Write a python program to find a factorial of a number?

In Python, you can compute factorial in **multiple ways**:

1. Using Iteration (Loop method)

```
def factorial_iterative(n):
    if n < 0:
        return "Factorial not defined for negative numbers"
    fact = 1
    for i in range(1, n + 1):
        fact *= i
    return fact

# Example
print(factorial_iterative(5))  # Output: 120
```

---

2. Using Recursion

```
def factorial_recursive(n):
    if n < 0:
        return "Factorial not defined for negative numbers"
    if n == 0 or n == 1:
        return 1
    return n * factorial_recursive(n - 1)

# Example
print(factorial_recursive(5))  # Output: 120
```

⚠️ For very large n, recursion can hit Python's recursion depth limit.

3. Using math.factorial()

```
import math

n = 5

print(math.factorial(n))  # Output: 120
```

✅ This is **optimized in C**, so fastest and safest.

---

4. Using functools.reduce()

```
from functools import reduce


def factorial_reduce(n):
  if n < 0:
    return "Factorial not defined for negative numbers"
  if n == 0:
    return 1
  return reduce(lambda x, y: x * y, range(1, n + 1))


print(factorial_reduce(5))  # Output: 120
```

---

5. Using While Loop

```
def factorial_while(n):
  if n < 0:
    return "Factorial not defined for negative numbers"
  fact, i = 1, 1
  while i <= n:
    fact *= i
    i += 1
  return fact
```

```
print(factorial_while(5))  # Output: 120
```

6. Using NumPy (for scientific computing)

```
import scipy.special
import numpy as np

arr = np.array([3, 4, 5])
print(scipy.special.factorial(arr, exact=True))  # [6 24 120]
```

🔑 **Summary**

- **Best choice** → math.factorial() (fast, safe).
- **Learning purpose** → Recursive, iterative, reduce.