CS242-254: Programming Assignment I

กำหนดส่ง 28 กุมภาพันธ์ 2568 (ภายใน 23:59 น.)

จงเขียนฟังก์ชันสำหรับข้อต่อไปนี้

1. Write a function **get_dice_throws_result()** that throws a number of dice (given by num_throws) and counts how often the dice value, (given by dice_to_check) occurs.
   (Hint: use random.randint(). You results may not be the same as shown below)
   Example:

```
print("30000 throws,", get_dice_throws_result(30000, 6), "sixes")
print("6 throws,", get_dice_throws_result(6, 6), "sixes")
print("600000 throws,", get_dice_throws_result(600000, 5), "fives")
```

```
30000 throws, 4913 sixes
6 throws, 0 sixes
600000 throws, 99929 fives
```

2. Write a **get_sum_of_divisors()** function that sum of all the divisors less than the number passed to the function itself.
   For an integer, a divisor is a number which divides exactly into the integer (a factor of the integer), e.g., the divisors of 6 are 1, 2, 3, 6. So, the sum of all the divisors less than the number itself is (1+2+3 = 6).
   Example:

```
print("get_sum_of_divisors(6)", get_sum_of_divisors(6))
print("get_sum_of_divisors(24)", get_sum_of_divisors(24))
print("get_sum_of_divisors(25)", get_sum_of_divisors(25))
print("get_sum_of_divisors(5628)", get_sum_of_divisors(5628))
```

```
get_sum_of_divisors(6) 6
get_sum_of_divisors(24) 36
get_sum_of_divisors(25) 6
get_sum_of_divisors(5628) 9604
```

3. A perfect number is an integer that is equal to the sum of its divisors (including 1, excluding the number itself), e.g., the sum of the divisors of 28 is 28 (1 + 2 + 4 + 7 + 14). Write the **check_perfection()** function which checks for perfection and prints either '# is a perfect number'or '# is NOT a perfect number'.
   Example:

```
check_perfection(28)
check_perfection(54)
check_perfection(496)
```

```
28 is a perfect number
54 is NOT a perfect number
496 is a perfect number
```

4. Write the **user_number_guess()** function which keeps prompting the user to guess a hidden number until the user correctly guesses the number. At each guess the function lets the user know if the guess is too high or too low. At the end, the function also prints the number of guesses taken.
   Example:

   ```
   user_number_guess(random.randrange(1, 100))
   ```

   ```
   Enter your guess (1 - 99): 50 Too high
   Enter your guess (1 - 99): 25 Too high
   Enter your guess (1 - 99): 13 Too low
   Enter your guess (1 - 99): 20 Too low
   Enter your guess (1 - 99): 23 Correct! Number of guesses: 5
   ```

5. Write a function **triangle_type(a, b, c)** that takes three side lengths oof a triangle and determines whether it is:
   - Equilateral (all sides equal)
   - Isosceles (two sides equal)
   - Scalene (all sides different)
   - Not a valid triangle (violates the triangle inequality rule)

6. Write a function **rps_winner(player1, player2)** that determines the winner of a Rock-Paper-Scissors game.
   - Valid inputs: “rock”, “paper”, “scissors”
   - Determine the winner or return “Tie” if both players choose the same.

7. Write a function **password_strength(password)** that evaluates password strength based on these rules:
   - **Strong**: At least 8 characters long, contains both letters and numbers.
   - **Medium**: At least 6 characters long, but missing either letters or numbers.
   - **Weak**: Fewer than 6 characters.

8. Write a function **mood_detector(message**) that analyzes a text message and classifies the mood as:
   - “**Happy**” if it contains words like “happy”, “joy”, “excited”, “great”, “fun”.
   - “**Sad**” if it contains “sad”, “depressed”, “bad”, “terrible”, “cry”
   - “**Neutral**” otherwise
   - The function should ignore case sensitivity.

9. Write a function **validate_credit_card(card_number)** that checks if a credit card number is valis using the Luhn Algorithm:
   - Double every second digit from right to left.
   - If doubling a digit results in a number greater than 9, subtract 9 from it.
   - Sum all digits. If the total is divisible by 10, the card is valid.

10. Write a function **convert_time(time_str)** that converts a 12-hour format time (AM/PM) to 24-hour format.
    Example:

    ```
    Print(convert_time(“07:15 PM”))
    Print(convert_time(“12:30 AM”))
    Print(convert_time(“11:59 AM”))
    ```

    ```
    19:15
    00:30
    11:59
    ```