МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ КИЇВСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМЕНІ ТАРАСА ШЕВЧЕНКА

ЗВІТ

про виконання самостійної роботи №1

«Дослідження**​** ефективності роботи різних алгоритмів формування чисел Фібоначчі**​**»

З дисципліни «Теорія алгоритмів»

Варіант ─ 14

Виконав студент

групи КН-22

Пашковський П.В.

Викладач:

Доманецька І. М.

**Перелік алгоритмів:**

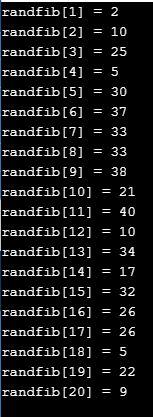
1)Циклічне обчислення за класичною формулою **(1​.1)**

2)Рекурсивне обчислення чисел Фібоначчі за класичною формулою з використанням допоміжного масиву для зберігання вже згенерованих чисел. **(2​.2)**

3)Обчислення чисел Фібоначчі з використанням значення “золотого перерізу”. **(3​.2)**

4)Матричний спосіб обчислення чисел Фібоначчі. **​(4.1)**

**Масив randfib:**

****

**Таблиця значень:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Згенеровані | Метод 1 | Метод 2 | Метод 3 | Метод 3 (int) | Метод 4 |
| числа |  |  | (double) |  |  |
|  |  |  |  |  |  |
| 2 | 1 | 1 | 1.67 | 1 |  |
|  |  |  |  |  |  |
| 10 | 55 | 55 | 55.5 | 55 | 55 |
|  |  |  |  |  |  |
| 25 | 75025 | 75025 | 7.5e+04 | 75025 | 75025 |
|  |  |  |  |  |  |
| 5 | 5 | 5 | 5.46 | 5 | 5 |
|  |  |  |  |  |  |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 30 | 832040 | 832040 | 8.32e+05 | 832040 | 832040 |
|  |  |  |  |  |  |
| 37 | 24157817 | 24157817 | 2.42e+07 | 24157817 | 24157817 |
|  |  |  |  |  |  |
| 33 | 3524578 | 3524578 | 3.52e+06 | 3524578 | 3524578 |
|  |  |  |  |  |  |
| 33 | 3524578 | 3524578 | 3.52e+06 | 3524578 | 3524578 |
|  |  |  |  |  |  |
| 38 | 39088169 | 39088169 | 3.91e+07 | 39088169 | 39088169 |
|  |  |  |  |  |  |
| 21 | 10496 | 10496 | 1.09e+04 | 10496 | 10496 |
|  |  |  |  |  |  |
| 40 | 102334155 | 102334155 | 1.02e+08 | 102334155 | 102334155 |
|  |  |  |  |  |  |
| 10 | 55 | 55 | 55.5 | 55 | 55 |
|  |  |  |  |  |  |
| 34 | 5702887 | 5702887 | 5.7e+06 | 5702887 | 5702887 |
|  |  |  |  |  |  |
| 17 | 1597 | 1597 | 1.6e+03 | 1597 | 1597 |
|  |  |  |  |  |  |
| 32 | 2178309 | 2178309 | 2.18e+06 | 2178309 | 2178309 |
|  |  |  |  |  |  |
| 26 | 121393 | 121393 | 1.21e+05 | 121393 | 121393 |
|  |  |  |  |  |  |
| 26 | 121393 | 121393 | 1.21e+05 | 121393 | 121393 |
|  |  |  |  |  |  |
| 5 | 5 | 5 | 5.46 | 5 | 5 |
|  |  |  |  |  |  |
| 22 | 17711 | 17711 | 1.77+04 | 17711 | 17711 |
|  |  |  |  |  |  |
| 9 | 34 | 34 | 34.5 | 34 | 34 |
|  |  |  |  |  |  |

**Таблиця часу:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Метод 1 | Метод 2 | Метод | Метод 3 | Метод 4 |
|  |  | 3(double) | (int) |  |
|  |  |  |  |  |
| 76 | 87 | 162 | 75 | 77 |
|  |  |  |  |  |

**Таблиця часу 1-ого методу з різними типами даних:**

|  |  |  |
| --- | --- | --- |
| int | long int | double |
|  |  |  |
| 76 | 80 | 245 |
|  |  |  |

**Висновки:**

Дослідивши роботу різних методів обчислення чисел Фібоначчі, виявилося, що найшвидшим методом є третій, що обчислює числа Фібоначчі за значенням “золотого перерізу” у цілочисельному форматі. Матричний та циклічний методи теж досить ефективні.

Зазвичай рекурсивний метод є найповільнішим, але у даному випадку показує середній результат завдяки допоміжному масиву. Третій метод з дійсними числами працює найповільніше, до того ж з похибкою.

**Код програми:**

#include <iostream>

#include <cmath>

#include <iomanip>

#include <ctime>

using namespace std;

int help2[47];

long int fibonacci1(int n){

long int a[n];

int i;

a[0]=0;

a[1]=1;

for (i=2;i<=n;i++)

{

a[i] = a[i-1]+a[i-2];

}

return a[n];

}

int fibonacci2(int i){

if (i == 0) return 0;

if (i == 1) return 1;

if (help2[i] > 0) return help2[i];

help2[i] = fibonacci2(i-1) + fibonacci2(i-2); return help2[i];

}

double fibonacci3(int n){

double fib, p=sqrt (5);

double f = (1+p)/2;

fib = pow(f,n)/p + 0.5;

return fib;

}

int fibonacci4(int n)

{

int a = 1, ta,

b = 1, tb,

c = 1, res = 0, tc,

d = 0, rd = 1;

while (n)

{

if (n & 1)

{

tc = res;

res = res\*a + rd\*c;

rd = tc\*b + rd\*d;

}

ta = a; tb = b; tc = c;

a = a\*a + b\*c;

b = ta\*b + b\*d;

c = c\*ta + d\*c;

d = tc\*tb+ d\*d;

n /= 2;

}

return res;

}

int main(){

int i;

for (i = 0; i < 47; i++) {

help2[i] = 0;

}

int randfib[20];

srand(time(NULL));

for (i = 0; i < 20; i++) {

randfib[i] = rand() % 47;

cout << "randfib["<< i+1 << "] = " << randfib[i] << endl;

}

cout<<endl;

int stime1=clock();

for (i = 0; i < 20; i++) {

cout <<"1 method["<<i<<"]: "<<fibonacci1(randfib[i])<<endl;

}

int etime1=clock();

cout<<endl<<"time spent on the first method: "<<etime1-stime1<<endl;

int stime2 = clock();

for (i = 0; i < 47; i++) {

help2[i] = 0;

}

for (i = 0; i < 20; i++) {

cout <<"2 method["<<i<<"]: "<<fibonacci2(randfib[i])<<endl;

}

int etime2 = clock();

cout<<endl<<"time spent on the second method: "<<etime2-stime2<<endl;

int stime3 = clock();

for (i = 0; i < 20; i++) {

cout <<"3 method["<<i<<"]:

"<<setprecision(3)<<fibonacci3(randfib[i])<<endl;

}

int etime3=clock();

cout<<endl<<"time spent on the third method: "<<etime3-stime3<<endl; int stime = clock();

for (i = 0; i < 20; i++) {

cout <<"3 method["<<i<<"]: "<<setprecision(3)<<int (fibonacci3(randfib[i]))<<endl;

}

int etime=clock();

cout<<endl<<"time spent on the third method with integral numbers:

"<<etime-stime<<endl;

int stime4 = clock();

for (i = 0; i < 20; i++) {

cout <<"4 method["<<i<<"]: "<<fibonacci4(randfib[i])<<endl;

}

int etime4=clock();

cout<<endl<<"time spent on the fourth method: "<<etime4-stime4<<endl;

cout<<endl;

return 0;

}