hw1-suchanat-ratanarueangrong

January 16, 2024

1 Lab 1: Basic Python Programming

1.1 1. Basic usage

John Doe is a 29 years-old system engineer who earns ฿41500.00 a month.

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

[ ]:

```python
# Write your code here
info = ("John Doe", 29, "system engineer", 41500.00)
# name = "John Doe"
# age = 29
# position = "system engineer"
# salary = 41500.00
```

What is the type of each variables?

[ ]:

```python
# Write your code here
print(type(info[0]))
print(type(info[1]))
print(type(info[2]))
print(type(info[3]))
```

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

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

[ ]:

```python
# Write your code here
salary = info[3]*1.07
```

Prints his information again with his new salary.

[ ]:

```python
# Write your code here
print(f"{info[0]} {info[1]} {info[2]} {salary}")
```

```
John Doe 29 system engineer 44405.0
```

Now, he decides to resign. Delete his information from the system.

[ ]:

```
# Write your code here
temp = list(info)
temp.clear()
test_tup = tuple(temp)
# del(salary)
# print result
print("The tuple after clearing values : " + str(test_tup))
```

```
The tuple after clearing values : ()
```

1.2 2. Variable and Expression

2.1 Write a code to convert temperature unit from celcius to other units

[ ]:

```
C = 34.5
```

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

[ ]:

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

[ ]:

```
94.1
```

Kelvin
$K = C + 273.15$

[ ]:

```
K = C+273.15
K
```

[ ]:

```
307.65
```

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

[ ]:

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

[ ]:

```
25.6125
```

1.3 3. Multi-item variables

List

[ ]:

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

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

[ ]:

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

```
Enter your name: nut
```

Insert `new_name` into `names` list.

[ ]:
```
# Write your code here
names.append(new_name)
```

Select your name from the list

[ ]:
```
names
```

[ ]:
```
['Thomas', 'Kate', 'Mike', 'Amelia', 'James', 'Megan', 'nut']
```

[ ]:
```
# Write your code here
for i in names:
  if i == new_name:
    print(f"{i}")
```

```
nut
```

Merge `another_names` into `names`.

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

[ ]:
```
# Write your code here
for i in another_names:
  names.append(i)
names
```

[ ]:
```
['Thomas',
 'Kate',
 'Mike',
 'Amelia',
 'James',
 'Megan',
 'nut',
 'Peter',
 'Steve',
 'Sam',
 'Charlotte']
```

Change `Amelia`'s name to `Amy`

[ ]:
```
# Write your code here
names[3] = "Amy"
names
```

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

Dictionary

[ ]:

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

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

[ ]:

```
# Write your code here
capital_city["Thailand"] = "Bangkok"
capital_city
```

[ ]:

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

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

[ ]:

```
# Write your code here
capital_city.update({"Australia": "Canberra"})
capital_city
```

[ ]:

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

1.4 4. Control Flows and conditional statements

1.4.1 if…elif…else

1. Define a variable to get input age from user.

[ ]:

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

```
Enter your age: 70
```

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

[ ]:

```
# Write your code here
if age>=0 and age<=2:
  print("Babies")
elif age>=3 and age<=12:
  print("Children")
elif age>=13 and age<=19:
  print("Teenager")
elif age>=20 and age<=29:
  print("Young Adults")
elif age>=30 and age<=45:
  print("Middle-aged Adults")
elif age>=46 and age<=59:
  print("Old Adult")
else:print("Elderly")
```

```
Elderly
```

1.4.2 Looping

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

[ ]:

```
# Write your code here
multiplier = int(input("Type a multiplier: "))
for i in range(1,13):
  print(f"{multiplier} * {i} = {multiplier*i}")
```

```
Type a multiplier: 7
7 * 1 = 7
7 * 2 = 14
7 * 3 = 21
7 * 4 = 28
```

```
7 * 5 = 35
7 * 6 = 42
7 * 7 = 49
7 * 8 = 56
7 * 9 = 63
7 * 10 = 70
7 * 11 = 77
7 * 12 = 84
```

2. Write a code that construct the following pattern.

---

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

[ ]:
```
# Write your code here
step = int(input("The number: "))
for i in range(0, step+1):
  for j in range(0, i):
    print("*", end='')
  print()
```

```
The number: 5

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

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

[ ]:
```
languages = ['C/C++', 'Python', 'R', 'Java', 'SQLs', 'Assembly', 'Go', 'Rust',␣
  ↪'Kotlin']
```

[41]:
```
# Write your code here
for i in languages:
  if(i != "Assembly"):
    print(f"I love <{i}>!")
  else:print(f"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.

[ ]:

```
# Write your code here
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.

[ ]:

```
lower_bound = 1
upper_bound = 100
divisor = 7

result = []
```

[ ]:

```
# Write your code here
for i in range(lower_bound, upper_bound):
  if i % 7 == 0:
    print(i)
```

```
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:
*##########
**#########
***########
****#######
*****######
******#####
*******####
********###
*********##
**********#
```

```
  File "<ipython-input-30-6183328cb379>", line 2
    output:
       ^
SyntaxError: invalid syntax
```

[ ]:

```
# Write your code here
test = int(input("Enter a number: "))
for i in range(1, test+1):
  for j in range(0, i):
    print("*", end=' ')
  for x in range(0, test-j):
    print("#", end=' ')
  print()
```

```
Enter a number: 10
* # # # # # # # # # #
* * # # # # # # # # #
* * * # # # # # # # #
* * * * # # # # # # #
* * * * * # # # # # #
* * * * * * # # # # #
* * * * * * * # # # #
* * * * * * * * # # #
* * * * * * * * * # #
* * * * * * * * * * #
```

1.5 5. Functions

1. Define a function `average` that takes arbitrary number of arguments and calculate the mean of input.

[ ]:

```python
# Write your code here
def average(numbers):
  temp = 0
  for num in numbers:
    temp = temp + num
  return temp/len(numbers)

print(average([2, 3, 4]))
```

```
3.0
```

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

[ ]:

```python
# Write your code here
def sumproduct(num1, num2):
  temp = 0
  for i in range(0, len(num1)):
    temp = num1[i] * num2[i] + temp
  return temp
print(sumproduct([1,2,3],[4,5,6]))
```

```
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, ...`

[ ]:

```python
g = int(input("Enter a number: "))

def fibonacci(x):
    if x < 0:
        print("Incorrect input")
    elif x == 0:
        return 0
    elif x == 1:
        return 1
    else:
        return fibonacci(x-1) + fibonacci(x-2)

print(fibonacci(g))
```

```
Enter a number: 9
34
```

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.

[ ]:

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

[ ]:

```
# Write your code here
str = input("Enter a word: ")
def is_palindrome(str):
  str.replace(' ', '')
  if str.lower() == str[::-1].lower():
    return True
  else: return False
print(is_palindrome(str))
```

```
Enter a word: radar
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`

[ ]:

```python
# Write your code here
str1 = "Meaning of Life"
str2 = "Engine of a Film"
def is_anagram(str1, str2):
    str1 = str1.replace(' ', '').lower()
    str2 = str2.replace(' ', '').lower()
    if sorted(str1) == sorted(str2):
        return True
    else:
        return False

print(is_anagram(str1, str2))
```

```
True
```
