Пояснительная записка

НИУ ВШЭ, ФКН

Образовательная программа “Программная инженерия”, 2 курс

Курс «Архитектура вычислительных систем»

Практические примеры построения многопоточных приложений.

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 решения и оповещают преподавателя о том, что всё решили с помощью условной переменной. 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 и оповещает студентов о том, что проверил работу. Ожидающие студенты проверяют список и если находят свою оценку, то говорят о том, что ознакомились с оценкой и завершают экзамен. Часть работ будет проверена до того, как все написали экзамен, часть работ будет проверена после того, как все студенты допишут работы.

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: ExamThreads.exe <Число студентов>

Пример: ExamThreads.exe 10

Тестирование

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

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

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

C:\Users\xiaom\source\repos\ExamThreads\ExamThreads>ExamThreads.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\ExamThreads\ExamThreads>ExamThreads.exe 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\ExamThreads\ExamThreads>ExamThreads.exe 5
Student 1: I'm starting to solve!
Student 3: I'm starting to solve!
Student 5: 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

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

7) 100

Ввод:

```
C:\Users\xiaom\source\repos\ExamThreads\ExamThreads>ExamThreads.exe 100
```

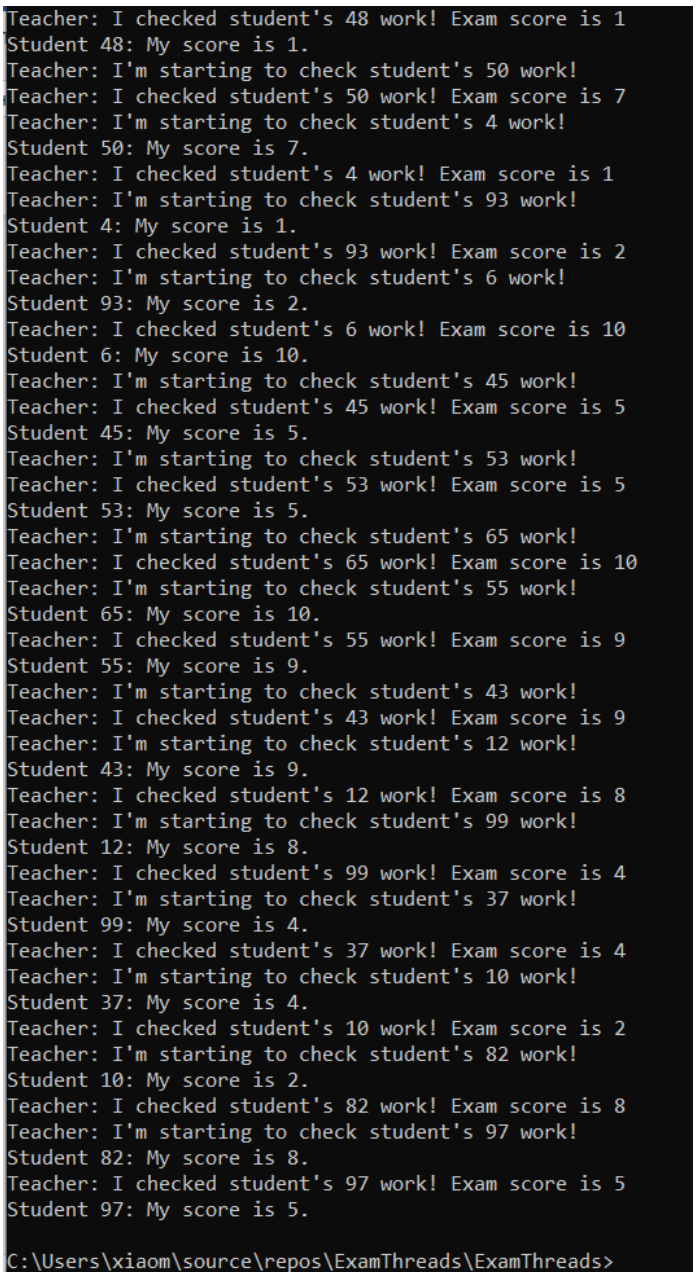
Студенты начинают работу:

```
Student 53: I'm starting to solve!
Student 8: I'm starting to solve!
Student 10: I'm starting to solve!
Student 9: I'm starting to solve!
Student 57: I'm starting to solve!
Student 58: I'm starting to solve!
Student 59: I'm starting to solve!
Student 60: I'm starting to solve!
Student 61: I'm starting to solve!
Student 14: I'm starting to solve!
Student 63: I'm starting to solve!
Student 64: I'm starting to solve!
Student 65: I'm starting to solve!
Student 66: I'm starting to solve!
Student 67: I'm starting to solve!
Student 68: I'm starting to solve!
Student 69: I'm starting to solve!
Student 70: I'm starting to solve!
Student 71: I'm starting to solve!
Student 72: I'm starting to solve!
Student 73: I'm starting to solve!
Student 74: I'm starting to solve!
Student 75: I'm starting to solve!
Student 76: I'm starting to solve!
Student 77: I'm starting to solve!
Student 79: I'm starting to solve!
Student 19: I'm starting to solve!
Student 21: I'm starting to solve!
Student 81: I'm starting to solve!
Student 82: I'm starting to solve!
Student 24: I'm starting to solve!
Student 23: I'm starting to solve!
Student 26: I'm starting to solve!
Student 86: I'm starting to solve!
Student 87: I'm starting to solve!
Student 29: I'm starting to solve!
Student 89: I'm starting to solve!
Student 90: I'm starting to solve!
Student 91: I'm starting to solve!
Student 92: I'm starting to solve!
Student 93: I'm starting to solve!
Student 95: I'm starting to solve!
Student 94: I'm starting to solve!
Student 96: I'm starting to solve!
Student 97: I'm starting to solve!
Student 98: I'm starting to solve!
Student 34: I'm starting to solve!
Student 100: I'm starting to solve!
```

Студенты решают экзамен, начинается проверка:

```
Student 94: I'm starting to solve!
Student 96: I'm starting to solve!
Student 97: I'm starting to solve!
Student 98: I'm starting to solve!
Student 34: I'm starting to solve!
Student 100: I'm starting to solve!
Student 39: I solved all problems!
Teacher: I'm starting to check student's 39 work!
Student 59: I solved all problems!
Student 61: I solved all problems!
Student 94: I solved all problems!
Student 13: I solved all problems!
Student 36: I solved all problems!
Student 9: I solved all problems!
Student 86: I solved all problems!
Student 92: I solved all problems!
Student 62: I solved all problems!
Student 25: I solved all problems!
Student 88: I solved all problems!
Student 40: I solved all problems!
Student 58: I solved all problems!
Student 24: I solved all problems!
Student 20: I solved all problems!
Student 85: I solved all problems!
Teacher: I checked student's 39 work! Exam score is 8
Teacher: I'm starting to check student's 59 work!
Student 39: My score is 8.
Student 73: I solved all problems!
Student 74: I solved all problems!
Student 29: I solved all problems!
Student 11: I solved all problems!
Student 30: I solved all problems!
Student 38: I solved all problems!
Student 69: I solved all problems!
Student 19: I solved all problems!
Student 89: I solved all problems!
Student 78: I solved all problems!
Student 22: I solved all problems!
Student 27: I solved all problems!
Student 60: I solved all problems!
Teacher: I checked student's 59 work! Exam score is 1
Teacher: I'm starting to check student's 61 work!
Student 59: My score is 1.
Student 68: I solved all problems!
Student 79: I solved all problems!
Student 3: I solved all problems!
Student 2: I solved all problems!
Student 87: I solved all problems!
```

Проверка заканчивается:



Список источников

1. https://habr.com/ru/post/182626/ - справочные материалы по потокам в C++

Текст программы

```
#include <iostream>
#include <mutex>
#include <chrono>
#include <thread>
#include <condition_variable>
#include <queue>
#include <string>

using namespace std;

// Mutex вывода.
std::mutex printMutex;
// Mutex препдавателя в ожидании ответов.
std::mutex teacherMutex;
// Mutex доступа к оценкам.
std::mutex scoresLock;
// Mutex очереди на оценивание.
std::mutex queueMutex;
// Условие появления работы в очереди.
std::condition_variable queueCheck;
// Условие появления оценки.
std::condition_variable scoreCheck;
// 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)
        {
                std::unique_lock<std::mutex> locker(printMutex);
                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();

                {
                        std::unique_lock<std::mutex> locker(queueMutex);
                        saySmth("I solved all problems!");
                        // Сдаёт работу.
                        works.push(number);
                        queueCheck.notify_one();
                }
```

```
		std::mutex waitingScore;
		bool isScoreGot = false;
		// до тех пор, пока не будет получена оценка.
		while (!isScoreGot)
		{
			std::unique_lock<std::mutex> locker(waitingScore);
			scoreCheck.wait(locker, [&]()
				{
					scoresLock.lock();
					int a = scores[number - 1];
					scoresLock.unlock();
					return a != 0;
				});
			// если есть оценка.
			scoresLock.lock();
			int score = scores[number - 1];
			scoresLock.unlock();
			// 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));
	}
};

void threadStudentFunction(int a)
{
```

```
	Student stud = Student(a);
	stud.takeExam();
}

// Преподаватель.
class Teacher
{
public:
	// Проверяет работу.
	static void checkWork(int n)
	{
		printMutex.lock();
		cout << "Teacher: I'm starting to check student's " << n << " work!" << endl;
		printMutex.unlock();

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

		printMutex.lock();
		cout << "Teacher: I checked student's " << n << " work! Exam score is " << score << endl;
		printMutex.unlock();

		scoresLock.lock();
		scores[n - 1] = score;
		scoresLock.unlock();
	}

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

```
            while (numberOfCheckedWorks != numberOfStudents)
            {
                std::unique_lock<std::mutex> locker(teacherMutex);
                queueCheck.wait(locker, [&]() {return !works.empty();});
                // если есть работы в очереди, проверять их.
                while (!works.empty())
                {
                    queueMutex.lock();
                    int work = works.front();
                    works.pop();
                    queueMutex.unlock();
                    checkWork(work);
                    numberOfCheckedWorks++;
                    // работа проверена.
                    scoreCheck.notify_all();
                }
            }
        }
};

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; }
        // Оценки.
        scores = std::vector<int>(numberOfStudents);
        for (int i = 0; i < numberOfStudents; i++)
        {
            scores.push_back(0);
        }
        // Поток преподавателя.
        std::thread serverThread(Teacher::startExam, numberOfStudents);
```

```
	// Потоки учеников.
	std::thread* threads = new std::thread[numberOfStudents];
	for (int i = 0; i < numberOfStudents; i++)
	{
		threads[i] = std::thread(threadStudentFunction, i + 1);
	}
	for (int i = 0; i < numberOfStudents; i++)
	{
		threads[i].join();
	}
	delete[] threads;
	serverThread.join();
	return 0;
}
```