1. **Užduotis. Fibonačio sekos savybės**

**Fibonačio skaičių seka** – sveikųjų skaičių seka, nusakoma taip: F1 = 1, F2 = 1, Fn = Fn-2 + Fn-1. Seka prasideda šiais skaičiais: 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144, 233. Kiekvienas šios sekos skaičius lygus dviejų prieš jį einančių skaičių sumai.

Knygoje „Matematikos pasaulyje“ pateikiamos tokios Fibonačio sekos savybės:

1) sekos kiekvienas trečias skaičius yra lyginis;

2) sekos kiekvienas ketvirtas skaičius dalijasi iš 3;

3) sekos kiekvienas penktas skaičius dalijasi iš 5;

4) sekos kiekvienas penkioliktas skaičius dalijasi iš 10.

Parašykite programą, kuri patikrintų n (3 ≤ n ≤ 93) narių Fibonačio seką ir išspausdintų tuos narius, kurie atitinka aukščiau nurodytas savybes.

Pradinių duomenų faile **fibonati\_data.txt** įrašyta **n** reikšmė. Rezultatų faile **fibonati\_res.txt** turi būti įrašyti Fibonačio sekos nariai, tenkinantys užduotyje nurodytas sekos savybes: nario reikšmė, kelintas tai yra sekos narys, kurią savybę tenkina.

|  |  |
| --- | --- |
| fibonati\_data.txt | fibonati\_res.txt |
| 13 | 2 3 1  3 4 2  5 5 3  8 6 1  21 8 2  34 9 1  55 10 3  144 12 1  144 12 2 |

1. **Reikalavimai**

* Parašykite funkciją **unsigned long long Narys(unsigned long long f1, unsigned long long f2)**, grąžinančią naują Fibonačio sekos narį.
* Parašykite funkciją **int Savybe(unsigned long long narys, int kelintas)**, grąžinančią, kurią savybę atitinka Fibonačio sekos narys. Kadangi vienas sekos narys gali atitikti kelias savybes, ši funkcija turi nebūtinai grąžinti tikslią savybę. Ji turėtų grąžinti tokį skaičių, iš kurio eitų sužinoti, kurias savybes atitinka narys.
* Programoje gali būti daugiau funkcijų, grąžinančių apskaičiuotą reikšmę per funkcijos vardą.

1. **Programos tekstas C++ kalba**

#include <iostream>

#include <iomanip>

#include <cmath>

#include <fstream>

using namespace std;

unsigned long long Narys(unsigned long long f1, unsigned long long f2);

int Savybe(unsigned long long narys, int kelintas);

int main(){

int n, n1, pir = 1, antr = 1, temp, real;

ifstream df ("fibonati\_data.txt");

ofstream rf ("fibonati\_res.txt");

df >> n;

n1 = n;

n = n - 2;

while (n > 0){

temp = Narys(pir, antr);

antr = pir;

pir = temp;

n = n - 1;

real = Savybe(temp, n1 - n);

while (real > 0){

if (Savybe(temp, n1 - n))rf << temp << " " << n1 - n << " " << real % 10 << endl;

real = real / 10;

}

}

rf.close();

df.close();

return 0;

}

unsigned long long Narys(unsigned long long f1, unsigned long long f2)

{

return f1 + f2;

}

int Savybe(unsigned long long narys, int kelintas)

{

int indeks = 0, sk = 0;

if (kelintas % 3 == 0){

sk = sk + 1 \* pow(10, indeks);

indeks++;

}

if (kelintas % 4 == 0){

sk = sk + 2 \* pow(10, indeks);

indeks++;

}

if (kelintas % 5 == 0){

sk = sk + 3 \* pow(10, indeks);

indeks++;

}

if (kelintas % 15 == 0){

sk = sk + 4 \* pow(10, indeks);

indeks++;

}

if (sk > 0)return sk;

else return false;

}