Пояснительная записка

НИУ ВШЭ, ФКН

Образовательная программа “Программная инженерия”, 2 курс

Курс «Архитектура вычислительных систем»

Разработка многопоточных приложений с использованием OpenMP

Вариант **21**

Стегнина Варвара Валерьевна

Группа БПИ196

2020 г.

Задание

Вариант 21. Задача про экзамен. Преподаватель проводит экзамен у группы студентов. Каждый студент заранее знает свой билет и готовит по нему ответ. Подготовив ответ, он передает его преподавателю. Преподаватель просматривает ответ и сообщает студенту оценку. Требуется создать многопоточное приложение, моделирующее действия преподавателя и студентов. При решении использовать парадигму «клиент-сервер».

Описание метода решения

Все студенты и преподаватель находятся в разных потоках. В начале все студенты приступают к решению экзамена и говорят об этом. Студенты решают экзамен с разной скоростью: у них уходит на это от 5 до 20 секунд. После решения всех задач они становятся в очередь на проверку. Преподаватель проверяет работы, находящиеся в очереди, перед проверкой он говорит, какую работу начал проверять. На работу у него уходит от 1 до 3 секунд. После проверки работы преподаватель ставит оценку в список. Ожидающие студенты проверяют список и если находят свою оценку, то говорят о том, что ознакомились с оценкой и завершают экзамен.

Запись в консоль, работа с очередью и списком происходят в критических секциях. В начале программы создаётся <Количество студентов> + 1 поток, поток с номером 0 – преподаватель, остальные – студенты. Преподаватель ожидает появления работ в очереди, пока не проверит <Количество студентов> работ. Студенты ожидают появления оценки в списке.

Входные и выходные данные

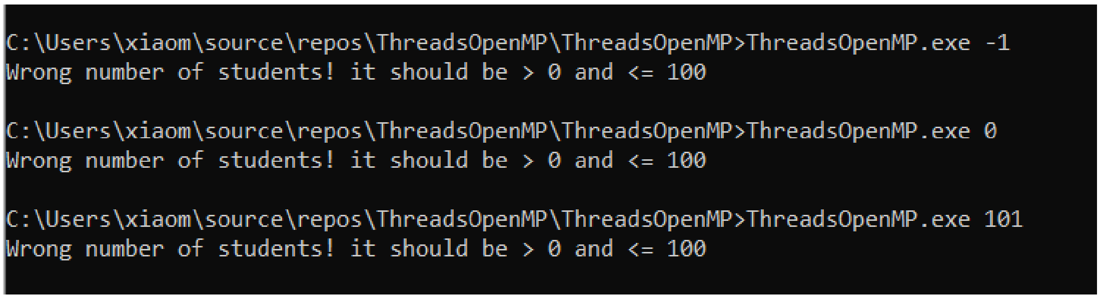
Аргумент командной строки – число студентов. Ограничения на число студентов N: 0 < N <= 100. В консоль выводится сообщения от преподавателя и студентов.

Формат командной строки: ThreadsOpenMP.exe <Число студентов>

Пример: ExamThreads.exe 10

Тестирование

1) -1
2) 0
3) 101 – неверные аргументы



```
C:\Users\xiaom\source\repos\ThreadsOpenMP\ThreadsOpenMP>ThreadsOpenMP.exe -1
Wrong number of students! it should be > 0 and <= 100

C:\Users\xiaom\source\repos\ThreadsOpenMP\ThreadsOpenMP>ThreadsOpenMP.exe 0
Wrong number of students! it should be > 0 and <= 100

C:\Users\xiaom\source\repos\ThreadsOpenMP\ThreadsOpenMP>ThreadsOpenMP.exe 101
Wrong number of students! it should be > 0 and <= 100
```

4) 1

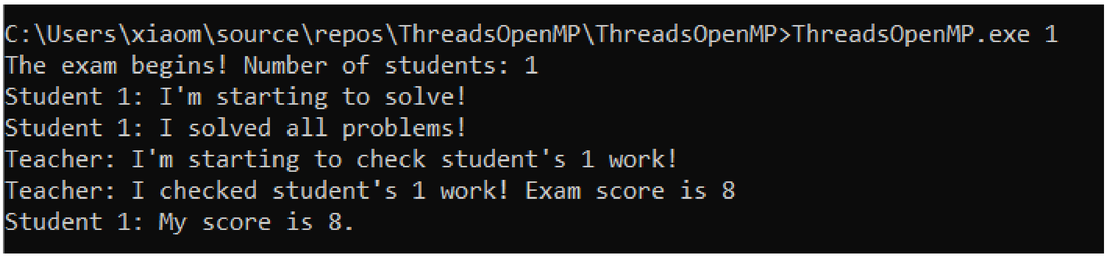
Студент начинает решать экзамен

Студент заканчивает решать экзамен

Преподаватель начинает проверку

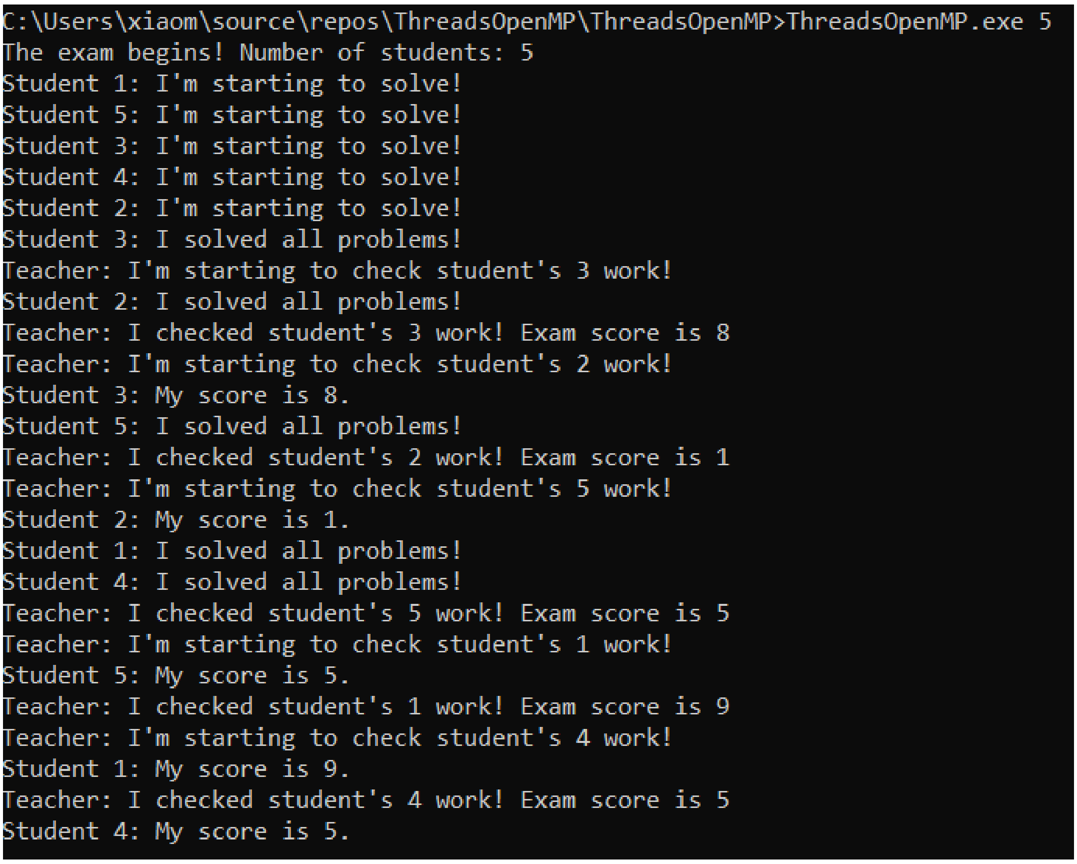
Преподаватель заканчивает проверку

Студент смотрит оценку



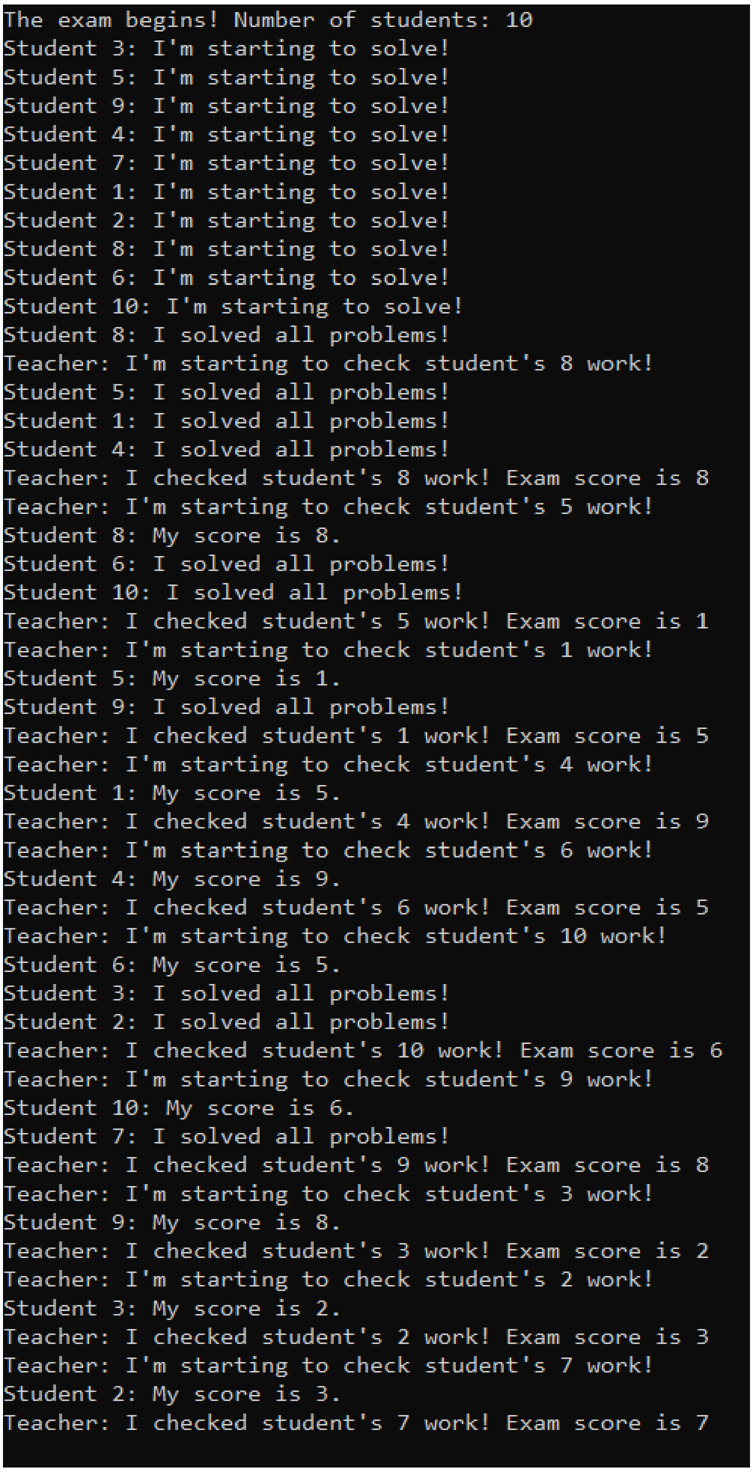
```
C:\Users\xiaom\source\repos\ThreadsOpenMP\ThreadsOpenMP>ThreadsOpenMP.exe 1
The exam begins! Number of students: 1
Student 1: I'm starting to solve!
Student 1: I solved all problems!
Teacher: I'm starting to check student's 1 work!
Teacher: I checked student's 1 work! Exam score is 8
Student 1: My score is 8.
```

5) 5



```
C:\Users\xiaom\source\repos\ThreadsOpenMP\ThreadsOpenMP>ThreadsOpenMP.exe 5
The exam begins! Number of students: 5
Student 1: I'm starting to solve!
Student 5: I'm starting to solve!
Student 3: I'm starting to solve!
Student 4: I'm starting to solve!
Student 2: I'm starting to solve!
Student 3: I solved all problems!
Teacher: I'm starting to check student's 3 work!
Student 2: I solved all problems!
Teacher: I checked student's 3 work! Exam score is 8
Teacher: I'm starting to check student's 2 work!
Student 3: My score is 8.
Student 5: I solved all problems!
Teacher: I checked student's 2 work! Exam score is 1
Teacher: I'm starting to check student's 5 work!
Student 2: My score is 1.
Student 1: I solved all problems!
Student 4: I solved all problems!
Teacher: I checked student's 5 work! Exam score is 5
Teacher: I'm starting to check student's 1 work!
Student 5: My score is 5.
Teacher: I checked student's 1 work! Exam score is 9
Teacher: I'm starting to check student's 4 work!
Student 1: My score is 9.
Teacher: I checked student's 4 work! Exam score is 5
Student 4: My score is 5.
```

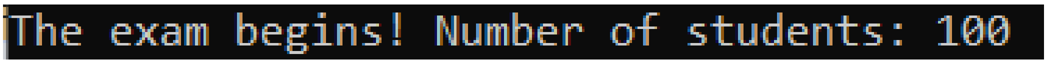
6) 10



```
The exam begins! Number of students: 10
Student 3: I'm starting to solve!
Student 5: I'm starting to solve!
Student 9: I'm starting to solve!
Student 4: I'm starting to solve!
Student 7: I'm starting to solve!
Student 1: I'm starting to solve!
Student 2: I'm starting to solve!
Student 8: I'm starting to solve!
Student 6: I'm starting to solve!
Student 10: I'm starting to solve!
Student 8: I solved all problems!
Teacher: I'm starting to check student's 8 work!
Student 5: I solved all problems!
Student 1: I solved all problems!
Student 4: I solved all problems!
Teacher: I checked student's 8 work! Exam score is 8
Teacher: I'm starting to check student's 5 work!
Student 8: My score is 8.
Student 6: I solved all problems!
Student 10: I solved all problems!
Teacher: I checked student's 5 work! Exam score is 1
Teacher: I'm starting to check student's 1 work!
Student 5: My score is 1.
Student 9: I solved all problems!
Teacher: I checked student's 1 work! Exam score is 5
Teacher: I'm starting to check student's 4 work!
Student 1: My score is 5.
Teacher: I checked student's 4 work! Exam score is 9
Teacher: I'm starting to check student's 6 work!
Student 4: My score is 9.
Teacher: I checked student's 6 work! Exam score is 5
Teacher: I'm starting to check student's 10 work!
Student 6: My score is 5.
Student 3: I solved all problems!
Student 2: I solved all problems!
Teacher: I checked student's 10 work! Exam score is 6
Teacher: I'm starting to check student's 9 work!
Student 10: My score is 6.
Student 7: I solved all problems!
Teacher: I checked student's 9 work! Exam score is 8
Teacher: I'm starting to check student's 3 work!
Student 9: My score is 8.
Teacher: I checked student's 3 work! Exam score is 2
Teacher: I'm starting to check student's 2 work!
Student 3: My score is 2.
Teacher: I checked student's 2 work! Exam score is 3
Teacher: I'm starting to check student's 7 work!
Student 2: My score is 3.
Teacher: I checked student's 7 work! Exam score is 7
```

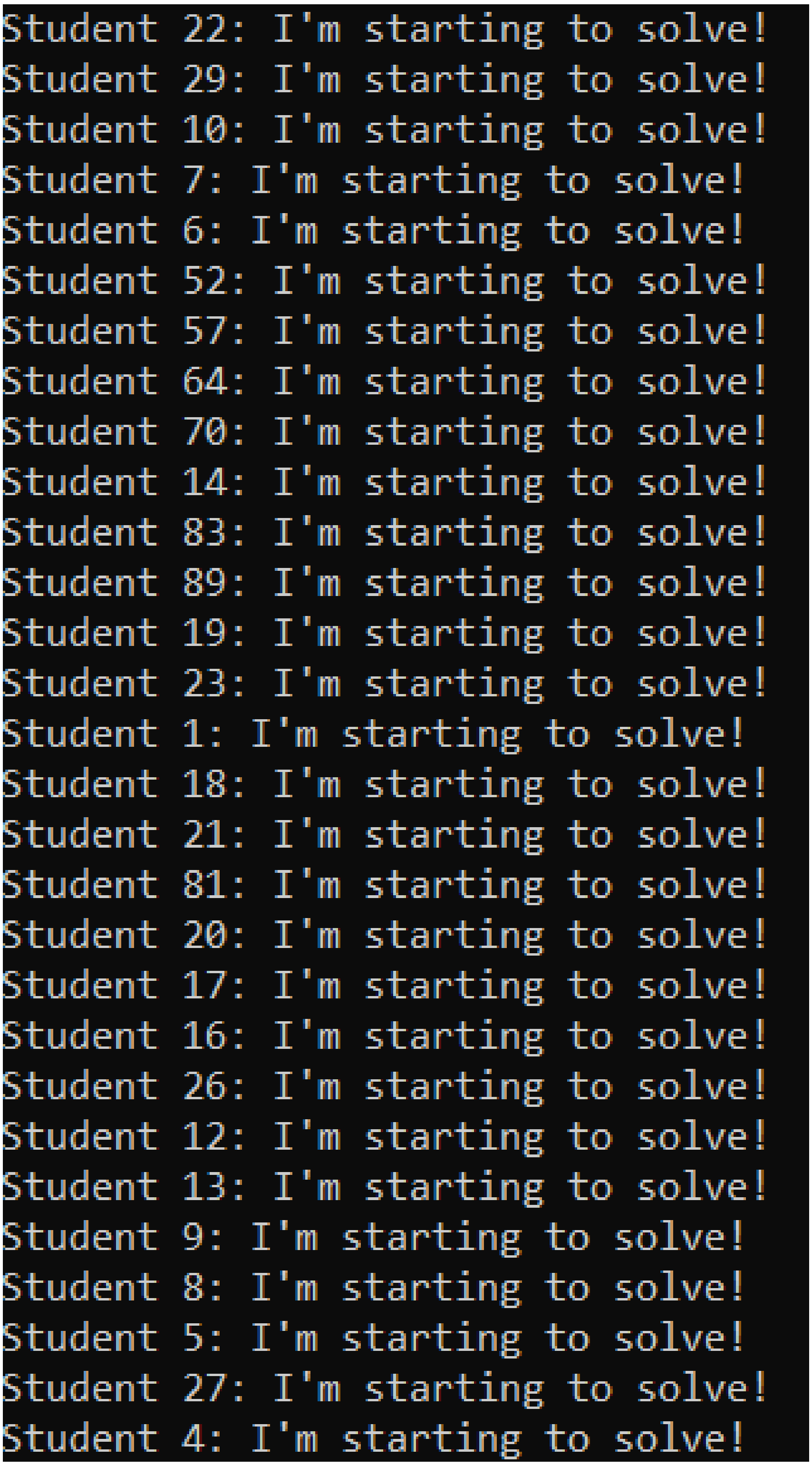
7) 100

Начало:



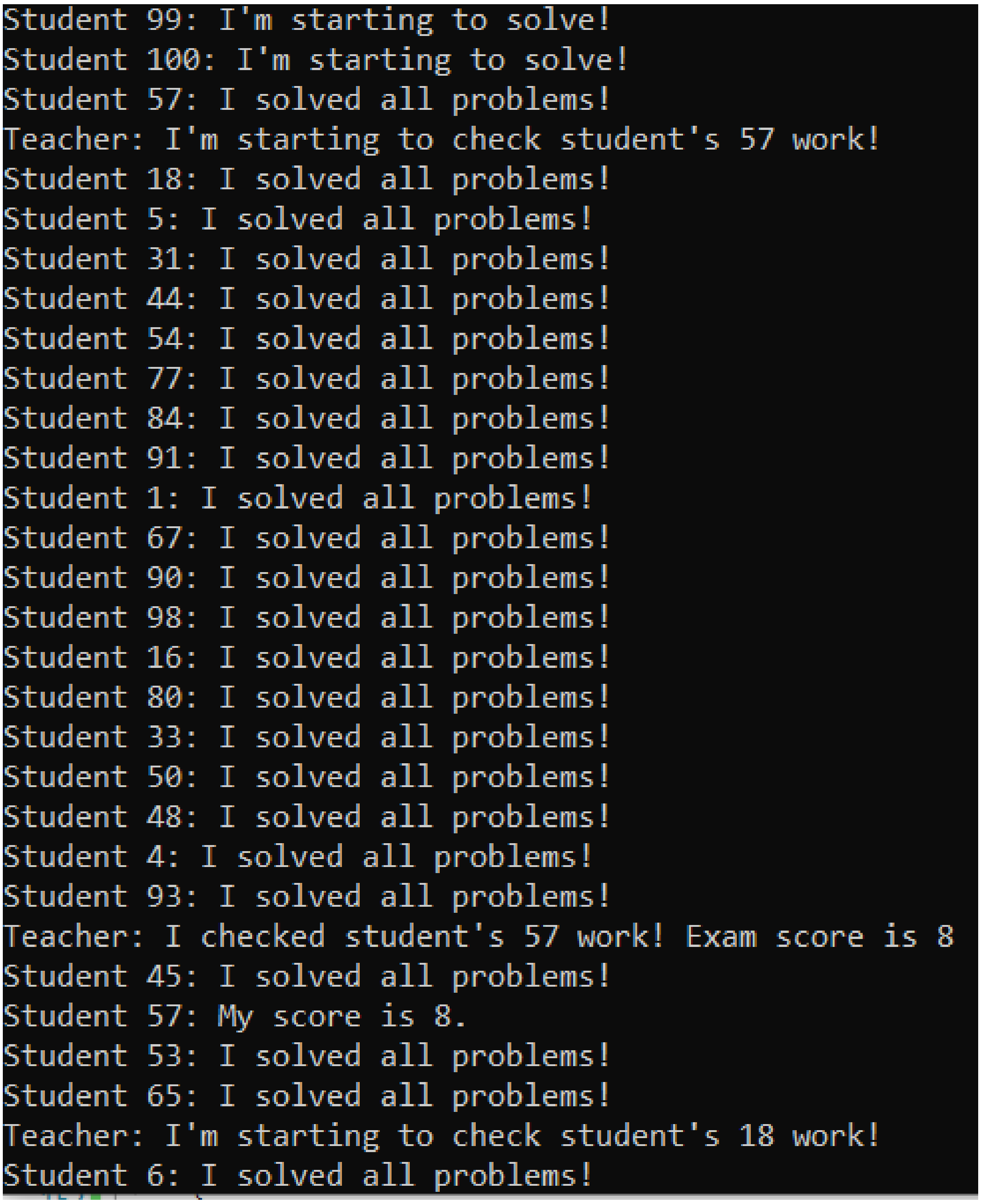
```
The exam begins! Number of students: 100
```

Студенты начинают работу:



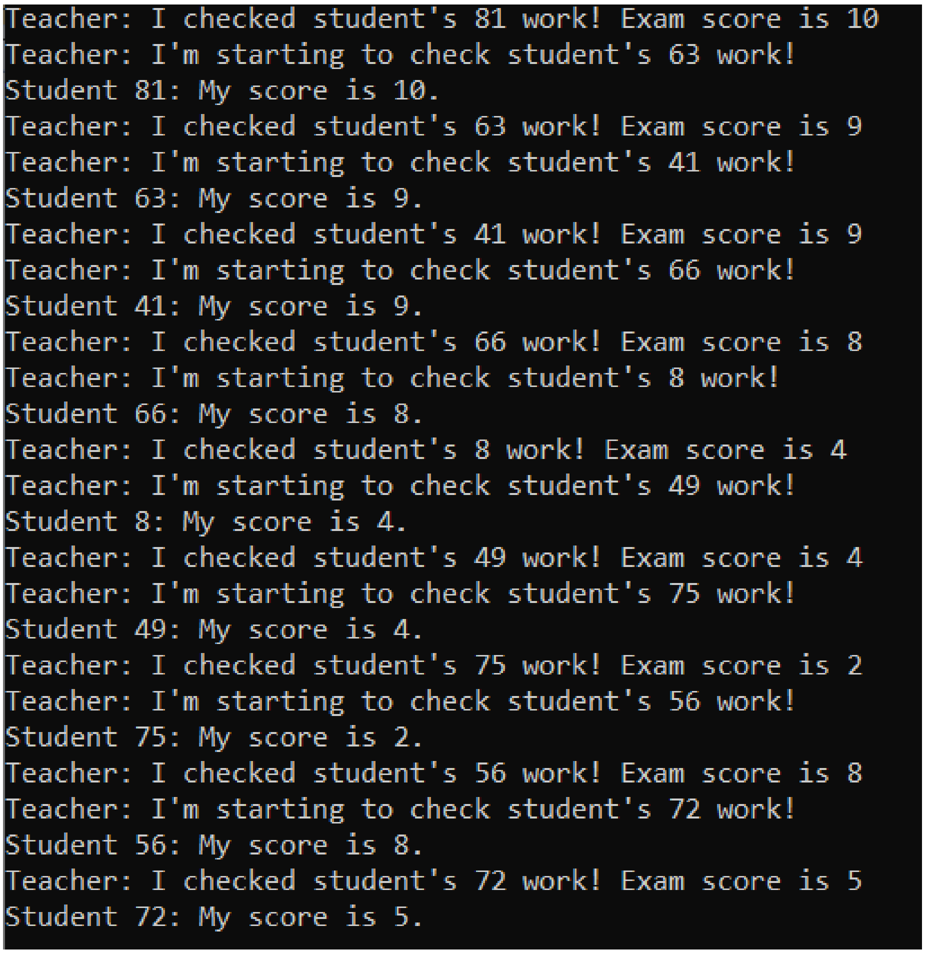
```
Student 22: I'm starting to solve!
Student 29: I'm starting to solve!
Student 10: I'm starting to solve!
Student 7: I'm starting to solve!
Student 6: I'm starting to solve!
Student 52: I'm starting to solve!
Student 57: I'm starting to solve!
Student 64: I'm starting to solve!
Student 70: I'm starting to solve!
Student 14: I'm starting to solve!
Student 83: I'm starting to solve!
Student 89: I'm starting to solve!
Student 19: I'm starting to solve!
Student 23: I'm starting to solve!
Student 1: I'm starting to solve!
Student 18: I'm starting to solve!
Student 21: I'm starting to solve!
Student 81: I'm starting to solve!
Student 20: I'm starting to solve!
Student 17: I'm starting to solve!
Student 16: I'm starting to solve!
Student 26: I'm starting to solve!
Student 12: I'm starting to solve!
Student 13: I'm starting to solve!
Student 9: I'm starting to solve!
Student 8: I'm starting to solve!
Student 5: I'm starting to solve!
Student 27: I'm starting to solve!
Student 4: I'm starting to solve!
```

Студенты решают экзамен, начинается проверка:



```
Student 99: I'm starting to solve!
Student 100: I'm starting to solve!
Student 57: I solved all problems!
Teacher: I'm starting to check student's 57 work!
Student 18: I solved all problems!
Student 5: I solved all problems!
Student 31: I solved all problems!
Student 44: I solved all problems!
Student 54: I solved all problems!
Student 77: I solved all problems!
Student 84: I solved all problems!
Student 91: I solved all problems!
Student 1: I solved all problems!
Student 67: I solved all problems!
Student 90: I solved all problems!
Student 98: I solved all problems!
Student 16: I solved all problems!
Student 80: I solved all problems!
Student 33: I solved all problems!
Student 50: I solved all problems!
Student 48: I solved all problems!
Student 4: I solved all problems!
Student 93: I solved all problems!
Teacher: I checked student's 57 work! Exam score is 8
Student 45: I solved all problems!
Student 57: My score is 8.
Student 53: I solved all problems!
Student 65: I solved all problems!
Teacher: I'm starting to check student's 18 work!
Student 6: I solved all problems!
```

Проверка заканчивается:



```
Teacher: I checked student's 81 work! Exam score is 10
Teacher: I'm starting to check student's 63 work!
Student 81: My score is 10.
Teacher: I checked student's 63 work! Exam score is 9
Teacher: I'm starting to check student's 41 work!
Student 63: My score is 9.
Teacher: I checked student's 41 work! Exam score is 9
Teacher: I'm starting to check student's 66 work!
Student 41: My score is 9.
Teacher: I checked student's 66 work! Exam score is 8
Teacher: I'm starting to check student's 8 work!
Student 66: My score is 8.
Teacher: I checked student's 8 work! Exam score is 4
Teacher: I'm starting to check student's 49 work!
Student 8: My score is 4.
Teacher: I checked student's 49 work! Exam score is 4
Teacher: I'm starting to check student's 75 work!
Student 49: My score is 4.
Teacher: I checked student's 75 work! Exam score is 2
Teacher: I'm starting to check student's 56 work!
Student 75: My score is 2.
Teacher: I checked student's 56 work! Exam score is 8
Teacher: I'm starting to check student's 72 work!
Student 56: My score is 8.
Teacher: I checked student's 72 work! Exam score is 5
Student 72: My score is 5.
```

Список источников

1. https://software.intel.com/content/www/ru/ru/develop/articles/more-work-sharing-with-openmp.html - справочные материалы OpenMP
2. https://docs.microsoft.com/ru-ru/cpp/parallel/openmp/reference/openmp-directives?view=msvc-160 – документация по OpenMP в Visual C++
3. https://habr.com/ru/company/intel/blog/85273/ - статья на habr по OpenMP

Текст программы

```
#include <iostream>
#include <chrono>
#include <thread>
#include <queue>
#include <string>
#include <omp.h>
//#include <windows.h>

using namespace std;

// Очередь работ.
queue<int> works;
// Список оценок.
std::vector<int> scores;

// Студент.
class Student
{
	// Номер студента.
	int number;

	// Студент говорит что-то.
	void saySmth(std::string smth)
	{
#pragma omp critical(print)
		{
			cout << "Student " << number << ": " << smth << endl;
		}
	}
```

```
public:
    Student(int n)
    {
        number = n;
        srand(static_cast<unsigned>(n * n + static_cast<unsigned>(time(0))));
    }

    // Процесс сдачи экзамена.
    void takeExam()
    {
        saySmth("I'm starting to solve!");

        solve();

        {
#pragma omp critical(queue)
            {
                saySmth("I solved all problems!");
                // Сдаёт работу.
                works.push(number);
            }
        }

        bool isScoreGot = false;
        // до тех пор, пока не будет получена оценка.
        while (!isScoreGot)
        {
            int a;
#pragma omp critical(scores)
            {
                a = scores[number - 1];
            }
            if (a != 0)
            {
                // если есть оценка.
                int score;
#pragma omp critical(scores)
                {
                    score = scores[number - 1];
                }
                // Студент говорит, что знает свою оценку.
                saySmth("My score is " + std::to_string(score) + ".");
                isScoreGot = true;
            }
        }
    }

    void solve()
    {
        int t = rand() % 16 + 5; // от 5 до 20 секунд.
        std::this_thread::sleep_for(std::chrono::seconds(t));
        //Sleep(t * 1000);
    }
};

void threadStudentFunction(int a)
{
    Student stud = Student(a);
    stud.takeExam();
}

// Преподаватель.
class Teacher
{
```

```
public:
        // Проверяет работу.
        static void checkWork(int n)
        {
#pragma omp critical(print)
                {
                        cout << "Teacher: I'm starting to check student's " << n << " work!" << endl;
                }

                int t = rand() % 3 + 1; // от 1 до 3 секунд.
                std::this_thread::sleep_for(std::chrono::seconds(t));
                //Sleep(t * 1000);
                int score = rand() % 10 + 1;

#pragma omp critical(print)
                {
                        cout << "Teacher: I checked student's " << n << " work! Exam score is " << score
<< endl;
                }

#pragma omp critical(scores)
                {
                        scores[n - 1] = score;
                }
        }

        static void startExam(int numberOfStudents)
        {
                int numberOfCheckedWorks = 0;
                // до тех пор, пока не будут проверены все работы.
                while (numberOfCheckedWorks != numberOfStudents)
                {
```

```
            // если есть работы в очереди, проверять их.
            while (!works.empty())
            {
                int work;
#pragma omp critical(queue)
                {
                    work = works.front();
                    works.pop();
                }
                checkWork(work);
                numberOfCheckedWorks++;
                // работа проверена.
            }
        }
    }
};

int main(int argc, char* argv[]) {
    int numberOfStudents = std::stoi(argv[1]);
    if (numberOfStudents <= 0 || numberOfStudents > 100) { cout << "Wrong number of students! it should be > 0 and <= 100" << endl; return -1; }
    cout << "The exam begins! Number of students: " << numberOfStudents << endl;
    // Оценки.
    scores = std::vector<int>(numberOfStudents);
    for (int i = 0; i < numberOfStudents; i++)
    {
        scores.push_back(0);
    }

    // Потоки студентов и преподавателя.
#pragma omp parallel num_threads(numberOfStudents + 1)
    {
        auto threadN = omp_get_thread_num();
        // Поток преподавателя.
        if (threadN == 0)
        {
            Teacher::startExam(numberOfStudents);
        }
        // Потоки студентов.
        else
        {
            threadStudentFunction(omp_get_thread_num());
        }
    }

    return 0;
}
```