

Пояснительная записка к домашнему заданию №4
**Многопоточное приложение,
реализующее задачу Пути Кулака**

НИУ ВШЭ, ДПИ
Тарасюк Инна Валерьевна, БПИ192(1)
Вариант 24

1 Условие задания

Задача о Пути Кулака. На седых склонах Гималаев стоят два древних буддистских монастыря: Гуань-Инь и Гуань-Янь. Каждый год в день сошествия на землю боддисатвы Араватти монахи обоих монастырей собираются на совместное празднество и показывают свое совершенствование на Пути Кулака. Всех соревнующихся монахов разбивают на пары, победители пар бьются затем между собой и так далее, до финального поединка. Монастырь, монах которого победил в финальном бою, забирает себе на хранение статую боддисатвы. Реализовать многопоточное приложение, определяющего победителя. В качестве входных данных используется массив, в котором хранится количество энергии Ци каждого монаха. При решении использовать принцип дихотомии.

2 Описание работы программы

2.1 Выбор модели построения многопоточного приложения

Модель построения основана на параллельных секциях в OpenMP. Каждая секция выполняется в отдельном потоке, в каждой из которых происходит вызов функции `threadFunction` (см. 3.3. алгоритм) для одной половины исходной команды. Точкой синхронизации является конец блока `sections`.

3 Работа программы

3.1 Переменные

1. `n` - количество жителей двух монастырей; значение задаётся пользователем;
2. `max_energy` - максимально допустимое значение энергии Ци; задаётся пользователем;
3. `team` - массив размера `n`, состоящий из жителей двух монастырей.

3.2 Функции

Название функции	Тип возвращаемого значения	Входные параметры
<code>main</code>	<code>int</code>	-
<code>input</code>	<code>int</code>	-
<code>threadFunction</code>	<code>vector<int></code>	<code>team</code> , <code>int l</code> , <code>int v</code>

1. `main()`

2. `int input()`
3. `void threadFunction(std::vector<int> team, int l, int r)`

3.3 Алгоритм

Программе на вход подается число `n` и `max_energy`, осуществляется их проверка на корректность с помощью функции `input()`. Создаётся массив размера `n`, который заполняется случайными числами в диапазоне `[1; max_energy]` - значения энергии Ци. Параллельно вызываются функции `threadFunction` для каждой половины исходной команды. Функция `threadFunction` определяет, у какого из двух подряд идущих монахов энергия больше. Затем все проигравшие в парах монахи удаляются из исходного набора. Победившие делятся на две команды, снова параллельно вызываются `threadFunction()` для каждой половины исходной команды. Процесс продолжается до тех пор, пока количество оставшихся в наборе монахов не меньше 2. Сведения о всех сражавшихся монахах и победителе среди них выводятся на экран.

4 Входные данные

1. `n` - размер массива для энергий монахов;
2. `max_energy` - максимально возможная энергия монаха.

4.1 Ограничения

Представленные ниже ограничения были введены для удобства и ввиду здравого смысла.

Переменная	Минимальное значение	Максимальное значение
<code>n</code> (количество монахов)	2	40
<code>max_energy</code>	1	-

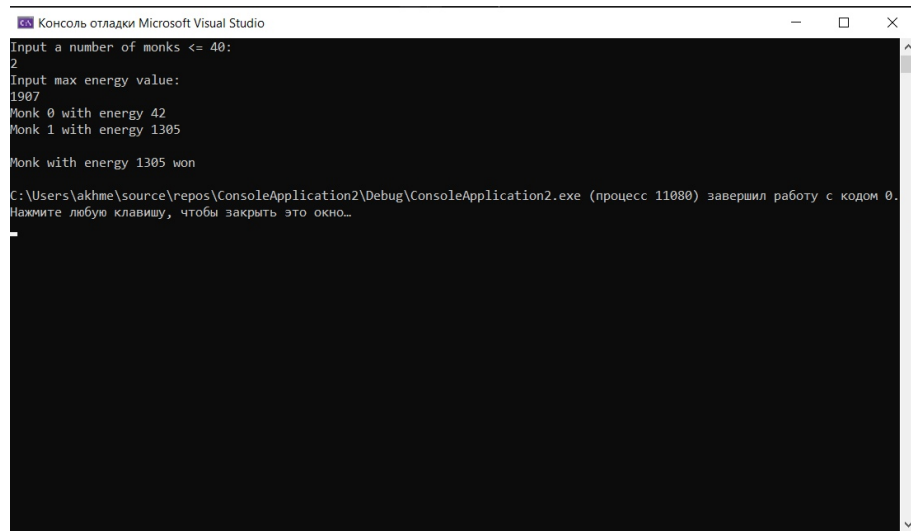
5 Тестирование программы

5.1 Некорректные данные

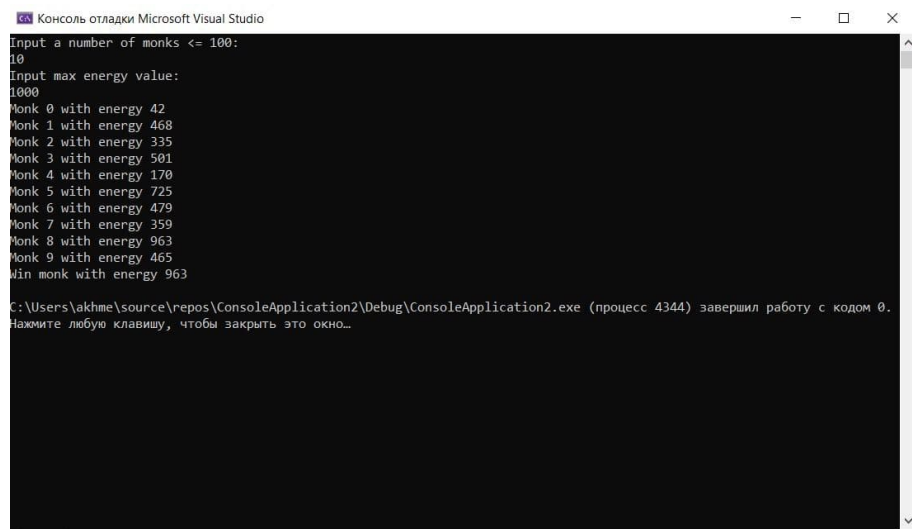
```
Консоль отладки Microsoft Visual Studio
Input a number of monks <= 40:
1
Input a number of monks <= 40:
41
Input a number of monks <= 40:
40
Input max energy value:
12
Monk 0 with energy 6
Monk 1 with energy 12
Monk 2 with energy 11
Monk 3 with energy 5
Monk 4 with energy 6
Monk 5 with energy 5
Monk 6 with energy 7
Monk 7 with energy 7
Monk 8 with energy 11
Monk 9 with energy 9
Monk 10 with energy 6
Monk 11 with energy 6
Monk 12 with energy 2
Monk 13 with energy 4
Monk 14 with energy 2
Monk 15 with energy 12
Monk 16 with energy 8
Monk 17 with energy 3
Monk 18 with energy 4
Monk 19 with energy 1
Monk 20 with energy 4
Monk 21 with energy 1
Monk 22 with energy 3
Monk 23 with energy 10
Monk 24 with energy 5
Monk 25 with energy 11
Monk 26 with energy 10
Monk 27 with energy 9
Monk 28 with energy 3
Monk 29 with energy 12
Monk 30 with energy 12
Monk 31 with energy 7
Monk 32 with energy 12
Monk 33 with energy 7
Monk 34 with energy 10
Monk 35 with energy 5
Monk 36 with energy 12
Monk 37 with energy 8
Monk 38 with energy 8
Monk 39 with energy 7
Monk with energy 12 won
```

```
Консоль отладки Microsoft Visual Studio
Input a number of monks <= 40:
rn
Incorrect input. Try again.
Input a number of monks <= 40:
-1
Input a number of monks <= 40:
5
Input max energy value:
max
Incorrect input. Try again.
Input max energy value:
100
Monk 0 with energy 42
Monk 1 with energy 68
Monk 2 with energy 35
Monk 3 with energy 1
Monk 4 with energy 70
Monk with energy 70 won
C:\Users\akhme\source\repos\ConsoleApplication2\Debug\ConsoleApplication2.exe (процесс 17308) завершил работу с кодом 0.
Нажмите любую клавишу, чтобы закрыть это окно...
```

5.2 Корректные данные



```
Консоль отладки Microsoft Visual Studio
Input a number of monks <= 40:
2
Input max energy value:
1907
Monk 0 with energy 42
Monk 1 with energy 1305
Monk with energy 1305 won
C:\Users\akhme\source\repos\ConsoleApplication2\Debug\ConsoleApplication2.exe (процесс 11080) завершил работу с кодом 0.
Нажмите любую клавишу, чтобы закрыть это окно...
```



```
Консоль отладки Microsoft Visual Studio
Input a number of monks <= 100:
10
Input max energy value:
1000
Monk 0 with energy 42
Monk 1 with energy 468
Monk 2 with energy 335
Monk 3 with energy 501
Monk 4 with energy 170
Monk 5 with energy 725
Monk 6 with energy 479
Monk 7 with energy 359
Monk 8 with energy 963
Monk 9 with energy 465
Win monk with energy 963
C:\Users\akhme\source\repos\ConsoleApplication2\Debug\ConsoleApplication2.exe (процесс 4344) завершил работу с кодом 0.
Нажмите любую клавишу, чтобы закрыть это окно...
```

6 Текст программы

```
#include <iostream>
#include <omp.h>
#include <chrono>
#include <vector>
```

```

#include <algorithm>

using std::string;
using std::vector;

int input() {
    int n;
    std::cin >> n;
    bool test = true;
    do {
        if (!(test = std::cin.good())) {
            std::cout << "Incorrect input. Try again." << std::endl;
            std::cin.clear();
            std::cin.ignore(std::numeric_limits<std::streamsize>::max(), '\n');
        }
    } while (!test);
    return n;
}

void threadFunction(vector<int>& team, int l, int r) {
    for (int i = l; i < r - 1; i += 2) {
        (team[i] < team[i + 1]) ? team[i] = 0 : team[i + 1] = 0;
    }
}

int main() {
    int n;
    int max_energy;
    do {
        std::cout << "Input a number of monks <= 40:" << std::endl;
        n = input();
    } while (n <= 1 || n > 40);
    do {
        std::cout << "Input max energy value: " << std::endl;
        max_energy = input();
    } while (max_energy <= 0);

    vector<int> team;
    for (size_t i = 0; i < n; ++i) {
        team.push_back(rand() % max_energy + 1);
        std::cout << "Monk " << i << " with energy " << team[i] << std::endl;
    }

    while (n / 2 + n % 2 >= 2) {

```

```

#pragma omp parallel
{
#pragma omp sections
{
#pragma omp section
{
threadFunction(team, 0, n / 2);
}
#pragma omp section
{
threadFunction(team, n / 2, n);
}
}
}

int a = 0;
team.erase(std::remove(team.begin(), team.end(), a), team.end())
n = team.size();

}
(team[0] > team[1]) ? std::cout << "\nMonk with energy " << team[0] << "
<< std::endl : std::cout << "\nMonk with energy " << team[1] <<

return 0;
}

```

Список литературы

1. информация о потоках <https://habr.com/ru/post/279653/>
2. информация об openmp <http://ccfit.nsu.ru/arom/data/openmp.pdf>
3. информация о дихотомии <https://habr.com/ru/company/otus/blog/504310/>
4. информация о дихотомии <http://www.machinelearning.ru/wiki/index.php?title=>
5. <http://softcraft.ru/>