#include <stdio.h>

void multiply(int F[2][2], int M[2][2]);

void power(int F[2][2], int n);

/\* function that returns nth Fibonacci number \*/

int fib(int n)

{

  int F[2][2] = {{1,1},{1,0}};

  if (n == 0)

    return 0;

  power(F, n-1);

  return F[0][0];

}

/\* Optimized version of power() in method 4 \*/

void power(int F[2][2], int n)

{

  if( n == 0 || n == 1)

      return;

  int M[2][2] = {{1,1},{1,0}};

  power(F, n/2);

  multiply(F, F);

  if (n%2 != 0)

     multiply(F, M);

}

void multiply(int F[2][2], int M[2][2])

{

  int x =  F[0][0]\*M[0][0] + F[0][1]\*M[1][0];

  int y =  F[0][0]\*M[0][1] + F[0][1]\*M[1][1];

  int z =  F[1][0]\*M[0][0] + F[1][1]\*M[1][0];

  int w =  F[1][0]\*M[0][1] + F[1][1]\*M[1][1];

  F[0][0] = x;

  F[0][1] = y;

  F[1][0] = z;

  F[1][1] = w;

}

/\* Driver program to test above function \*/

int main()

{

  int n = 9;

  printf("%d", fib(9));

  getchar();

  return 0;

}

/\*From GeeksForGeeks\*/

Resourse link : <http://www.geeksforgeeks.org/program-for-nth-fibonacci-number/>

Sometimes you will find it hard to generate larger Fibonacci numbers within the time limit . It is the most efficient algorithm.