Assignment1_data-Science-Python

September 28, 2024

1 Assignment 1: Introduction to Data Science & Python

1.0.1 1. Factors of a Number with Even or Odd Check:

This program finds the factors of a number and determines if each factor is even or odd. Example: For 20, the factors are 1 (odd), 2 (even), 4 (even), 5 (odd), 10 (even), and 20 (even).

[3]:
```python
# 1. Program to find factors of a number and check if the factor is even or odd
def find_factors(num):
    factors = []
    for i in range(1, num + 1):
        if num % i == 0:
            if i % 2 == 0:
                print(f"{i} is an even factor")
            else:
                print(f"{i} is an odd factor")
            factors.append(i)
    return factors

# Example usage:
print("Factors of 20 and their type:")
find_factors(20)
```

```
Factors of 20 and their type:
1 is an odd factor
2 is an even factor
4 is an even factor
5 is an odd factor
10 is an even factor
20 is an even factor
```

[3]:
```
[1, 2, 4, 5, 10, 20]
```

1.0.2 2. Sort Words Alphabetically:

This program accepts a sequence of words from the user and prints them in alphabetical order. You can run this function interactively with user input.

```
[4]: # 2. Program to sort input words alphabetically
     def sort_words():
         words = input("Enter a sequence of words, separated by spaces: ").split()
         words.sort()
         print("Sorted sequence of words:", " ".join(words))

     # Example usage:
     sort_words()
```

```
Enter a sequence of words, separated by spaces:  one two three four five

Sorted sequence of words: five four one three two
```

1.0.3 3. Find Numbers with All Even Digits Between 1000 and 3000:

This program finds numbers between 1000 and 3000 where all digits are even. Example: Some of the numbers found are 2000, 2020, 2040, etc.

```
[5]: # 3. Program to find numbers between 1000 and 3000 where all digits are even
     def even_digit_numbers():
         result = []
         for num in range(1000, 3001):
             digits = str(num)
             if all(int(digit) % 2 == 0 for digit in digits):
                 result.append(digits)
         print(",".join(result))

     # Example usage:
     even_digit_numbers()
```

```
2000,2002,2004,2006,2008,2020,2022,2024,2026,2028,2040,2042,2044,2046,2048,2060,
2062,2064,2066,2068,2080,2082,2084,2086,2088,2200,2202,2204,2206,2208,2220,2222,
2224,2226,2228,2240,2242,2244,2246,2248,2260,2262,2264,2266,2268,2280,2282,2284,
2286,2288,2400,2402,2404,2406,2408,2420,2422,2424,2426,2428,2440,2442,2444,2446,
2448,2460,2462,2464,2466,2468,2480,2482,2484,2486,2488,2600,2602,2604,2606,2608,
2620,2622,2624,2626,2628,2640,2642,2644,2646,2648,2660,2662,2664,2666,2668,2680,
2682,2684,2686,2688,2800,2802,2804,2806,2808,2820,2822,2824,2826,2828,2840,2842,
2844,2846,2848,2860,2862,2864,2866,2868,2880,2882,2884,2886,2888
```

1.0.4 4. Count Letters and Digits in a Sentence:

This program accepts a sentence and calculates the number of letters and digits in it. You can run this function interactively with user input.

```
[7]: # 4. Program to count the number of letters and digits in a sentence
     def count_letters_digits():
         sentence = input("Enter a sentence: ")
         letters = sum(c.isalpha() for c in sentence)
         digits = sum(c.isdigit() for c in sentence)
```

```
    print(f"LETTERS: {letters} DIGITS: {digits}")

# Example usage:
count_letters_digits()
```

```
Enter a sentence:  Python12345
```

```
LETTERS: 6 DIGITS: 5
```

1.0.5 5. Palindrome Number Check:

This program checks whether a given number is a palindrome (reads the same forwards and backwards). Example: 12321 is a palindrome, but 12345 is not.

[8]:
```
# 5. Program to check if a number is a palindrome
def is_palindrome(num):
    num_str = str(num)
    if num_str == num_str[::-1]:
        print(f"{num} is a palindrome")
    else:
        print(f"{num} is not a palindrome")

# Example usage:
is_palindrome(12321)
is_palindrome(12345)
```

```
12321 is a palindrome
12345 is not a palindrome
```