

Rajalakshmi Engineering College

Name: shobbika T
Email: 240701502@rajalakshmi.edu.in
Roll no: 240701502
Phone: 7305423247
Branch: REC
Department: I CSE FE
Batch: 2028
Degree: B.E - CSE

Scan to verify results



NeoColab_REC_CS23221_Python Programming

REC_Python_Week 4_CY

Attempt : 1
Total Mark : 40
Marks Obtained : 40

Section 1 : Coding

1. Problem Statement

Amrita is developing a password strength checker for her website. She wants the checker to consider the length and the diversity of characters used in the password. A strong password should be long and include a mix of character types: uppercase, lowercase, digits, and special symbols.

She also wants the feedback to be user-friendly, so she wants to include the actual password in the output. Help Amrita finish this password checker using Python's built-in string methods.

Character Types Considered:

Lowercase letters (a-z) Uppercase letters (A-Z) Digits (0-9) Special characters (from string.punctuation, e.g. @, !, #, \$)

Input Format

The input consists of a single string representing the user's password.

Output Format

The program prints the strength of the password in this format:

If the password length < 6 characters or fewer than 2 of the 4 character types, the output prints "<password> is Weak"

If password length ≥ 6 and at least 2 different character types, the output prints "<password> is Moderate"

If Password length ≥ 10 and all 4 character types present, the output prints "<password> is Strong"

Refer to the sample output for formatting specifications.

Sample Test Case

Input: password123

Output: password123 is Moderate

Answer

```
# You are using
import string
s=input().strip()
def checker (s):
    l=len(s)
    has_lower = any(c.islower() for c in s)
    has_upper = any(c.isupper() for c in s)
    has_digit = any(c.isdigit() for c in s)
    has_special = any(c in string.punctuation for c in s)
    types_count = sum([has_lower, has_upper, has_digit, has_special])
    if l >= 10 and types_count == 4:
        strength = "Strong"
    elif l >= 6 and types_count >= 2:
        strength = "Moderate"
    else:
        strength = "Weak"
```

```
print(f"{s} is {strength}")  
checker(s)
```

Status : Correct

Marks : 10/10

2. Problem Statement

Arjun is working on a mathematical tool to manipulate lists of numbers. He needs a program that reads a list of integers and generates two lists: one containing the squares of the input numbers, and another containing the cubes. Arjun wants to use lambda functions for both tasks.

Write a program that computes the square and cube of each number in the input list using lambda functions.

Input Format

The input consists of a single line of space-separated integers representing the list of input numbers.

Output Format

The first line contains a list of the squared values of the input numbers.

The second line contains a list of the cubed values of the input numbers.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 1 2 3

Output: [1, 4, 9]

[1, 8, 27]

Answer

```
input_numbers = list(map(int, input().split()))
```

```
square = lambda x: x ** 2
cube = lambda x: x ** 3
squared_list = list(map(square, input_numbers))
cubed_list = list(map(cube, input_numbers))
print(squared_list)
print(cubed_list)
```

Status : Correct

Marks : 10/10

3. Problem Statement

Create a program for a mathematics competition where participants need to find the smallest positive divisor of a given integer n . Your program should efficiently determine this divisor using the `min()` function and display the result.

Input Format

The input consists of a single positive integer n , representing the number for which the smallest positive divisor needs to be found.

Output Format

The output prints the smallest positive divisor of the input integer in the format: "The smallest positive divisor of $[n]$ is: $[smallest\ divisor]$ ".

Refer to the sample output for the exact format.

Sample Test Case

Input: 24

Output: The smallest positive divisor of 24 is: 2

Answer

```
n = int(input())
smallest_divisor = min((i for i in range(2, n + 1) if n % i == 0), default=1)
print(f"The smallest positive divisor of {n} is: {smallest_divisor}")
```

Status : Correct

Marks : 10/10

4. Problem Statement

Implement a program for a retail store that needs to find the highest even price in a list of product prices. Your goal is to efficiently determine the maximum even price from a series of product prices. Utilize the `max()` inbuilt function in the program.

For example, if the prices are 10 15 24 8 37 16, the even prices are 10 24 8 16. So, the maximum even price is 24.

Input Format

The input consists of a series of product prices separated by a space.

The prices should be entered as a space-separated string of numbers.

Output Format

If there are even prices in the input, the output prints "The maximum even price is: " followed by the maximum even price.

If there are no even prices in the input, the output prints "No even prices were found".

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 10 15 24 8 37 16

Output: The maximum even price is: 24

Answer

```
# You are using Python
p=list(map(int,input().split()))
even=list(filter(lambda x: x%2==0 , p))
if even:
    print("The maximum even price is:",max(even))
else:
    print("No even prices were found")
```

Status : Correct

Marks : 10/10