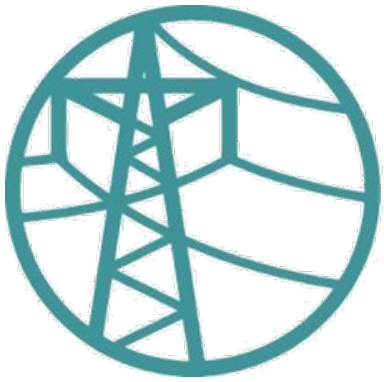

WYDZIAŁ
ELEKTROTECHNIKI
I INFORMATYKI
POLITECHNIKI RZESZOWSKIEJ

Michał Kruczek

Sprawozdanie z projektu nr 1

Spis treści

1 Wstęp

Treść zadania jest następująca:

Znajdź liczbę trójelementowych kombinacji liczb z zadanego ciągu,których suma jest równa zadanej liczbie M.

Przykład:

Wejście : [1,2,5,1,2,1,2,4]

M = 6

Wyjście: Liczba kombinacji wynosi 2: [2,2,2] , [1,1,4]

1.1. Opis problemu

Problem zadania projektowego polega na odnalezieniu w ciągu liczb podciągu trójelementowego równego zadanej liczbie M. Ciąg użyty w zadaniu jest wprowadzany ręcznie przy każdym uruchomieniu programu. Można również użyć do tego generatora liczb pseudolosowych,tworzącego nową tablice wraz z rozpoczęciem pracy programu.

2 Opis podstaw teoretycznych zagadnienia

Rozwiązanie zadania polegało na napisaniu algorytmu takiego,żeby potrafił on pogrupować elementy tablicy w trójki,dodać je ze sobą, porównać do podanej liczby M oraz wyświetlić ilość takich trójelementowych kombinacji.

3 Opis szczegółów implementacji problemu

3.1. Biblioteki

iostream	Deklaruje obiekty kontrolują odczytywanie ze strumieni standardowych i zapisywanie ich w tych strumieniach. Jest to często jedyny nagłówek potrzebny do wprowadzania danych i danych wyjściowych z programu w języku C++.
fstream	Definiuje kilka klas, które obsługują operacje iostream na sekwencjach przechowywanych w plikach zewnętrznych.
chrono	Dołącza standardowy nagłówek, aby zdefiniować klasy i funkcje, które reprezentują czasy trwania i czasy natychmiastowe oraz manipulują nimi.

3.2. Zmienne

zapis	Służy do zapisu danych do pliku zewnętrznego
n	Przechowuje rozmiar tablicy ciągu
start	Przechowuje czas na początku działania programu
finish	Przechowuje czas na końcu działania programu
elapsed	Przechowuje czas, który upłynął przez okres działania programu
A[n]	Przechowuje elementy tablicy o rozmiarze n
M	Przechowuje liczbę ,do której przyrównywana jest suma wyrazów trójelementowego podciągu
a,i,k,j	Liczniki pętli for
ilosc	Zawiera ilosc kombinacji trzech wyrazów tablicy równych liczbie M
obecnyIndeks	Zmienna pomocnicza,przechowująca indeks danego licznika pętli

3.3. Funkcje

main	Funkcja programu, zawiera inicjalizacje i deklaracje zmiennych, funkcje związane z obliczaniem czasu obliczeń oraz zapisywaniem danych do pliku; w niej wywoływane są pozostałe funkcje,za pomocą zagnieżdżonych pętli for znajduje ona powtarzające się elementy tablicy, a wyniki działań zapisuje do pliku
SprawdzCzyKombinacjaIstnieje	Funkcja,która sprawdza wynik,będący efektem działania funkcji main
DodajAktualnaKombinacje	Funkcja,która usuwa duplikaty znalezionych w ciągu kombinacji

4 Pseudokod programu

c=100

B[c]

indeks=0

liczba kombinacji=0

n=8

A[n]

ilosc=0

dla i=1 do n wykonuj

Wczytaj A[n]

dla i=1 do n wykonuj

Wypisz A[n]

Wczytaj M

dla i=1 do n,dla j=i+1 do n oraz dla k=j+1 do n wykonuj

jeżeli indeks=0,oraz kombinacja A[i],A[j],A[k] istnieje dla M

wyprowadź [A[i],A[j],A[k]]

wyprowadź ilość kombinacji wynosi M w przeciwnym razie

wyprowadź ilość kombinacji wynosi 0

5 Kod programu

```
#include <iostream>
#include <fstream>
#include <chrono>
using namespace std;
//zapisywanie elementow tablicy
//deklarowanie zmiennych globalnych
//stała wartość c
const int c = 100;
//tablica pomocnicza
int B[c];
//zmienne pomocnicze
int obecnyIndeks = 0;
//deklaracja funkcji
bool SprawdzCzyKombinacjaIstnieje(int a, int j, int k);
void DodajAktualnaKombinacje(int a, int j, int k);
int main()
{
   // czas rozpoczęcia pracy programu
autostart=chrono::high_resolution_clock::now();
   ofstream zapis("wyniki.txt");
   //rozmiar głównej tablicy
   const int n = 8;
   //inicjalizacja zmiennych
   int M, ilosc = 0;
   //inicjalizacja tablicy
   int A[n];
   cout << "podaj elementy tablicy " << endl;
   for (int i = 0; i < n; i++)
   {
      cin >> A[i];//wpisywanie liczb do tablicy
   }
   cout << "Tablica A: " << endl;
   for (int i = 0; i < n; i++)
   {
        cout << A[i] << " ";//wyswietlenie liczb tablicy
   }
   cout << endl;
    cout << "Podaj liczbe M: " << endl;//podanie liczby M
   cin >> M;
   for (int i = 0; i < n; i++)
   {
      for (int j = i + 1; j < n; j++)
      {
         for (int k = j + 1; k < n; k++)
         {
  {if (A[i] + A[j] + A[k] == M)
            {
                        if (obecnyIndeks == 0 || ! SprawdzCzyKombinacjaIstnieje(A[i], A[j], A[k])) {
                     zapis <<"["<< A[i] << "," << A[j] << "," << A[k] <<"]"<< endl; //wyświetlenie znalezionych kombinacji
                ilosc++;
                 DodajAktualnaKombinacje(A[i], A[j], A[k]);
               }
            }
```

```
        }
      }
    }
    zapis <<"Ilosc kombinacji wynosi: "<< ilosc << endl;//Podanie ilości znalezionych kombinacji
    cout <<"Wyniki zapisano w pliku tekstowym wyniki.txt"<<endl;
    zapis.close ();
    // czas zakończenia pracy programu
                auto finish = std::chrono::high_resolution_clock::now();
     chrono::duration<double> elapsed = finish - start;  //obliczenie czasu pracy programu
     cout << "Czas trwania: " << elapsed.count() << " s\n"; //wyświetlenie czasu pracy programu
}
bool SprawdzCzyKombinacjaIstnieje(int a, int j, int k) { //funkcja pomocnicza - sprawdza czy dana kombinacja występuje w ciągu
    for (int i = 0; i < obecnyIndeks;i++) {
      if (i + 1 < obecnyIndeks)
      {
        if (B[i] == a && B[i + 1] == j && B[i + 2] == k)
          return true;
      }
    }
    return false;
}
void DodajAktualnaKombinacje(int a, int j, int k) {
    B[obecnyIndeks] = a;
    B[obecnyIndeks + 1] = j;
    B[obecnyIndeks + 2] = k;
    obecnyIndeks += 3;
}
```

6 Złożoność obliczeniowa

Ukazany wyżej algorytm posiada złożoność rzędu O (n^3) . Wynika to z potrójnie zagnieżdżonej pętli for wykonującej powstawanie kolejnych trójelementowych kombinacji.

7 Podsumowanie

Powyższy algorytm rozwiązuje problem podany w treści zadania. Ukazuje on składające się z trzech wyrazów podciągi równe podanej przez użytkownika liczbie M. Co prawda,jest on dosyć powolny ze względu na złożoność rzędu O (n^3),jednak spełnia wymagania niezbędne do prawidłowego funkcjonowania.

8 Netografia

https://www.samouczekprogramisty.pl/podstawy-zlozonosci-obliczeniowej/

https://docs.microsoft.com/pl-pl/cpp/standard-library/cpp-standard-library-header-files?view=msvc-160

https://home.agh.edu.pl/~pamalino/programowanie/cpp/index.php

https://eduinf.waw.pl/inf/alg/001_search/index.php

http://pa.prz.edu.pl/index.php?page=md0