**ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ**

**НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИ УНИВЕРСИТЕТ «ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»**

Факультет компьютерных наук

Департамент программной инженерии

Практические приемы построения многопоточных приложений. Вариант 18

**Исполнитель**

Студент группы БПИ192

Раджбхандари А. Д.

**Задание**

Задача о наследстве. У старого дона Энрике было два сына, у каждого из сыновей – еще по два сына, каждый из которых имел еще по два сына. Умирая, дон Энрике завещал все свое богатство правнукам в разных долях. Адвокат дон Хосе выполнил задачу дележа наследства в меру своих способностей. Правнуки заподозрили адвоката в укрывательстве части наследства. Требуется создать многопоточное приложение, которое при известных сумме завещания дона Энрике и доле каждого наследника, проверяет честность адвоката. При решении использовать принцип дихотомии.

Составление​​ программы.

Сначала считываем пользовательский ввод общей суммы денег, долей наследства каждого правнука и полученную каждым правнуком сумму.

Затем параллельно суммируются доли всех сыновей, проверяем корректность суммы долей всех сыновей (если сумма больше 1, то сумма некорректна). Сравниваем в цикле для каждого внука полученную сумму с долей. Поскольку порядок проверки долей у сыновей не влияет на результат, удобнее проверять доли сыновей последовательно обычным циклом, вместо принципа дихотомии.

Текст программы:

#include <iostream>

#include <vector>

#include <string>

#include <omp.h>

using namespace std;

void Enter(vector<double>& childrenProp, vector<double>& childrenSum, double& totalSum)

{

string line;

do {

for (int i = 0; i < 1; i++) {

cout << "Введите общую сумму денег (больше нуля)" << endl;

cin >> line;

try {

totalSum = stoi(line);

}

catch (const exception&) {

i--;

}

}

} while (totalSum <= 0);

double currentFraction = 0;

for (int i = 0; i < 8; i++) {

cout << "Введите долю сына №" << i + 1 << " (Число от 0 до 1)" << endl;

cin >> line;

try {

currentFraction = stod(line);

}

catch (const exception&) {

i--;

}

if (currentFraction > 1 || currentFraction < 0) {

cout << "Доля не может быть больше 1 или меньше 0!" << endl;

i--;

currentFraction = 0;

}

else {

childrenProp.push\_back(currentFraction);

currentFraction = 0;

}

}

int sum = 0;

for (int i = 0; i < 8; i++) {

cout << "Введиет сумму, которую получил сын №" << i + 1 << endl;

cin >> line;

try {

sum = stod(line);

}

catch (const exception&) {

i--;

}

if (sum < 0) {

cout << "Сумма должна быть не меньше нуля!" << endl;

i--;

sum = 0;

}

else {

childrenSum.push\_back(sum);

sum = 0;

}

}

}

int main()

{

setlocale(LC\_CTYPE, "rus");

//Доли правнуков

vector<double> GreatGrandSonsFractions;

//Деньги, которые получили правнуки

vector<double> GreatGrandSonsSums;

bool check = true;

//Все доли

double AllFractions = 0;

//Общая сумма, полученных денег

double total = 0;

//Пользовательский ввод

Enter(GreatGrandSonsFractions, GreatGrandSonsSums, total);

#pragma omp parallel for

for (int i = 0; i < 8; i++)

{

//Суммируем все доли

AllFractions += GreatGrandSonsFractions[i];

}

if (AllFractions > 1) {

//Если сумма всех долей больше единицы, была допущена ошибка

cout << "Сумма всех долей не может быть больше единицы (Правнуки не могут получить больше денег, чем было всего у Дона Энрике!)" << endl;

return 0;

}

#pragma omp parallel for

for (int j = 0; j < 8; j++)

{

if (GreatGrandSonsFractions[j] \* GreatGrandSonsSums[j] != total) {

cout << "Адвокат ошибся" << endl;

cout << "Правнук № " << j + 1 << " должен был получить " << GreatGrandSonsFractions[j] \* total << ", а получил: " << GreatGrandSonsSums[j] << endl;

}

else {

cout << "Правнук № " << j + 1 << " получил правильную сумму!" << endl;

}

}

return 0;

}