// fib1.cpp

```
// A recursive function for a function having one parameter that
// generates the nth Fibonacci number.
// f(i+2)=fi+f(i+1)
#include <iostream>
#include <cmath>
using namespace std;
// The full recursive version:
unsigned long Fib1( int n );
int main()
{
char ans;
int N;
do
{
cout << "I will display fibonacci numbers 0-N." << endl;
cout << "Enter an limit, please. Be patient! This recursive"
<< endl << "Fibonacci routine will take about 17 "
<< endl << "seconds for N = 45"
             " alone" << endl;
cin >> N;
for ( int i = 0; i < N; i++ )
cout << Fib1(i) << endl;
cout << "Y/y to continue, anything else quits" << endl;
cin >> ans;
} while ( 'Y' == ans || 'y' == ans );
}

unsigned long Fib1( int n )
{
if (n == 0 || n == 1)
return 1;
return Fib1( n - 1 ) + Fib1( n - 2 );
}
```

output

```
I will display fibonacci numbers 0-N.
Enter an limit, please. Be patient! This recursive
Fibonacci routine will take about 17
seconds for N = 35 alone
45
1
1
2
3
5
8
13
21
34
55
89
144
233
377
610
987
1597
2584
4181
6765
10946
17711
28657
46368
75025
121393
196418
317811
514229
832040
1346269
2178309
3524578
5702887
9227465
14930352
24157817
39088169
63245986
102334155
165580141
267914296
433494437
701408733
1134903170
Y/y to continue, anything else quits
```

```
// fib2.cpp
#include <iostream>
#include <cmath>
using namespace std;
// The full recursive version:
unsigned long Fib2( int n );
int main()
{
char ans;
int N;
do
{
cout << "I will display fibonacci numbers 0-N." << endl;
cout << "Enter an limit, please. Be patient! This recursive"
<< endl << "Fibonacci routine will take about 2 "
<< endl << "seconds for N = 47"
             " alone" << endl;
cin >> N;
for ( int i = 0; i < N; i++ )
cout << Fib2(i) << endl;
cout << "Y/y to continue, anything else quits" << endl;
cin >> ans;
} while ( 'Y' == ans || 'y' == ans );
return 0;
}


unsigned long Fib2(int n)
{
  /* Declare an array to store fibonacci numbers. */
  int f[n+1];
  int i;

  /* 0th and 1st number of the series are 1 and 1*/
  f[0] = 0;
  f[1] = 1;

  for (i = 2; i <= n; i++)
  {
      /* Add the previous 2 numbers in the series
         and store it */
      f[i] = f[i-1] + f[i-2];
  }

  return f[n];
}
```

Output

```
I will display fibonacci numbers 0-N.
Enter an limit, please. Be patient! This recursive
Fibonacci routine will take about 2
seconds for N = 45 alone
47
0
1
1
2
3
5
8
13
21
34
55
89
144
233
377
610
987
1597
2584
4181
6765
10946
17711
28657
46368
75025
121393
196418
317811
514229
832040
1346269
2178309
3524578
5702887
9227465
14930352
24157817
39088169
63245986
102334155
165580141
267914296
433494437
701408733
1134903170
1836311903
Y/y to continue, anything else quits
```