**Crypto Hmwk 3**

**Due Wed Sept. 24th**

**Instructions:**

**\*Print out these pages.**

**\*Write your answers below.**

**\*Staple your computer programs and printouts to it.**

**Homework Assignment 3**

**1) Find the number of Pythagorean primes not exceeding 500. π1,4(500) = \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.**

**2) Find π3,4(500) = \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**3) Find π(500) = \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**4) Verify that π1,4(500) + π3,4(500) = π(500).**

**5) Find πsafe(10,000), the number of safe primes not exceeding 10,000\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_.**

The following is a deterministic algorithm to determine primality. It is not practical for large numbers n.

**int Is\_Prime(int n)**

{

//returns the value 1 if n is a prime, the value 0 otherwise.

for(int d = 2; d\*d <=n; ++n) if(n%d == 0) return 0;

return 1;

}

**Remarks**

We divide n by the numbers 2, 3, 4, … , n1/2.

If none of these divide n without a remainder, then n is a prime.

**Lemma**

If n is composite, then n has a divisor d < n1/2.

**Proof**

Suppose n is composite. Then n = dd’, where d, d’ > 1.

But then Min {d, d’} < n1/2. For, suppose that Min{d, d’} > n1/2.

Then n = dd’ > n1/2n1/2 = n, a contradiction. Qed.

The following is an algorithm for π(x) that uses the function Is\_Prime(n).

**π(x)**

**int Pi(double x)**

{

Returns the value of the number of primes not exceeding x.

int count = 1; //2 is the only even prime.

for(int j = 3; j <=x; j = j+2) If (Is\_Prime(j) ) ++count;

return count;

}

**Safe Primes**

The following algorithm uses Is\_Prime(n) to determine whether a number is a safe prime.

The first few safe primes are 5, 7, 11, …

**int Is\_Safe\_Prime(int n)**

{

int flag = Is\_Prime(n);

if( ! flag) return 0; //if n isn’t a prime, then it isn’t a safe prime.

flag = Is\_Prime( (n – 1)/2 );

if( ! flag)return 0; //if (n-1)/2 isn’t prime, then n is an unsafe prime.

return 1;

}

**Number of Primes p ≡ 1 (mod 4) π1,4(x)**

The following algorithm counts the number of Pythagorean primes. These are the primes that are congruent to 1 modulo 4.

The first few are 5, 13, 17, 29, …

int P14(double x)

{

int count = 0;

for( int j = 5; j <=x; j = j+4)

if(Is\_Prime(j)) ++count;

return count;

}

**Remark**

It is known that π1,4(x) ~ ½x/loge x.

So half of the primes are congruent to 1 mod 4. The others are congruent to 3 mod 4.

Thus, we also know that π3,4(x) ~ ½x/loge x.

**Number of Safe Primes**

The following algorithm finds the number of safe primes not exceeding x by using the function Is\_Safe\_Prime(n).

**πsafe(x)**

int Pi\_Safe(double x)

{

//This will count the number of safe primes, beginning with 7.

int count = 0;

for(int j = 5; j <= x; j = j + 2)

if(Is\_Safe\_Prime(j) ) ++count;

return count;

}

Example: πsafe(11) = |{5, 7, 11}| = 3.

**Remark**

Whereas π(x) ~ x/loge x, it is conjectured that πsafe(x) ~ Cx/(loge x)2 for some constant C > 0.