コンピュータ

北海道大学理学部数学科

第 5 回

1　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るから、プログラムを作成できる。

1.1　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、$mod(k, m)$ の値を調べれば k が素数かどうか判定できる。

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。
   (a) $j = 2, 3, \ldots, k - 1$ について、k を j で割った余りを見る。
   (b) ある j について余りが 0 となれば、k は素数ではない。

この流れで問題になるのは j に関する繰り返し処理である。これを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。値が 1 ならば素数、0 ならば素数ではないとしよう。
2. まず、素数であると仮定しておく。つまり、$isPrime = 1$ と置く。
   (a) $j = 2, 3, \ldots, k - 1$ について k を j で割った余りを見る。
   (b) ある j について余りが 0 となれば、k は素数ではない。
   (c) $isPrime = 0$

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。

1.2 プログラム

```
clear;
N=100;

for k=[4:N]
  # k を素数だと仮定しておく
  isPrime=1;
  for j=[2:____]
    # 約数があれば素数ではない
    if( mod(k,j) == 0)
      isPrime=0;
      break;
    endif
  endfor
  # 素数であれば k を出力
  if ( isPrime == 1 )
    printf( "%d\n", k);
  endif
endfor
```

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、$mod(k, j) == 0$ という条件が j について**一度でも成立すれば**よい。これは極めて処理しやすい。逆に、「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 文を使用した。break 文は、最も内側のループから抜け出す処理をする。この break 文がなくても正しく動作するが、不要な処理を続けることになる。

2 エラトステネスのふるい

定義の通りに書いたプログラムは、k を判定するために k 以下の自然数について約数かどうかを調べている。約数かどうかの判定は k 以下の素数について調べれば十分だから、効率を上げるために判定済みの素数を記録することにしよう。

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 を用意しよう。初期値として 1 を代入し、k が素数ではないとわかれば $p(k) = 0$ とする。

1. 配列 p([1:N]) を用意する。初期値として全ての要素に 1 を代入する。これは、1 から N までの各 k が素数であることを仮定する。

   (a) 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。

この処理を以下のようなプログラムにできる。

2.1 プログラム

```
clear;
N=1000;

for k=[1:N]
  p(k)=1;
endfor

for m=[2:_______]
  if( p(m)==1 )
    l=2;
    while( l*m <= N )
        p(l*m)=0;
        l=l+1;
    endwhile
  endif
endfor

for k=[1:N]
  if( p(k) == 1 )
    printf("%d ",k);
  endif
endfor
printf("\n");
```

3 課題

1. 穴埋めになっているループの上限を $k-1$ として, 定義通りのプログラムを実行する。N は input 文で入力する。
2. 穴埋めになっているループの上限を N として, ふるいのプログラムを実行する。N は input 文で入力する。

3. 上の 2 つのプログラムとも、穴埋めとしているループの上限は $\sqrt{k}$ と $\sqrt{N}$ を越える最小の自然数でよい。それぞれプログラムの該当部分を ceil(sqrt(k)) と ceil(sqrt(N)) に変更し、出力結果が変わらないことを確認せよ。また、これでよい理由を説明せよ。