BIS 420 PROGRAMMING FOR DATA SCIENCE

PRAJAKTA POHARE
CHAPTER 7 EXERCISE 7.5
ILLINOIS STATE UNIVERSITY

The mathematician Srinivasa Ramanujan found an infinite series that can be used to

generate a numerical approximation of 1/π:

$$\frac{1}{\pi} = \frac{2\sqrt{2}}{9801} \sum_{k=0}^{\infty} \frac{(4k)!(1103 + 26390k)}{(k!)^4 396^{4k}}$$

Write a function called estimate_pi that uses this formula to compute and return an estimate of π. It should use a while loop to compute terms of the summation until the last term is smaller than 1e-15 (which is Python notation for 10−15). You can check the result by comparing it to math.pi.

Output:

```
import math


def estimate_pi():
    k = 0
```

```python
    total_sum = 0
    factor = 2 * math.sqrt(2) / 9801

    while True:
        numerator = math.factorial(4 * k) * (1103 + 26390 * k)
        denominator = (math.factorial(k) ** 4) * (396 ** (4 * k))
        term = factor * (numerator / denominator)

        if term < 1e-15:
            break

        total_sum += term
        k += 1

    return 1 / total_sum

estimated_pi = estimate_pi()
print(f"Estimated value of pi: {estimated_pi}")
print(f"Difference from math.pi: {abs(math.pi - estimated_pi)}")
```

BIS420_PrajaktaPohare_Ch7_7.3.py | BIS420_PrajaktaPohare_Ch7_7.4.py | BIS420_PrajaktaPohare_Ch7_7.5.py

Users > prajaktapohare > Library > CloudStorage > OneDrive-ILStateUniversity > BIS420 > Week 7 > BIS420_PrajaktaPohare_Ch7_7.5.py > ...

```python
import math

def estimate_pi():
    k = 0
    total_sum = 0
    factor = 2 * math.sqrt(2) / 9801

    while True:
        numerator = math.factorial(4 * k) * (1103 + 26390 * k)
        denominator = (math.factorial(k) ** 4) * (396 ** (4 * k))
        term = factor * (numerator / denominator)

        if term < 1e-15:
            break

        total_sum += term
        k += 1

    return 1 / total_sum


estimated_pi = estimate_pi()
print(f"Estimated value of pi: {estimated_pi}")
print(f"Difference from math.pi: {abs(math.pi - estimated_pi)}")
```