1. **Užduotis.** **Detalės**

Pradinių duomenų failo **detales\_data.txt** pirmoje eilutėje įrašytas bendrovės sandėlyje esančių detalių rūšių kiekis **n** (n < 1000). Likusiose **n** eilučių įrašyti duomenys apie kiekvieną detalę: triženklis detalės kodas (sveikasis skaičius), tos rūšies detalių skaičius (sveikasis skaičius), detalės vieneto kaina (realusis skaičius) ir brokuotų detalių kiekis (sveikasis skaičius).

Parašykite programą, skaičiuojančią:

* kiek yra kiekvienos rūšies nebrokuotų detalių;
* kiek kainavo kiekvienos rūšies nebrokuotos detalės;
* kiek iš viso buvo brokuotų detalių;
* kiek kainavo visos detalės (įskaitant ir brokuotas);
* kiek procentų nuostolių patirs bendrovė dėl brokuotų detalių.

Skaičiavimų rezultatus išveskite į rezultatų failą **detales\_res.txt**. Rezultatų faile turi būti: pirmosiose **n** eilučių: pradiniai duomenys, nebrokuotų detalių skaičius, nebrokuotų detalių kaina, visų detalių kaina (įskaitant ir brokuotas), po to išvedamas bendras brokuotų detalių skaičius, pinigų suma, kurią kainavo visos detalės ir kiek procentų nuostolių patirs bendrovė dviejų ženklų po kablelio tikslumu.

|  |  |
| --- | --- |
| detales\_data.txt | detales\_res.txt |
| 3  123 12 1.20 5  213 15 2.30 9  415 14 3.21 2 | 123 12 1.20 5 7 8.40 14.40  213 15 2.30 9 6 13.80 34.50  415 14 3.21 2 12 38.52 44.94  16  93.84  35.29 |

1. **Reikalavimai**

* Naudokite sveikųjų ir realiųjų skaičių masyvus: sveikųjų – kiekiams ir kodams, realiųjų – detalių kainoms saugoti.
* Sukurkite pradinių duomenų skaitymo funkciją **void**. Joje į masyvus sudėkite detalių kodus, kiekius (visų ir brokuotų) ir detalės vieneto kainas.
* Sukurkite funkciją **void**, kuri suformuotų: nebrokuotų detalių masyvą (sveikųjų skaičių masyvas) bei nebrokuotų ir visų detalių kainų masyvus (realiųjų skaičių).
* Sukurkite detalių kainų sumos skaičiavimo funkciją, grąžinančią apskaičiuotą sumą per funkcijos vardą. Funkciją panaudokite atlikdami d ir e užduotis.
* Sukurkite brokuotų detalių kiekio skaičiavimo funkciją, grąžinančią apskaičiuotą kiekį per funkcijos vardą. Funkciją panaudokite atlikdami c užduotį.
* Rezultatų išvedimui į failą parašykite funkciją **void**.

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

#include <iostream>

#include <iomanip>

#include <cmath>

#include <fstream>

using namespace std;

void Skaito (int &n, int A[], int B[], double C[], int D[]);

void Formuoja(int n, int A[], int B[], int C[], double D[], double E[], double F[]);

void Raso(int n, int A[], int B[], double C[], int D[], int E[], double G[], double F[], int vis, double sum, double nuost);

int Detsum(int n, int A[]);

double Detkain(int n, double A[]);

int main(){

int A[100], B[100], D[100], E[100], n, N[100];

double C[100], F[100], G[100];

Skaito(n, A, B, C, D);

Formuoja(n, B, D, E, C, F, G);

Raso(n, A, B, C, D, E, G, F, Detsum(n, D), Detkain(n, F), (Detkain(n, F) - Detkain(n, G)) / Detkain(n, F) \* 100);

return 0;

}

void Skaito (int &n, int A[], int B[], double C[], int D[])

{

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

df >> n;

for (int i = 0; i < n; i++)df >> A[i] >> B[i] >> C[i] >> D[i];

df.close();

}

void Formuoja(int n, int A[], int B[], int C[], double D[], double E[], double F[])

{

for (int i = 0; i < n; i++){

C[i] = A[i] - B[i];

E[i] = D[i] \* A[i];

F[i] = C[i] \* D[i];

}

}

int Detsum(int n, int A[])

{

int s = 0;

for (int i = 0; i < n; i++)s = s + A[i];

return s;

}

void Raso(int n, int A[], int B[], double C[], int D[], int E[], double G[], double F[], int vis, double sum, double nuost)

{

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

for (int i = 0; i < n; i++)rf << A[i] << " " << B[i] << " " << fixed << setprecision(2) << C[i] << " " << D[i] << " " << E[i] << " " << fixed << setprecision(2) << G[i] << " " << F[i] << endl;

rf << vis << endl << fixed << setprecision(2) << sum << endl << nuost;

rf.close();

}

double Detkain(int n, double A[])

{

double s = 0;

for (int i = 0; i < n; i++)s = s + A[i];

return s;

}