['Thomas', 'Kate', 'Mike', 'Amy', 'James', 'Megan', 'thanaboon', 'Peter', 'Steve',
'Sam', 'Charlotte']
```

Dictionary

In [29]:

```python
capital_city = {'England':'London',
                'Spain':'Madrid',
                'Japan':'Tokyo',
                'Australia':'Sydney',
                'Germany':'Berlin',
               }
```

Add a record `Thailand` and it's capital city to this dictionary

In [30]:

```python
capital_city['Thailand'] = 'Bangkok'
print(capital_city)
```

```
{'England': 'London', 'Spain': 'Madrid', 'Japan': 'Tokyo', 'Australia': 'Sydney', 'G
ermany': 'Berlin', 'Thailand': 'Bangkok'}
```

You may notice that the capital city of `Australia` is wrong. It should be `Canberra` . Correct this mistake.

In [31]:

```python
capital_city['Australia'] = 'Canberra'
print(capital_city)
```

```
{'England': 'London', 'Spain': 'Madrid', 'Japan': 'Tokyo', 'Australia': 'Canberra',
'Germany': 'Berlin', 'Thailand': 'Bangkok'}
```

4. Control Flows and conditional statements

if...elif...else

1. Write a program that takes two numbers as input from the user, called A and B. Then, print the number that is greater.

Example:

- If `A = 25` and `B = 15` , the program should print: "A = 25"
- If `A = 10` and `B = 22` , the program should print: "B = 22"

In [33]:

```
A = int(input("A: "))
B = int(input("B: "))

if A > B :
    print(f"A = {A}")
else :
    print(f"B = {B}")
```

```
A = 25
```

2. Define a variable to get input age from user

In [34]:

```
age = int(input("Input your age: "))
print(age)
```

```
19
```

Write a series of if...elif...else statement that categorize input age into following groups:

> Babies: 0-2 years old
>
> Children: 3-12 years old
>
> Teenager: 13-19 years old
>
> Young Adults: 20-29 years old
>
> Middle-aged Adults: 30-45 years old
>
> Old Adult: 46-59 years old
>
> Elderly: Above 60 years old

In [35]:

```
if  age >= 60 :
    print('Elderly')
elif age > 45 :
    print('Old Adult')
elif age > 29 :
    print('Middle-aged Adults')
elif age > 19 :
    print('Young Adults')
elif age > 12 :
    print('Teenager')
elif age > 2 :
    print('Children')
else :
    print('Babies')
```

```
Teenager
```

Looping

1. Write a code to create a multiplication table of an input number (multiplier from 1-12).

In [36]:

```
n = int(input("input number: "))
for i in range(1,13) :
    print(f"{n} * {i} = {n*i}")
```

```
3 * 1 = 3
3 * 2 = 6
3 * 3 = 9
3 * 4 = 12
3 * 5 = 15
3 * 6 = 18
3 * 7 = 21
3 * 8 = 24
3 * 9 = 27
3 * 10 = 30
3 * 11 = 33
3 * 12 = 36
```

2. Write a code that construct the following pattern.

```
input: 5
output:
*
**
***
****
*****
```

In [38]:

```
m = int(input("Input Triange size: "))
for i in range(1,m+1) :
    print("*"*i)
```

```
*
**
***
****
*****
```

3. Creates a loop to print `I love <programming language>!` except for Assembly, print `Not you, Assembly`.

In [39]:

```
languages = ['C/C++', 'Python', 'R', 'Java', 'SQLs', 'Assembly', 'Go', 'Rust', 'Kot
```

In [40]:

```
for i in languages :
    if i != "Assembly" :
        print(f"I love {i}!")
    else :
        print("Not you, Assembly.")
```

```
I love C/C++!
I love Python!
I love R!
I love Java!
I love SQLs!
Not you, Assembly.
I love Go!
I love Rust!
I love Kotlin!
```

4. Write a code to print every number from 1 to 25 except the one that is divisible by 3.

In [41]:
```
for i in range(1,26) :
    if i % 3 != 0 :
        print(i)
```

```
1
2
4
5
7
8
10
11
13
14
16
17
19
20
22
23
25
```

5. Write a code that finds the number that is divisible by 7 in a given range.

In [42]:
```
lower_bound = 1
upper_bound = 100
divisor = 7

result = []
```

In [43]:
```
for i in range(lower_bound,upper_bound+1) :
    if i % divisor == 0 :
        result.append(i)
print(result)
```

```
[7, 14, 21, 28, 35, 42, 49, 56, 63, 70, 77, 84, 91, 98]
```

6. Write a code that construct the following pattern.

```
input: 5
output:
*#####
**####
***###
****##
*****#

input: 10
output:
*##########
**#########
```

```
    ***########
    ****#######
    *****######
    ******#####
    *******####
    ********###
    *********##
    **********#
```

In [45]:

```
k = int(input("Input size"))
for i in range (1,k+1) :
    print("*" * i + "#" * (k+1-i))
```

```
*#####
**####
***###
****##
*****#
```

5. Functions

1. Define a function `average` that takes *list of numbers* and calculate the mean of input. It should look like this:

> average([1,2,3,4]) output: 2.5

In [51]:

```
# Write your code here
def average(list):
    s = len(list)
    sum = 0
    for i in list :
        sum += i
    return sum/s

avg =average([1,2,3,4])
print(avg)
```

```
2.5
```

2. Define a function `sumproduct` that takes 2 *equal-sized* lists and calculate sum of the products of two lists.
It should look like this:

> sumproduct([1,2,3],[4,5,6])
> output: 32

(1 * 4) + (2 * 5) + (3 * 6) = 32

In [52]:

```
def sumproduct(l1,l2):
    s  = len(l1)
    sum = 0
    for i in range(0,s):
        sum+= l1[i]*l2[i]
    return sum

sp = sumproduct([1,2,3],[4,5,6])
print(sp)
```

```
32
```

3. Define a function `fibonacci` that returns Fibonacci number at `n` position.
A Fibonacci number at position `n` is defined by `F(n) = F(n-1) + F(n-2)`. Where `F(0) = 0` and `F(1) = 1`
Example: `0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, ...`

In [3]:

```
def fibonacci(n) :
    fib = [0,1]
    for i in range(2,n) :
        fib.append(fib[i-1]+fib[i-2])
    return fib[n-1]

n = 7
fib = fibonacci(n)
print(fib)
```

```
8
```

4. Define a function `is_palindrome` that takes input string and check whether it is a palindrome or not.
A string is a palindrome if it reads the same forward and backwards.

Example: `madam`, `race car`, `borrow or rob`, `amore roma`, `never odd or even`
Do not consider whitespace. Use `str.replace(' ', '')` to remove whitespace from your string.
Case-insensitive. You can turn everything into lower or uppercase using `str.lower()` or `str.upper()`

Hint: you can reverse the string using `[::-1]` slice.

In [ ]:

```
str1 = "radar" # palindrome
str2 = "rotator" # palindrome
str3 = "lemon" # not palindrome
```

In [11]:

```
def is_palindrome(str):
    str_new = str.replace(' ','').lower()
    str_rev = str_new[::-1]
    l = len(str_new)
    is_pal = True
    for i in range(0,l//2) :
        if str_new[i] != str_rev[i]:
            is_pal = False
            break
    return is_pal
```

```
str_1 = "abc"
print(is_palindrome(str_1))

str_2 = "madam"
print(is_palindrome(str_2))

str_3 = "abc CBa"
print(is_palindrome(str_3))
```

```
False
True
True
```

5. An `anagram` is a word or phrase formed by rearranging the letters of a different word or phrase.
Define a function `is_anagram` that takes in 2 strings and check whether it is possible to compose a second string using letters in the first string or not.

Example: `Tom Marrvolo Riddle` can be rearraged into `I am Lord Voldermort`
`Meaning of Life` can be rearranged into `Engine of a Film`
Do not consider whitespace. Use `str.replace(' ', '')` to remove whitespace from your string.
Case-insensitive. You can turn everything into lower or uppercase using `str.lower()` or `str.upper()`
Returns only `True` of `False`

In [14]:

```
def is_anagram(str_1,str_2) :
    str1_new = str_1.replace(' ','').lower()
    str2_new = str_2.replace(' ','').lower()
    dict1 = {}
    dict2 = {}

    for c in str1_new :
        dict1[c] = dict1.get(c,0) + 1

    for c in str2_new :
        dict2[c] = dict2.get(c,0) + 1

    if dict1 == dict2 :
        return True
    else :
        return False

str1 = "Meaning of Life"
str2 = "Engine of a Film"

print(is_anagram(str1,str2))

str3 = "Tom Marrvolo Riddle"
str4 = "I am Lord Voldermort"
print(is_anagram(str3,str4))

print(is_anagram(str1,str3))
```

```
True
True
False
```