コンピュータ

北海道大学理学部数学科

素数判定とエラトステネスのふるい

- 今回は素数の判定を問題とする。
- 具体的には N 以下の素数を全て列挙することである。
- N 以下の素数とは、1 から N までの全ての自然数について、どれも約数に持たない数であった。
- この条件を満たす自然数を出力するプログラムを作成する。

定義の通りに解決する

- N 以下の自然数 k について、k 未満の自然数を約数に持てば k は素数でない。
- $m = 2$ から $k - 1$ について、$k\%m$ の値を調べれば k が素数かどうか判定できる。
- ($k - 1$ まで調べる必要があるかどうか？)

定義の通りに解決する

プログラムを作る際には、まず次のような処理の流れを考える。

1. $k = 2, 3$ は素数である。
2. $k = 4, 5, \ldots, N$ について、以下を繰り返す。
   - 2.1 $j = 2, 3, \ldots, k-1$ について、k を j で割った余りを見る。
   - 2.2 ある j について余りが 0 となれば、k は素数ではない。

定義の通りに解決する

考えるべきは変数 j に関する繰り返し処理である。もう少していねいに書く。

1. 変数 *isPrime* を用意して、k が素数かどうかという情報を保持する。
2. 変数 *isPrime* の値が 1 ならば k は素数、0 ならば k は素数ではないとしよう。
3. まず、素数であると仮定しておく。つまり、$isPrime = 1$ と置く。
   - 3.1 $j = 2, 3, \ldots, k-1$ について k を j で割った余りを見る。
   - 3.2 ある j について余りが 0 となれば、k は素数ではない。
   - 3.3 $isPrime = 0$

定義の通りに解決する

ここまで処理の流れを書ければプログラムへ反映することができる。

```
n=100
for k in range(4,n+1):
        isPrime=1
  for j in range(2,k):
    if( k % j == 0 ):
      isPrime=0
  if( isPrime == 1 ):
    print(k)
```

プログラムでは、変数 isPrime が 1 か 0 かに応じて k が素数かどうかを判定している。このような変数をフラグと呼ぶ。

- 「素数でない」という情報は、$k\%j == 0$ という条件が j について**一度でも成立すれば**よく、プログラムとして処理しやすい。
- 「素数である」という情報は、**全ての** j について調べなければ判定できない。これはプログラムとして処理しにくい。
- 従って、まず素数であると仮定して「素数でない」ことを判定するプログラム例を示した。

- 素数でないことがわかった段階で不要な j に関する処理を省略するため、break 文を使用する。break 文は、最も内側のループから抜け出す処理をする。

```
n=100
for k in range(4,n+1):
        isPrime=1
  for j in range(2,k):
    if( k % j == 0 ):
      isPrime=0
                          break
  if( isPrime == 1 ):
    print(k)
```

課題 1

プログラム例を参考に、定義に従って n 以下の素数を出力するプログラムを作成し、実行する。ただし、j を k-1 まで調べる必要はなく、$\sqrt{k}$ 程度まで調べればよい。この理由を考えて、j の上限に反映させること（3 点）。

用語

以下、「配列」とは数値を要素とするリストである。

エラトステネスのふるい

- 定義通りのアルゴリズムは各自然数が素数かどうかを網羅的に判定した。
- 考え方を変えて、素数の倍数は素数ではないことを利用する。
- ある自然数が素数かどうかを記録するために、n 個の要素をもつ配列 p を用意する。全ての配列要素に初期値として 1 を代入し、素数であると仮定する。
- k が素数ではないとわかれば p[k]=0 とする。
- まず、2 は素数である。2 の倍数 k について p[k]=0 とする。
- p[3]=1 である。3 の倍数 k について p[k]=0 とする。
- p[m]=1 である各 m について、 m の倍数は素数ではない。m は素数として十分である。

プログラム例

```
n=20
# 配列 p を定義
p=[1]*(n+1)
# p の各配列要素へ 1 を代入
for k in range(n+1):
    p[k]=1
# k を 2 から n まで
for k in range(2,n):
  # k の倍数は素数ではない
  # k が素数の場合のみ次の処理を実行するよう修正する
  for j in range(k,n):
    if(k*j<n):
      p[k*j]=0
# 各 k について p[k]=1 であれば素数だから出力する
for k in range(2,n):
  if(p[k]==1):
    print(k)
```

課題 2

次の条件を加えてプログラム例を実行する（4 点）。

- k の変化する上限を考える。
- k が素数でない場合は j に関する for 文を実行しない。

素数定理

$pn(k)$ を k 以下の素数の数と定義する。素数定理とは、
$pn(k) \sim k/\log k$ を主張する定理である。k を 1000 程度まで
$pn(k)$ を求めプロットすると図 1 のように変化する。

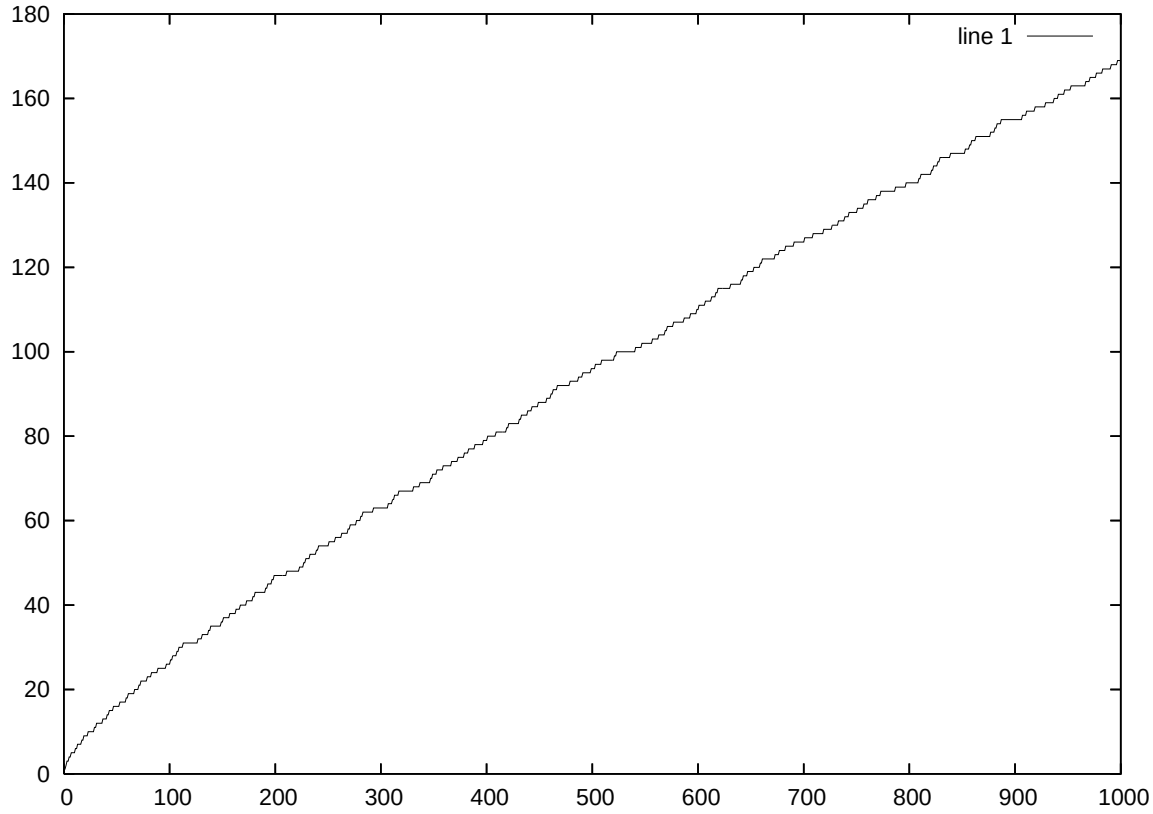

Figure: k 以下の素数の数

課題 3

k を与えたときに $pn(k)$ を出力する関数を作成する。課題 2 のプログラムの配列 p[n] を利用し、$n < k$ の範囲で p[n] の和を出力すればよい (3 点)。

```
def pn(n):
    s=0
    for m in range(2,n):
        s=s+
    return(s)
```