```
// Test pierwszości - test Millera-Rabina
// www.algorytm.org
// Tomasz Lubinski (c) 2007

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

int powerOf2[31] =
{ 1<<0,  1<<1,  1<<2,  1<<3,  1<<4,  1<<5,  1<<6,
  1<<7,  1<<8,  1<<9,  1<<10, 1<<11, 1<<12, 1<<13,
  1<<14, 1<<15, 1<<16, 1<<17, 1<<18, 1<<19, 1<<20,
  1<<21, 1<<22, 1<<23, 1<<24, 1<<25, 1<<26, 1<<27,
  1<<28, 1<<29, 1<<30 };

// calculates a^b mod m
int power_modulo_fast(int a, int b, int m)
{
   int i;
   int result = 1;
   long int x = a%m;

   for (i=1; i<=b; i<<=1)
   {
      x %= m;
      if ((b&i) != 0)
      {
         result *= x;
         result %= m;
      }
      x *= x;
   }

   return result;
}

//Miller-Rabin test
int Miller_Rabin(int n, int k)
{
   int s = 0;
   int s2 = 1;
   int a, d, i, r, prime;
   srand(time(NULL));

   if (n<4)
   {
      return 1;
   }
   if (n%2 == 0)
```

```
    {
        return 0;
    }

    // calculate s and d
    while ((s2 & (n-1)) == 0)
    {
        s  += 1;
        s2 <<= 1;
    }
    d = n/s2;

    // try k times
    for (i=0; i<k; i++)
    {
        a = 1+(int) ((n-1)*rand()/(RAND_MAX+1.0));
        if (power_modulo_fast(a, d, n) != 1)
        {
            prime = 0;
            for (r=0; r<=s-1; r++)
            {
                if (power_modulo_fast(a, power0f2[r]*d, n) == n - 1)
                {
                    prime = 1;
                    break;
                }
            }
            if (prime == 0)
            {
                return 0;
            }
        }
    }

    return 1;
}

int main()
{
    int n, k;

    printf("Podaj liczbe do sprawdzenia.\n");
    scanf("%d", &n);

    printf("Podaj dokladnosc sprawdzenia.\n");
    scanf("%d", &k);

    if (Miller_Rabin(n, k) == 1)
```

```
    {
        printf("Liczba jest prawdopodobnie pierwsza.\n");
    }
    else
    {
        printf("Liczba jest zlozona.\n");
    }

    return 0;
}
```