ICPC2011 国内予選問題 A　チェビシェフの定理

方法 1

整数xが素数であるかどうかを判定するために、xを$2 \le i \le \sqrt{x}+1$なる整数iで割り切れるかどうかをチェックすれば良い[1]。方法 2 に比べるとかなり遅いが許容範囲である。

方法 2

n の最大値は 123456 なので、まず、エラトステネスのふるいにより 123456×2 までの素数を全て求めて（素数表を作成）から、素数の数を数える。

[方法 1 によるプログラム例]

```
// ICPC 2011 国内予選問題 A
// http://icpc2011.ait.kyushu-u.ac.jp/icpc2011/contest/all_ja.html#section_A
//          Filename = pa1.c
//          Compile:                cc pa1.c -lm
//          Execution: ./a.out < A0 > A0.result
//          Check:                  diff A0.ans A0.result
// x が素数かどうかを判定するのに 2 <= d <= sqrt(x)+1 なる d で
// 割り切れるかどうかで判定

#include <stdio.h>
#include <stdlib.h>
#include <math.h>

// x が素数なら 1, 素数でなければ 0 を返す関数
// 素数判定：x >= 3 の場合、2 <= d <= sqrt(x)+1 なるどれかの d で x を
// 割り切ることができれば x は素数ではない。
// 浮動小数点計算の誤差を考慮して d <= sqrt(x) ではなく d <= sqrt(x)+1
// としている
int is_prime(int x)
{
  int d, dmax;
  if(x==2) return 1;     // 2 は素数
  dmax = sqrt(x)+1;                  // dmax まで割り切れるかどうかを試す
  for(d=2; d<=dmax; d++){
    if(x % d == 0) return 0;         // x は d で割り切れたので素数ではない
  }
  // 2 <= d <= sqrt(x)+1 なる全ての d で割り切ることができなかったので素数である
  return 1;
}
```

---

[1] xが 2 以上の整数aで割り切れるとすると$x = ab$と表すことが出来る。この時、$Min(a,b) \le \sqrt{x}$である。したがって、浮動小数点計算の誤差を考慮して、2 以上$\sqrt{x}+1$以下の整数で割り切れるかどうかをチェックすれば良い。

```
int main()
{
   int n;
   int c;                    //  素数の個数を数える変数
   int x;
   while(1){
      scanf(″%d″, &n);   // n を入力
      if(n==0) exit(0);    // n==0 なら終了
      c = 0;                 //  素数の個数を数える変数を 0 に初期化
      for(x = n+1; x <= 2*n; x++)     // n より大きく 2n 以下の各整数 x に対し
         c += is_prime(x);              //  それが素数なら素数の個数を1増やす
      printf(″%d¥n″,c);   //  答えを出力
   }
}
```

[方法 2 によるプログラム例]

```
// ICPC 2011 国内予選 Problem A
// http://icpc2011.ait.kyushu-u.ac.jp/icpc2011/contest/all_ja.html#section_A
//            Filename = pa2.c
//            Compile:                  cc pa2.c
//            Execution:  ./a.out < A0 > A0.result
//            Check:                    diff A0.ans A0.result
//
//  最初に「エラトステネスのふるい」により素数を求めておく
//
#include <stdio.h>
#define MAXN 123456
int pf[2*MAXN+1];       // i が素数なら pf[i] = 1, そうでなければ pf[i] = 0

//  エラトステネスのふるいにより配列 pf をセット
void prepare()
{
   int i,j;
   for(i=0; i <= 2*MAXN; i++) pf[i] = 1;                          //  配列 pf を 1 で初期化
   for(i=2; i<= MAXN; i++){
      if(pf[i] == 0) continue;          // i が非素数の場合は何もしない。
      j = i+i;                                 // j = 2 x i
      while(j <= 2*MAXN){                          // i より大きい i の倍数 j は非素数
         pf[j] = 0;
         j += i;
      }
   }
}
```

```
int main()
{
   int n;
   int c;     // 素数の個数を数える変数
   int i;
   prepare();              // エラトステネスのふるいで素数を求める
   while(1){
      scanf("%d", &n);   // n を入力
      if(n == 0) break;   // n == 0 なら終了
      c = 0;              // 素数の個数を数える変数を0に初期化
      for(i=n+1; i <= 2*n; i++)        // n < i <= 2*n なる各 i に対して
         c += pf[i];                   // それが素数なら素数の個数を1増やす
      printf("%d¥n",c);
   }
}
```