11. Taylor approximation of trigonometric function

From the Taylor series of $\sin(x)$:

$$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \cdots$$

and $\cos(x)$

$$\cos x = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \cdots$$

write a function **sincos_truncate(x,n)** to return an approximation of these two function's evaluation at x where $n \geq 1$ is the number of terms used in the summation. Round your answer to have 4 decimal digits.

Input Format

- x: a floating-point number
- n: an integer representing the number of terms to use in the Taylor series (where $n \geq 1$)

Output Format

Return a tuple (sin_x, cos_x) where:

- sin_x: approximation of $\sin(x)$ using the first n terms, rounded to 4 decimal places
- cos_x: approximation of $\cos(x)$ using the first n terms, rounded to 4 decimal places

Constraints

- $x \in \mathbb{R}$, $|x| \leq 10$
- $1 \leq n \leq 20$
- Must calculate using float (integer might overflow)

Sample Input

```
x = np.pi / 3
n = 6
```

Sample Output

```
(0.866, 0.5)
```

Implementation

Goal: Fill in the following function:

```
def sincos_truncate(x,n):
    sinx = # your code
    cosx = # your code
    return sinx, cosx
exec("\n".join(iter(input, "#Exit"))) # Don’t remove this line
```

Note: Must calculate using float (integer might be overflown)

Hint

- Use `np.cumprod` for calculating cumulative product
- use `np.round` to round the result
- Remember to use `dtype=np.float64` to avoid integer overflow