НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

Факультет компьютерных наук
Департамент программной инженерии
Дисциплина: «Архитектура вычислительных систем»

Домашнее задание №3 по дисциплине

«Архитектура вычислительных систем»

На тему:

«Задача про экзамен»

Пояснительная записка

Выполнил:
Моторкин Владимир,
студент гр. БПИ198.

Москва
2020

Содержание

1. Текст задания

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».

2. Применяемые расчетные методы

2.1. Теория решения задания

В данном задании необходимо использовать парадигму «клиент-сервер». В качестве сервера выступает учитель, который принимает запросы от студентов на проверку их ответа.

Каждый билет имеет 10 вопросов, в каждом 4 варианта ответа. Итоговая оценка принимает значения [0; 10]. Верные ответы генерируются случайным образом. Студент отвечает свой билет так же, генерируя случайные значения. Всего 10 студентов.

При сдаче работы студентом, его номер добавляется в очередь, ответы заносятся в массив с ответами студентов и оповещается учитель. Учитель берёт верхний id из очереди и по нему получает записанные ответы студентов, сравнивает с правильными и за каждый правильный ответ даёт 1 балл.

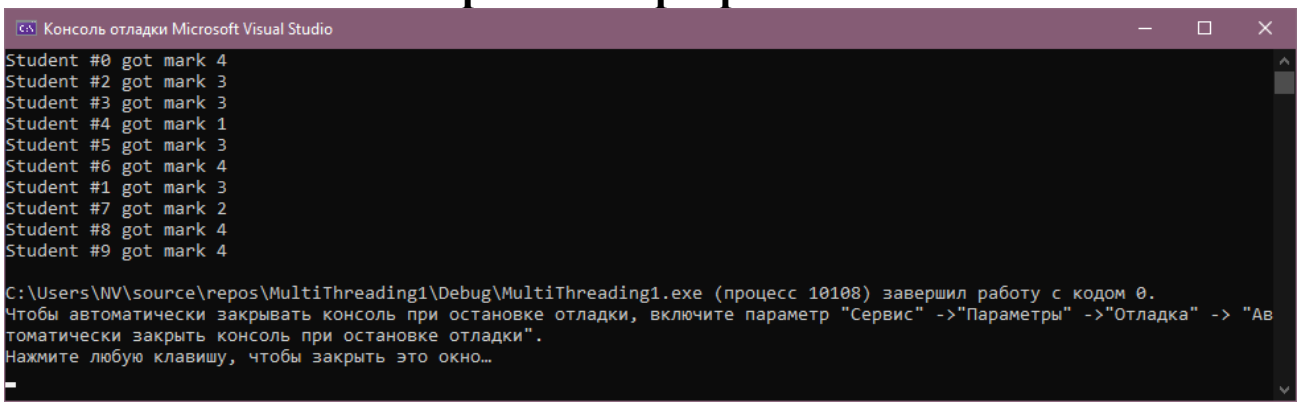
Главный поток (main) ожидает завершения потока учителя и завершает программу.

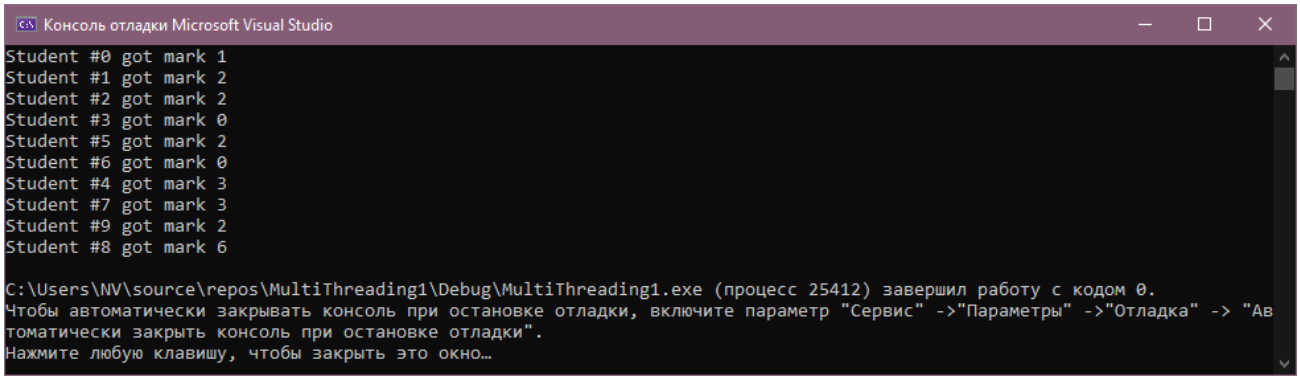
Основной библиотекой использовалась стандартная библиотека c++ (<thread>).

2.2. Описание переменных и функций программы

Тип	Название	Описание
mutex	lockQueue	Блокировка при добавлении
condition_variable	checkQueue	Условная переменная проверки добавления в очередь студентом своего id
Queue<int>	ids	Очередь для хранения id студентов
bool	flagDone	Флаг для отметки завершения проверки билетов
bool	flagNotified	Флаг для пометки о совершенном оповещении
int[][]	answers	Массив с верными ответами
int[][]	studentsAnswers	Массив с ответами студентов
void	studentFunc	Функция потока студента
void	teacherFunc	Функция потока учителя
void	fillArray	Функция для заполнения массива случайными ответами

3. Тестирование программы





ПРИЛОЖЕНИЕ 1

Список литературы

1. Потоки, блокировки и условные переменные в C++11 [Часть 1]. [Электронный ресурс] // URL: https://habr.com/ru/post/182610/ (дата обращения: 17.11.2020)
2. Потоки, блокировки и условные переменные в C++11 [Часть 2]. [Электронный ресурс] // URL: https://habr.com/ru/post/182626/ (дата обращения: 17.11.2020)
3. [C++] часть 3: синхронизация потоков в ресторане. [Электронный ресурс] // URL: https://yandex.ru/turbo/nuancesprog.ru/s/p/6546/ (дата обращения: 17.11.2020)

ПРИЛОЖЕНИЕ 2

Код программы

```
1.  #include <condition_variable>
2.  #include <iostream>
3.  #include <random>
4.  #include <thread>
5.  #include <mutex>
6.  #include <queue>
7.  using namespace std;
8.
9.  mutex lockQueue;
10.condition_variable checkQueue;
11.queue<int> ids;
12.bool flagDone;
13.bool flagNotified;
14.int answers[10][10];
15.int studentsAnswers[10][10];
16.
17.void studentFunc(int id, int* arr) {
18.
19.     for (size_t i = 0; i < 10; i++) {
20.             arr[i] = rand() % 4 + 1;
21.     }
22.     // Adding id and notifying.
23.     {
24.             unique_lock<mutex> locker(lockQueue);
25.             ids.push(id);
26.             flagNotified = true;
```

```
27.             checkQueue.notify_one();
28.     }
29.}
30.
31.void teacherFunc() {
32.     while (!flagDone)
33.     {
34.             unique_lock<mutex> locker(lockQueue);
35.             // Protection from wrong notifying.
36.             while (!flagNotified)
37.                     checkQueue.wait(locker);
38.             while (!ids.empty())
39.             {
40.                     int res = 0;
41.                     int id = ids.front();
42.                     for (size_t i = 0; i < 10; i++)
43.                     {
44.                             if (answers[id][i] == studentsAnswers[id][i])
45.                             {
46.                                     res++;
47.                             }
48.                     }
49.                     std::cout << "Student #" << id << " got mark " << res << std::endl;
50.                     ids.pop();
51.             }
52.             flagNotified = false;
53.     }
54.}
```

```
55.
56.void fillArray() {
57.     for (size_t i = 0; i < 10; i++) {
58.             for (size_t j = 0; j < 10; j++) {
59.                     answers[i][j] = rand() % 4 + 1;
60.             }
61.     }
62.}
63.
64.int main() {
65.     srand(time(NULL));
66.
67.     fillArray();
68.
69.     thread teacherThread(teacherFunc);
70.
71.     vector<thread> students;
72.     for (int i = 0; i < 10; i++)
73.             students.push_back(std::thread(studentFunc, i, ref(studentsAnswers[i])));
74.     for (auto& thread : students)
75.             thread.join();
76.
77.     flagDone = true;
78.
79.     teacherThread.join();
80.     return 0;
    }
```