README.md

Кирюхин Андрей, БПИ191

Четвертое домашнее задание по Архитектуре вычислительных систем. Вариант 18. Исходники находятся в папке code.

Выполнение задания

Условие:

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.

Разработка:

Для решения была выбрана следующая модель построения многопоточных приложений: **Рекурсивный параллелизм**(*используется в программах с одной или несколькими рекурсивными процедурами, вызов которых независим. Это технология «разделяй-и-властвуй» или «перебор-с-возвратами».*)
После консультации с преподавателем, было решено генерировать доли наследников и размер наследства случайно внутри программы (в условии не было четких требований касательно этого).

- **Основная рекурсивная функция**

```
//Рекурсивная функция для формирования древа семьи для подсчета суммы полученной наследниками от Дона
void funcTree(int depth, int leftInt, int rightInt) {
    //depth = 3, мы дошли до 8 правнуков, останавливаем вызов новых функций дальше
    if (depth == 3) {
        //Параллельный подсчет суммы
        double result = percentages[leftInt] *
                        inheritance; //На данной глубине концы интервалов равны, поэтому берем одно из крайних значений
#pragma omp critical
        {
            if (result - lawyerCalcInheritance[leftInt] > 0.001) {
                cout << "The lawyer deceived the " << leftInt + 1 << " heir for "
                     << result - lawyerCalcInheritance[leftInt] << "$" << endl;
            } else if (lawyerCalcInheritance[leftInt] - result > 0.01) {
                cout << "The lawyer gave " << lawyerCalcInheritance[leftInt] - result << "$ more to " << leftInt + 1
                     << " heir" << endl;
            } else {
                cout << "The lawyer correctly counted the share of " << leftInt + 1 << " heir" << endl;
            }
        }
    } else {
        //Передача в функцию интервалов, каждый раз разделяя их пополам
        int intLen = (rightInt - leftInt) / 2;
#pragma omp task shared(intLen, depth, leftInt) default(none)
        funcTree( depth: depth + 1, leftInt,  rightInt: leftInt + intLen);
#pragma omp task shared(intLen, depth, rightInt) default(none)
        funcTree( depth: depth + 1,  leftInt: rightInt - intLen, rightInt);
    }
}
```

В **main** в отдельном потоке запускается данная функция с параметрами *leftInt* = 0, *rightInt* = 7 (крайние значения индексов в

массиве долей наследников).
Затем интервал делится на два и от них уже вызываются два раза данная рекурсивная функция. Такое деление происходит пока глубина рекурсии не равна трём, затем для каждого наследника считается полученная сумма и передается в родительскую функцию (так происходит для всех узлов). В итоге в первой вызванной рекурсивной функции (с параметрами 0,7) будет значение общей суммы, которую получили наследники.

- **Функция генерации долей наследников**

```
/*
    Генерация процентных долей наследников на основе "завещания" Дона.
*/
void generatePercentage() {
    percentages = new double[8];

    double sum = 1;
    double tmp = (rand() % 20) / 100.0 + 0.05;
    percentages[0] = tmp;

    sum -= tmp;
    for (int i = 1; i < 7; i++) {
        tmp = (rand() % (int) (sum * 60)) / 100.0;
        sum -= tmp;
        percentages[i] = tmp;
    }
    percentages[7] = sum;
}
```

- **Функция генерации результатов распределения наследства адвокатом**

```
/*
    Генерация долей, посчитанных адвокатом (в этот момент он может обмануть наследников)
 */
void generateLawyerCalculatedInheritance() {

    lawyerCalcInheritance = new double[8];
    bool isLawyerLying = rand() % 3 == 0;
    if(!isLawyerLying){
        for (int i = 0; i < 8; i++) {
            lawyerCalcInheritance[i] = percentages[i] * inheritance;
        }
    } else {
        // Если адвокат решит обмануть наследников, он может с вероятностью 1/3 забирать у них 1/6 наследства,
        // либо он может перепутает проценты между наследниками
        if (rand() % 2 == 0) {
            for (int i = 0; i < 8; i++) {
                if (rand() % 3 == 0) {
                    lawyerCalcInheritance[i] = (percentages[i] / 6.0) * inheritance;
                } else {
                    lawyerCalcInheritance[i] = percentages[i] * inheritance;
                }
            }
        } else {
            for (int i = 0; i < 8; i++) {
                lawyerCalcInheritance[i] = percentages[7 - i] * inheritance;
            }
        }
    }
}
```

- **Точка входа в программу**

```
int main() {
    srand( _Seed: static_cast<unsigned int>(time( _Time: 0)));

    inheritance = rand() % 500000 + 500000;

    cout << "Don left an inheritance of " << inheritance << "$" << endl;

    generatePercentage();
    cout << "Don's will contained the following percentage distributions of inheritance among heirs:" << endl;
    for (int i = 0; i < 7; i++) {
        cout << percentages[i] * 100 << "%, ";
    }
    cout << percentages[7] * 100 << "%" << endl << endl;

    generateLawyerCalculatedInheritance();
    cout << "According to the lawyer's calculations, heirs should receive:" << endl;
    for (int i = 0; i < 7; i++) {
        cout << lawyerCalcInheritance[i] << "$, ";
    }
    cout << lawyerCalcInheritance[7] << "$" << endl << endl;

    //Подсчет суммы, которую наследники получили на руки, и её проверка
#pragma omp parallel default(none)
    {
#pragma omp single nowait
        funcTree( depth: 0,  leftInt: 0,  rightInt: 7);
    }
}
```

Тестирование программы

В данном блоке проведен ряд тестов программы. И так как все параметры (размер наследства, доли) генерируются случайно внутри программы, то продемонстрирую работоспособность программы при разных сценариях.

- **Адвокат обманул некоторых наследников**

```
Don left an inheritance of 510243$
Don's will contained the following percentage distributions of inheritance among heirs:
11%, 13%, 30%, 15%, 8%, 9%, 2%, 12%

According to the lawyer's calculations, heirs should receive:
9354.45$, 66331.6$, 153073$, 12756.1$, 40819.4$, 45921.9$, 10204.9$, 61229.2$

The lawyer deceived the 1 heir for 46772.3$
The lawyer correctly counted the share of 5 heir
The lawyer correctly counted the share of 3 heir
The lawyer deceived the 4 heir for 63780.4$
The lawyer correctly counted the share of 2 heir
The lawyer correctly counted the share of 7 heir
The lawyer correctly counted the share of 8 heir
The lawyer correctly counted the share of 6 heir

Process finished with exit code 0
```

- **Адвокат перепутал проценты между наследниками (дал кому-то больше, кому-то меньше)**

```
Don left an inheritance of 517045$
Don's will contained the following percentage distributions of inheritance among heirs:
17%, 23%, 0%, 34%, 12%, 3%, 2%, 9%

According to the lawyer's calculations, heirs should receive:
46534$, 10340.9$, 15511.3$, 62045.4$, 175795$, 0$, 118920$, 87897.7$

The lawyer deceived the 1 heir for 41363.6$
The lawyer deceived the 6 heir for 15511.3$
The lawyer gave 113750$ more to 5 heir
The lawyer deceived the 4 heir for 113750$
The lawyer gave 41363.6$ more to 8 heir
The lawyer gave 108579$ more to 7 heir
The lawyer gave 15511.3$ more to 3 heir
The lawyer deceived the 2 heir for 108579$

Process finished with exit code 0
```

- **Адвокат все правильно посчитал**

```
Don left an inheritance of 517179$
Don's will contained the following percentage distributions of inheritance among heirs:
10%, 36%, 28%, 2%, 8%, 8%, 2%, 6%

According to the lawyer's calculations, heirs should receive:
51717.9$, 186184$, 144810$, 10343.6$, 41374.3$, 41374.3$, 10343.6$, 31030.7$

The lawyer correctly counted the share of 1 heir
The lawyer correctly counted the share of 3 heir
The lawyer correctly counted the share of 5 heir
The lawyer correctly counted the share of 4 heir
The lawyer correctly counted the share of 7 heir
The lawyer correctly counted the share of 2 heir
The lawyer correctly counted the share of 8 heir
The lawyer correctly counted the share of 6 heir

Process finished with exit code 0
```

Таким образом была успешно разработана и протестирована данная программа.