02_Assignment_Python_LeoniePeters

December 7, 2022

1 Assignment 02

Python Basics II This tutorial was written by Terry Ruas (University of Göttingen); The references for external contributors for which this material was anyhow adapted/inspired are in the Acknowledgments section (end of the document).

This notebook will cover the following tasks:

1. String Length
2. Largest List Element
3. Character Frequency
4. Sorted List of Tuples
5. Check Brackets
6. Check Brackets II
7. Queue
8. Unlimited Power
9. Unlimited Power II

1.1 Task 01 – String Length

Write a program that reads in a string and prints the length of the input string. Do not use any built-in functions of Python, such as len(). For example, if the input is "Computer Science", the output should be length: 16.

[4]:
```
str=input("Please enter a word in here:")
counter=0
for i in str:
    counter=counter+1
print(counter)
```

```
Please enter a word in here: Hubschrauberlandeplatz
```

```
22
```

1.2 Task 02 – Largest List Element

Write a program that generates a list of 10 random integers between 1 and 100 and then finds and prints the largest element in the list. Do not use the built-in function max(). For example, if the input is [23,3,42,29,12,15,8,4,37,34], the output should be the largest element: 42.

Hint: Check out the module random.

[7]:
```
l=[]
#enter the lenght of list
n= int(input("Please enter a lenght of list:"))
for i in range(1,n+1):
    e=int(input("Enter Element:"))
    l.append(e)
print(l)
l.sort()
print("The largest Number is:",l[n-1])
```

```
Please enter a lenght of list: 3
Enter Element: 5
Enter Element: 7
Enter Element: 24
```

```
[5, 7, 24]
The largest Number is: 24
```

1.3 Task 03 – Character Frequency

Write a program that: * Reads in a string and removes any spaces from the string * Counts how often individual characters occur in the string * Stores the information on character occurrence frequency in a dictionary * Prints the dictionary.

For example, if the input is “santa claus”, the output should be: {‘s’: 2, ‘a’: 3, ‘n’: 1, ‘t’: 1, ‘c’: 1, ‘l’: 1, ‘u’: 1}.

[1]:
```
words=input("PLease choose a word")
symbols="".join(words)

print(symbols)

dct={}

for i in symbols:
    if i in dct:
        dct[i]+=1
    else:
        dct[i]=1

print(dct)
```

```
PLease choose a word Hubschrauberlandeplatz
```

```
Hubschrauberlandeplatz
{'H': 1, 'u': 2, 'b': 2, 's': 1, 'c': 1, 'h': 1, 'r': 2, 'a': 3, 'e': 2, 'l': 2,
'n': 1, 'd': 1, 'p': 1, 't': 1, 'z': 1}
```

1.4 Task 04 – Sorted List of Tuples

Write a program that: * Generates a list of 10 tuples, each tuple consisting of 3 random integers between 1 and 100 * Sorts the list of tuples in increasing order of the third element in each tuple * Prints the sorted list of tuples

For example, if the generated input list is: [(56, 77, 69), (43, 30, 38), (2, 77, 101), (93, 57, 4), (74, 21, 77), (39, 68, 68), (65, 53, 96), (16, 29, 88), (88, 70, 38)] The output should be: [(93, 57, 4), (43, 30, 38), (88, 70, 38), (39, 68, 68), (56, 77, 69), (74, 21, 77), (16, 29, 88), (65, 53, 96), (2, 77, 101)]

Hint: You are allowed and encouraged to use built-in functions, such as sorted(), for this task.

[20]:
```
number_lists=[
   ␣
  ↪(98,63,22),(21,9,44),(99,52,3),(12,1,64),(13,20,16),(53,37,22),(78,90,23),(14,48,70),(21,10
print("My list of numbers is:", number_lists)
number_lists.sort(key=lambda item:item[2])
print("Sortet list of numbers:", number_lists)
```

```
My list of numbers is: [(98, 63, 22), (21, 9, 44), (99, 52, 3), (12, 1, 64),
(13, 20, 16), (53, 37, 22), (78, 90, 23), (14, 48, 70), (21, 100, 89), (54, 71,
97)]
Sortet list of numbers: [(99, 52, 3), (13, 20, 16), (98, 63, 22), (53, 37, 22),
(78, 90, 23), (21, 9, 44), (12, 1, 64), (14, 48, 70), (21, 100, 89), (54, 71,
97)]
```

1.5 Task 05 – Check Brackets

Write a program that reads in a string, which is supposed to be a mathematical expression. Focus on brackets only and check whether left and right brackets are composed correctly. Ignore all other characters (i.e. you don't have to check correctness of operators and operands). Examples of correct input: * 3(2+5) (((())())) * (3+)(((4))) * Empty string

Examples of incorrect input: * (3(2+5) (((())())(()) * (3+)((4))) * ())(()

[7]:
```
x= input("Tippen Sie hier eine mathematische Formel in Form eines Strings ein")

count=0
aus=True
for a in x:
    if a == "(":
        count+= 1
    elif a == ")":
        count -= 1

    if count <0:
        aus = False

if count != 0:
    aus= False
```

```
if aus:
    print("Korrekte Mathematische Formel")
else:
    print("Inkorrekte Mathematische Formel")
```

```
Tippen Sie hier eine mathematische Formel in Form eines Strings ein 4*(11+8)

Korrekte Mathematische Formel
```

1.6 Task 06 – Check Brackets II

Extend previous program, so it can handle also square and curly brackets. Note that expressions in brackets cannot overlap. So, expression `{[()()]([[]])}{}` is correct, but expression `([)]` is not.

[17]:
```
input("Enter a few brackets")

string= input()
boo=False
for a in string:
    if a not in "([{}])":
        string = string.replace(a,"")
        string_2 = ""

while string_2 != string:
    string_2 =string
    string=string.replace("()","")
    string=string.replace("[]","")
    string=string.replace("{}","")

if string== "":
    print("Correct input")
else:
    print("Incorrect input")
```

```
Enter a few brackets ([{}])
```

```
---------------------------------------------------------------------------
NameError                                 Traceback (most recent call last)
Cell In [17], line 10
      7         string = string.replace(a,"")
      8         string_2 = string.replace(a,"")
---> 10 while string_2 != string:
     11     string_2 =string
     12     string=string.replace("()","")
```

```
NameError: name 'string_2' is not defined
```

1.7 Task 07 – Queue

Write a program that simulates a queue. It will read strings from the input. Consider these inputs as names of people coming to the end of a queue. Whenever “next” is given as input, the program will print out the name on the turn. The program finishes as soon as the queue is empty.

[3]:
```
print("Who stands in the queue?")
print("Hit enter after every name. r to remove, q to quit")

queue=[]

while True:
    data=input()
    if str.lower(data)=="q":
        break
    elif str.lower(data)=="next":
        print("Removing:", queue.pop(0))
    else:
        queue.append(data)
    print(queue)

for Names  in queue:
    print("So this Person came first:", Names)
```

```
Who stands in the queue?
Hit enter after every name. r to remove, q to quit

 Leonie

['Leonie']

 Maya

['Leonie', 'Maya']

 next

Removing: Leonie
['Maya']

 Leonie

['Maya', 'Leonie']

 q

So this Person came first: Maya
So this Person came first: Leonie
```

1.8 Task 08 – Unlimited Power

Write a function with two arguments – x and n. The function returns the value of x^n. Use recursion.

[18]:
```
x= int(input("x: "))
n= int(input("n: "))

result = x**n

print(f"x raised raised ti power of n is: {x} ^ {n} = {result}")
```

```
x:  9
n:  5

x raised raised ti power of n is: 9 ^ 5 = 59049
```

1.9 Task 09 – Unlimited Power II

Using function for factorial and function x^n from previous task, write a program that reads value of x and prints approximate value of e^x. Use this formula (Taylor series) for calculation

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + ... + \frac{x^n}{n!}$$

To get precise value of e^x, the series would have to be infinite. Suppose that there is some required accuracy, so the calculation finishes as soon as the value of the next element is smaller than given threshold (e.g., 0.000001).

[2]:
```
x= int(input("Schreibe eine Zahl für x: "))

def Wert(x, y):
    if y == 0:
        return 1
    else:
        return x * Wert(x, y-1)

def Funktion(x):
    if x == 0:
        return 1
    else:
        return x + Funktion(x-1)

def Mathe(x):
    n = 0
    Ergebniss = 0
    while Wert(x, n) / Funktion(n) > 0.0001:
        Ergebniss += Wert(x, n) / Funktion(n)
        n += 1
    return Ergebniss
print(f"e^{x} = {Mathe(x)}")
```

```
Schreibe eine Zahl für x:  3
```

```
---------------------------------------------------------------------------
OverflowError                             Traceback (most recent call last)
Cell In [2], line 22
     20         n += 1
     21     return Ergebniss
---> 22 print(f"e^{x} = {Mathe(x)}")

Cell In [2], line 18, in Mathe(x)
     16 n = 0
     17 Ergebniss = 0
---> 18 while Wert(x, n) / Funktion(n) > 0.0001:
     19     Ergebniss += Wert(x, n) / Funktion(n)
     20     n += 1

OverflowError: integer division result too large for a float
```

1.10 Acknowledgements

Redmond, Hsu, Saini, Gupta, Ramsey, Kondrich, Capoor, Cohen, Borus, Kincaid, Malik, and many others. - Stanford CS41

```
[ ]:
```