Task1:

You need to write a function in python that return all such numbers which are

divisible by 7 but are not a multiple of 5, between 2000 and 3200 (both included).

Return the obtained numbers in a comma-separated sequence on a single line.

In [8]:

```
def find_numbers():
    # Create an empty list to store the results
    result = []

    # Loop through numbers from 2000 to 3200 (both inclusive)
    for num in range(2000, 3201):
        # Check if the number is divisible by 7 and not a multiple of 5
        if num % 7 == 0 and num % 5 != 0:
            # Add the number to the result list
            result.append(str(num))  # Convert number to string for final output

    # Return the result as a comma-separated string
    return ",".join(result)

# Print the output
print(find_numbers())
```

```
2002,2009,2016,2023,2037,2044,2051,2058,2072,2079,2086,2093,2107,2114,2121,2128,2142,2149,2156,2163,2177,2184,21
91,2198,2212,2219,2226,2233,2247,2254,2261,2268,2282,2289,2296,2303,2317,2324,2331,2338,2352,2359,2366,2373,2387
,2394,2401,2408,2422,2429,2436,2443,2457,2464,2471,2478,2492,2499,2506,2513,2527,2534,2541,2548,2562,2569,2576,2
583,2597,2604,2611,2618,2632,2639,2646,2653,2667,2674,2681,2688,2702,2709,2716,2723,2737,2744,2751,2758,2772,277
9,2786,2793,2807,2814,2821,2828,2842,2849,2856,2863,2877,2884,2891,2898,2912,2919,2926,2933,2947,2954,2961,2968,
2982,2989,2996,3003,3017,3024,3031,3038,3052,3059,3066,3073,3087,3094,3101,3108,3122,3129,3136,3143,3157,3164,31
71,3178,3192,3199
```

Task2:

You need code a function that calculates and returns the value according to the

given formula:

P = Square root of [(2 * A * B)/C]

Following are the fixed values of A and B:

A is 50. B is 30.

The values of the literal C should be taken as console input to your program in a

comma-separated sequence.

Example

Let us assume the following comma separated input sequence is given to the program:

100,150,180

The output of the program should be: 18,22,24

In [2]:

```
import math

# Fixed values of A and B
A = 50
B = 30

# Input values for C as a comma-separated string
input_values = input("Enter values for C (comma-separated): ")

# Convert input string into a list of integers
C_values = list(map(int, input_values.split(',')))

# Calculate P for each value of C
P_values = [int(math.sqrt((2 * A * B) / C)) for C in C_values]

# Print the results as comma-separated values
print(','.join(map(str, P_values)))
```

```
54,38,31
```

In [3]:

```
# Fixed values of A and B
```

```python
A = 50
B = 30

# Input values for C as a comma-separated string
input_values = input("Enter values for C (comma-separated): ")

# Convert input string into a list of integers
C_values = list(map(int, input_values.split(',')))

# Function to calculate square root without using math module
def sqrt(n):
    x = n
    y = (x + 1) // 2
    while y < x:
        x = y
        y = (x + n // x) // 2
    return x

# Calculate P for each value of C
P_values = [int(sqrt((2 * A * B) // C)) for C in C_values]

# Print the results as comma-separated values
print(','.join(map(str, P_values)))
```

```
54,38,31
```

Task3:

You need to write a function that takes a comma separated sequence of words as

input and prints the words in a comma-separated sequence after sorting them alphabetically.

Suppose the following input is supplied to the program:

- without,hello,bag,world
- Then, the output should be:
- bag,hello,without,world

In [4]:

```python
def words(text):
    print(text)
    tex = text.split(",")
    tex.sort()
    return ",".join(tex)

input_word = input("Enter the words in a comma-separated sequence: ")
words(input_word)
```

```
wie,q2e,d,
```

Out[4]:

```
',d,q2e,wie'
```

In [5]:

```python
def sort_words(input_str):
    # Split the input string into a list of words
    words = input_str.split(',')

    # Sort the words alphabetically
    words.sort()

    # Join the sorted words back into a comma-separated string
    result = ','.join(words)

    # Print the result
    print(result)

# Input
input_str = input("Enter words separated by commas: ")
sort_words(input_str)
```

```
123,123asdksd,a
```

Task4:

You need to write a program that takes sequence of lines as input and prints the lines after making all characters in the sentence capitalized.

Suppose the following input is supplied to the program:

- Hello world

- Practice makes perfect

Then, the output should be:

- HELLO WORLD
- PRACTICE MAKES PERFECT

In [6]:

```
def capitalize_lines():
    print("Enter lines (press Enter twice to stop):")
    lines = []

    while True:
        line = input()
        if line == "":
            break
        lines.append(line.upper())

    print("\nCapitalized Output:")
    for line in lines:
        print(line)

capitalize_lines()
```

```
Enter lines (press Enter twice to stop):
Capitalized Output:
1
2
3
```

Task5:

You need to write a function that counts the number of vowels in a given sentence as input from console.

Suppose the following input is supplied to the program:

Hello world
Practice makes perfect

Then, the output should be:

- a appeared 2 times
- e appeared 5 times
- i appeared 1 time
- o appeared 2 times
- u appeared 0 time

PRACTICE MAKES PERFECT

In [ ]:

```
def sentence_input(text:str):
    vowels = ["a", "e", "i", "o", 'u']
    for i in range(len(vowels)):
        print(f"{vowels[i]} appeared {text.count(vowels[i])} times" )
        uppertext = text.upper()
    text = uppertext.split()
    print("")
    print(" ".join(text[2:]) ,end = "")
input_sentence = input("Enter a sentence: ")
sentence_input(input_sentence)
```

Task6:

You need write a function that traces and makes a list of all such numbers from1000 to 3000 in which all the digits are even numbers.

In [3]:

```
def find_even_digit_numbers():
    even_digit_numbers = []
    for number in range(1000, 3001):  # Include 3000 in the range
        num_str = str(number)
        if all(int(digit) % 2 == 0 for digit in num_str):  # Check if all digits are even
            even_digit_numbers.append(number)
    return even_digit_numbers

# Example usage
result = find_even_digit_numbers()
print(result)
```

```
[2000, 2002, 2004, 2006, 2008, 2020, 2022, 2024, 2026, 2028, 2040, 2042, 2044, 2046, 2048, 2060, 2062, 2064, 206
6, 2068, 2080, 2082, 2084, 2086, 2088, 2200, 2202, 2204, 2206, 2208, 2220, 2222, 2224, 2226, 2228, 2240, 2242, 2
244, 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, 26
66, 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]
```

Task7:

You need to write a code which accepts a sequence of comma separated 4 digit binary numbers as its input and then check whether they are divisible by 5 or not.

The numbers that are divisible by 5 are to be printed in a comma separated sequence.

Example:

- 0100,0011,1010,1001

Then the output should be:

- 1010

In [1]:

```
def binary_digit_num_check(accept_binary_num):
    binary_num = accept_binary_num.split(",")
    for num in binary_num:
        if int(num, 2) % 5 == 0:
            print(num, end = ",")
binary_num = input("Enter the binary number: ")
binary_digit_num_check(binary_num)
```

```
1010,
```

Task8:

Write a program that accepts a sentence and calculate the number of letters and digits.

Suppose the following input is supplied to the program:

- hello world! 123
- Then, the output should be:
- LETTERS 10
- DIGITS 3

In [1]:

```
def count_letters_digits(sentence):
    letters = sum(1 for char in sentence if char.isalpha())
    digits = sum(1 for char in sentence if char.isdigit())
    return letters, digits

sentence = input("Enter a sentence: ")

letters, digits = count_letters_digits(sentence)

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

```
LETTERS 10
DIGITS 3
```

In [ ]:

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js