//Bitwise Sieve

#define mx 2147483700

int prm[(mx/32)+5];

bool Check(int N,int pos){return (bool)(N & (1<<pos));}

int Set(int N,int pos){ return N=N | (1<<pos) ;}

void BWsieve(int N)

{

int i, j, sqrtN;

sqrtN = int( sqrt( N ) );

for( i = 3; i <= sqrtN; i += 2 )

{

if( Check(prm[i>>5],i&31)==0)

{

for( j = i\*i; j <= N; j += (i<<1) )

{

prm[j>>5]=Set(prm[j>>5],j & 31) ;

}

}

}

}

Usage :

input

if( input is even ) Not prime

if( input is odd )

{

if( Check(status[input>>5],input&31 ) == 0 ) Prime

else

Not Prime

}

bool isprime(int input)

{

if(input&1)

{

if( Check(prm[input>>5],input&31 ) == 0 )

return true;

else

return false;

}

else

{

return false;

}

}

//Binary GCD

int gcd(int a, int b)

{

while(b) b ^= a ^= b ^= a %= b;

return a;

}

//EXTENDED EUCLID

int xGCD(int a, int b, int &x, int &y) {

if(b == 0) {

x = 1;

y = 0;

return a;

}

int x1, y1, gcd = xGCD(b, a % b, x1, y1);

x = y1;

y = x1 - (a / b) \* y1;

return gcd;

}