

Assignment

March 18, 2019

1 Python Assignment 1

1.1 Average of best 2 marks

Design a Python program to find the average of best two marks out of three marks taken as input.

```
In [2]: print("Enter 3 numbers")
        a = int(input())
        b = int(input())
        c = int(input())
        avg = max((a + b) / 2, (a + c) / 2, (b + c) / 2)
        print(avg)
```

Enter 3 numbers

4

7

3

5.5

1.2 Student details

Write a program to read the following input from user and display. 1. Name 2. USN 3. Roll No 4. Mobile No 5. E-Mail id 6. Percentage of Marks

```
In [32]: n = int(input("Enter number of students: "))
        for i in range(0, n):
            name = input("Enter the name: ")
            usn = input("Enter the usn: ")
            roll_no = input("Enter the roll number: ")
            mobile = input("Enter the mobile number: ")
            email = input("Enter the email id: ")
            marks = input("Enter the percentage: ")
        print("The details are")
        print("Name\tUSN\tRoll No\tMobile No\tEmail id\tPercentage")
        for i in range(0, n):
            print(name + "\t" + usn + "\t" + roll_no + "\t" + mobile + "\t" + email +
                  "\t" + marks)
```

```

enter number of students: 2
enter the name: Akshay
enter the usn: 01
enter the roll number: 01
enter the mobile number: 1234567890
enter the email id: akmc
enter the percentage: 76
enter the name: dl
enter the usn: sdpcmsp
enter the roll number: sdck
enter the mobile number: s;dcm
enter the email id: ;sc
enter the percentage: ;sc

```

The details are

Name	USN	Roll No	Mobile No	Email id1	Percentage
dl	sdpcmsp	sdck	s;dcm	;sc	;sc
dl	sdpcmsp	sdck	s;dcm	;sc	;sc

1.3 Well bracketed

A string with parentheses is well bracketed if all parentheses are matched: every opening bracket has a matching closing bracket and vice versa. Write a Python function wellbracketed(s) that takes a string s containing parentheses and returns True if s is well bracketed and False otherwise. Here are some examples to show how your function should work.

```

>>>wellbracketed("22")
False
>>>wellbracketed("(a+b)(a-b)")
True
>>>wellbracketed("(a(b+c)-d)((e+f)")
False

```

```

In [38]: s = input("Enter a expression: ")
         c = 0
         for i in range(0, len(s)):
             if s[i] == '(':
                 c = c + 1
             if s[i] == ')':
                 c = c - 1
         if c == 0:
             print("True")
         else:
             print("False")

```

```

Enter a expression: (
False

```

1.4 Sum of Squares

A positive integer m is a sum of squares if it can be written as $k + x$ where $k > 0$, $x > 0$ and both k and x are perfect squares. Write a Python function `sumofsquares(m)` that takes an integer m returns True if m is a sum of squares and False otherwise. Here are some examples to show how your function should work.

```
>>> sumofsquares(41)
True
>>> sumofsquares(30)
False
>>> sumofsquares(17)
True
```

```
In [39]: import math
```

```
def sum_of_squares(n):
    a = int(math.sqrt(n))
    for i in range(0, a + 1):
        for j in range(0, a + 1):
            if (i**2) + (j**2) == n:
                return True
    return False
```

```
n = int(input("Enter a number: "))
print(sum_of_squares(n))
```

```
Enter a number: 41
True
```

1.5 Compute grade

Write a program to prompt for a score between 0.0 and 1.0. If the score is out of range, print an error message. If the score is between 0.0 and 1.0, print a grade using the following table. {Note: Create a function called `computegrade` that takes a score as its parameter and returns a grade as a string.}

```
In [7]: def compute_grade(grade):
        if grade > 0.8:
            return 'A'
        elif grade > 0.7:
            return 'B'
        elif grade > 0.6:
            return 'C'
        elif grade > 0.5:
            return 'D'
```

```

elif grade > 0.4:
    return 'E'
else:
    return 'F'

try:
    grade = float(input("Enter a grade:"))
    if 0.0 <= grade <= 1.0:
        print(compute_grade(grade))
    else:
        print("Invalid entry")
except ValueError:
    print("Not a valid type")

```

```

Enter a grade:wed
Not a valid type

```

1.6 Reverse a string

Write a program to reverse a string.

```

In [42]: s = input("Enter string:")
         print("Reversed string: " + s[::-1])

```

```

Enter string:yes
Reversed string: sey

```

1.7 Display palindrome

Write a program to display all the palindrome words appearing in an input text.

```

In [10]: s = input("Enter a text:")
         word_list = s.split()

         for word in word_list:
             if word == word[::-1]:
                 print(word)

```

```

Enter a text:hello aa
aa

```

1.8 Print ASCII code

Write a python program to display the ASCII code of the given character.

```

In [ ]: s = input("Enter a character: ")
         print("ASCII value of " + s + " is ", ord(s))

```

1.9 Compute score

Write a python program to compute percentage score of a student by reading his scores in 6 subjects, also display the name and score of the subject with highest and lowest score.

```
In [13]: m1 = float(input("Enter marks for the first subject:"))
        m2 = float(input("Enter marks for the second subject:"))
        m3 = float(input("Enter marks for the third subject:"))
        m4 = float(input("Enter marks for the fourth subject:"))
        m5 = float(input("Enter marks for the fifth subject:"))
        m6 = float(input("Enter marks for the sixth subject:"))
        score_list = [m1, m2, m3, m4, m5, m6]
        print("Score:", sum(score_list) / len(score_list))
        print("Max:", max(score_list))
        print("Min:", min(score_list))
```

```
Enter marks for the first subject:67
Enter marks for the second subject:78
Enter marks for the third subject:98
Enter marks for the fourth subject:56
Enter marks for the fifth subject:76
Enter marks for the sixth subject:67
Score: 73.66666666666667
Max: 98.0
Min: 56.0
```

1.10 Compute gross pay

Write a python program to prompt the user for hours and rate per hour to compute gross pay.

```
In [16]: hours = int(input("Enter the number of hours:"))
        rate = float(input("Enter the rate:"))
        gross_pay = hours * rate
        print("Gross pay:", gross_pay)
```

```
Enter the number of hours:8
Enter the rate:12
Gross pay: 96.0
```

1.11 Temperature Conversion

Write a program which prompts the user for a Celsius temperature, convert the temperature to Fahrenheit, and print out the converted temperature.

```
In [43]: c = float(input("Enter the temperature in Celsius:"))
        f = float((c * 1.8) + 32)
        print("Temperature in Fahrenheit is ", f)
```

```
Enter the temperature in Celsius:0
Temperature in Fahrenheit is 32.0
```

1.12 Flipping bits

Write a python program to read a number A which contains only digits 0's and 1's. Check if it is possible to make all the digits same by just flipping one digit. Print 'YES' if flipping results in making all digit's same else print 'NO'.

Test Case	Input	Output
Test Case 1	10	YES
Test Case 2	1011	YES
Test Case 3	0011	NO
Test Case 4	11	NO
Test Case 5	10	YES
Test Case 6	0	YES

```
In [28]: a = input("Enter a binary number:")
num_ones = 0
num_zeroes = 0
for bit in a:
    if bit == '0':
        num_zeroes += 1
    if bit == '1':
        num_ones += 1
if abs(len(a) - num_ones) == 1 or abs(len(a) - num_zeroes) == 1:
    print("YES")
else:
    print("NO")
```

```
Enter a binary number:1001
NO
```

1.13 Prime numbers in range

Write a program to list out all prime numbers between two integers m and n.

```
In [44]: print("Enter the range: ")
m = int(input())
n = int(input())
print("Prime numbers between ", m, " and ", n, " are")
for i in range(m, n + 1):
    if i > 1:
        for j in range(2, i):
            if (i % j) == 0:
                break
```

```

        else:
            print(i)

enter the range:
3
19
Prime numbers between 3 and 19 are
3
5
7
11
13
17
19

```

1.14 Frequency of characters

Write a program to count the frequency of characters in a string.

```

In [45]: s = input("Enter string: ")
        freq = {}
        for i in s:
            if i in freq:
                freq[i] += 1
            else:
                freq[i] = 1
        print("Frequency of all characters in " + s + " is :\n " + str(freq))

```

```

Enter string: alsxm
Frequency of all characters in alsxm is :
{'a': 1, 'l': 1, 's': 1, 'x': 1, 'm': 1}

```

1.15 String method

Write a Python function to get a string made of the first 2 and the last 2 chars from a given string. If the string length is less than 2, return empty string, otherwise return new string created.

```

In [27]: def str_method(s):
        if len(s) < 2:
            return ''
        else:
            return s[:2] + s[-2:]

        str_method("ppqqrr")

```

```

Out[27]: 'pprr'

```

1.16 Display string without spaces

Consider a string `s = Hello how are you?.` Write a program to display this string without spaces.

```
In [2]: s = "Hello, how are you?"
        print(s)
        s = s.replace(" ", "")
        print(s)
```

```
Hello, how are you?
Hello,howareyou?
```

1.17 Remove odd index values in a string

Write a Python program to remove the characters which have odd index values of a given string.

```
In [8]: def odd_values_string(str):
        result = ""
        for i in range(len(str)):
            if i % 2 == 0:
                result = result + str[i]
        return result

        s = input("Enter a string:")
        print(odd_values_string(s))
```

```
Enter a string:abcde
ace
```

1.18 Last character exchange

Write a Python program to change a given string to a new string where the first and last chars have been exchanged.

```
In [11]: s = input("Enter a string:")
        temp_beg = s[0]
        temp_end = s[-1]
        new_s = temp_end + s[1:-1] + temp_beg
        print(new_s)
```

```
Enter a string:abcde
ebcda
```


1.19 Add 'ing' and 'ly'

Write a Python program to add 'ing' at the end of a given string (length should be at least 3). If the given string already ends with 'ing' then add 'ly' instead. If the string length of the given string is less than 3, leave it unchanged. Sample String : 'abc' Expected Result : 'abcing' Sample String : 'string' Expected Result : 'stringly'

```
In [1]: s = input("Enter a string:")
        if len(s) >= 3:
            if s[-3:] == 'ing':
                s += 'ly'
            else:
                s += 'ing'
        print(s)
```

```
Enter a string:sds
sdsing
```

1.20 Fibonacci and factorial

Write a python program to generate first n Fibonacci number and factorial of n using functions.

```
In [30]: def factorial(n):
        fact = 1
        for i in range(1, n + 1):
            fact = fact * i
        print(fact)

        def fibonacci(n):
            fib1 = 0
            fib2 = 1
            fib3 = fib1 + fib2
            print(fib1)
            print(fib2)
            for i in range(2, n):
                print(fib3)
                fib1 = fib2
                fib2 = fib3
                fib3 = fib1 + fib2

        try:
            n = int(input("Enter a number:"))
            print("Factorial:")
            factorial(n)
            print("Fibonacci")
            fibonacci(n)
```

```
except ValueError:
    print("Not an integer.")
```

Enter a number:5

Factorial:

120

Fibonacci

0

1

1

2

3

1.21 HCF and LCM

Write a program to find HCF and LCM of number.

```
In [9]: def hcf(x, y):
        while (y):
            x, y = y, x % y
        return x

def lcm(x, y):
    lcm = (x * y) // hcf(x, y)
    return lcm

a = int(input("Enter a value for a:"))
b = int(input("Enter a value for b:"))

print(hcf(a, b))
print(lcm(a, b))
```

Enter a value for a:10

Enter a value for b:35

5

70

1.22 String operations

Write a python program to illustrate reverse, compare, copy and concatenate operations on strings without using built in functions, also compute lengths of the strings being read without using len().

```
In [10]: a = input("Enter a string a:")
        b = input("Enter a string b:")
```

```

c = ''
new_string = ''
new_string = a[::-1]
print(new_string)

print(a == b)

for ch in a:
    c += ch
print(c)

c = a + b
print(c)

count = 0
for ch in a:
    count += 1
print(count)

```

```

Enter a string a:Hello
Enter a string b:Kenobi
olleH
False
Hello
HelloKenobi
5

```

1.23 Horner's method

Write a python program to evaluate the polynomial

$$f'(x) = a_4x^4 + a_3x^3 + a_2x^2 + a_1x + a_0$$

for a given value of x and its coefficients using Horner's method.

```

In [16]: def horner(poly, n, x):
        result = poly[0]
        for i in range(1, n):
            result = result * x + poly[i]
        print("The value of polynomial at x = " + str(x) + " is " + str(result))

a0 = int(input("Enter a value for a0:"))
a1 = int(input("Enter a value for a1:"))
a2 = int(input("Enter a value for a2:"))
a3 = int(input("Enter a value for a3:"))
a4 = int(input("Enter a value for a4:"))

```

```

poly = [a0, a1, a2, a3, a4]
x = int(input("Enter the value for x:"))
n = len(poly)
horner(poly, n, x)

```

```

Enter a value for a0:0
Enter a value for a1:1
Enter a value for a2:2
Enter a value for a3:3
Enter a value for a4:4
Enter the value for x:2
The value of polynomial at x = 2 is 26

```

1.24 Decimal to binary

Write a python program to convert decimal number to it's equivalent binary.

```

In [40]: dec_val = int(input("Enter a decimal value:"))
        bin_val = bin(dec_val)
        print(bin_val)
        temp_dec = dec_val
        rem = int()
        l = []
        while temp_dec != 0:
            rem = temp_dec % 2
            l.insert(0, int(rem))
            temp_dec //= 2
        print(l)
        len(l)
        s = str()
        while ch in l:
            s += l[t]
        print(s)

```

```

Enter a decimal value:10
0b1010
[1, 0, 1, 0]

```

1.25 is_prime()

Write a python function `is_prime(num)` that accepts an integer argument and returns 1 if the argument is prime, a 0 otherwise. Write a python program that invokes this function to generate prime numbers between the given ranges.

```

In [50]: def is_prime(a):
        flag = 0

```

```

    for i in range(2, a):
        if a % i == 0:
            flag = 1
            break
    if flag == 1:
        return False
    else:
        return True

lower = int(input("Enter the lower value:"))
upper = int(input("Enter the upper value:"))

for i in range(lower, upper + 1):
    if is_prime(i):
        print(i)

```

```

Enter the lower value:3
Enter the upper value:13
3
5
7
11
13

```

1.26 String replacement

Write a python program to replace each constant in a string with the next one except letter 'z', 'Z', 'a' and 'A'. Thus the string Programming is Fun should be modified as Qsphsannjoh jt gvo.

```

In [56]: s = input("Enter a string:")
        new_s = ''
        ascii_val = 0
        for ch in s:
            if ch not in ['Z', 'z', 'a', 'A']:
                new_s += chr(ord(ch) + 1)
            else:
                new_s += ch
        new_s

```

```

Enter a string:program

```

```

Out[56]: 'qsphsan'

```

1.27 File operations

Write a program to read through a file and print the contents of the file (line by line) all in upper case, also count number of characters, words and lines in a file.

```

In [31]: line_count = 0
        char_count = 0
        word_count = 0
        s = ''
        with open("text_file.txt", 'r') as file:
            for line in file:
                words = line.split()
                line_count += 1
                word_count += len(words)
                char_count += len(line)
                print(line.upper())
        print(line_count, word_count, char_count)

```

SOME RANDOM STRING

ANOTHER RANDOM STRING

THIRD RANDOM STRING

CGPA:9.7

CGPA:8.5

NEXT STRING

6 13 90

1.28 Search file for CGPA

Write a program to prompt for a file name, and then read through the file and look for lines of the form: CGPA: 9.7 When a line that starts with "CGPA:" is encountered, pull apart the line to extract the floating- point number on the line. Count these lines and then compute the total of the CGPA. When the end of file is reached, print out the average CGPA.

```

In [11]: import re
        file_name = input("Enter a file name:")
        file = open(file_name, 'r')
        cgpa_list = []
        temp = []
        for line in file:
            if line.startswith("CGPA:"):
                temp = re.findall("[0-9]+.[0-9]+", line)
                if len(temp) == 1:
                    cgpa = float(temp[0])
                    cgpa_list.append(cgpa)
                temp = []
            else:

```

```

        continue
    print("Sum: ", sum(cgpa_list), " Mean: ", sum(cgpa_list) / len(cgpa_list))

```

Enter a file name:text_file.txt

Sum: 18.2 Mean: 9.1

1.29 Search file for string

Write a program to search and print all the lines starting with a specific word (taken as keyboard input) in a file.

```

In [13]: s = input("Enter the search string:")
        file = open("text_file.txt", 'r')
        for line in file:
            if line.startswith(s):
                print(line)

```

Enter the search string:CGPA

CGPA:9.7

CGPA:8.5

1.30 Student Name - USN file operations

Given two university information files “studentname.txt” and “usn.txt” that contains students Name and USN respectively. Write a python program to create a new file called “output.txt” and copy the content of files “studentname.txt” and “usn.txt” into output file in the sequence shown below. Display the contents of output file “output.txt” on to the screen.

```

In [29]: student_file = open("studentname.txt", 'r')
        usn_file = open("usn.txt", 'r')
        student_list = []
        usn_list = []
        for line in student_file:
            student_list.append(line.rstrip())
        for line in usn_file:
            usn_list.append(line.rstrip())

        student_usn_dict = dict(zip(student_list, usn_list))
        print(student_usn_dict)
        with open("output.txt", 'w') as output_file:
            output_file.write("Student Name\tUSN\n")
            for key, val in student_usn_dict.items():
                # print(key, val)
                output_file.write(key + "\t" + val + "\n")

```

```
{'Akshay': '4SF16IS012', 'Mahesh': '4SF16IS801', 'Ashik': '4SF16IS045', 'Harshith': '4SF16IS900'}
```