НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

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

Задача об инвентаризации по рядам

Пояснительная записка

Выполнил:
Попов Андрей Эдуардович,
студент гр. БПИ197.

Москва
2020

1. Текст задания

Задача об инвентаризации по рядам. После нового года в библиотеке университета обнаружилась пропажа каталога. После поиска и наказания виноватых, ректор дал указание восстановить каталог силами студентов. Фонд библиотека представляет собой прямоугольное помещение, в котором находится M рядов по N шкафов по K книг в каждом шкафу. Требуется создать многопоточное приложение, составляющее каталог. При решении задачи использовать метод «портфель задач», причем в качестве отдельной задачи задается составление каталога одним студентом для одного ряда.

2. Применяемые расчетные методы

2.1. Теория решения задания

Для решения поставленной задачи использовался метод «Портфель задач»:
Каждый поток обрабатывает ряд, к которому ещё ни один другой поток не приступал и с помощью массива булевых переменных запрещает прочим потокам изменять этот ряд.
Как только поток обработает ряд, происходит поиск нового ряда.
Названия книг создаются случайным образом из букв английского алфавита и пробелов.

2.2. Организация входных данных

Программа запрашивает параметры M, N, K из условия задачи, а также число студентов.
1 <= M, N, K <= 100, а также: 1 <= число студентов <= M.

2.3 Организация выходных данных

Программа сообщает содержимое каталога (пример: рисунок 1)

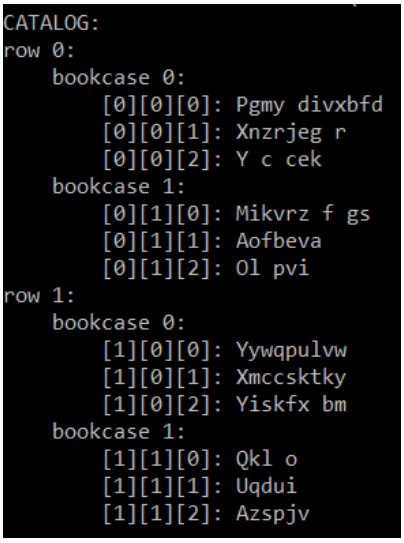

Рисунок 1.

.

2.4. Дополнительный функционал программы

Программа обрабатывает ситуацию неверного ввода, если были посланы не корректные числа или числа выходили за допустимый диапазон.

3. Тестирование программы

3.1. Корректные значения

Рисунок 2. M=1, N=1, K=1, students_count=1. Каталог создался корректно

```
Enter the number of rows(M)(Not less than 1, not more then 100): 1
Enter the number of bookcases(N)(Not less than 1, not more than 100): 1
Enter the number of books(K)(Not less than 1, not more than 100): 1
Enter the number of students(not less that 1 not more then M): 1
CATALOG:
row 0:
    bookcase 0:
        [0][0][0]: Pgmy divxbfd
```

Рисунок 3 M=100, N=20, K=5, students_count=100. Перед окончанием ввода данных

```
Enter the number of rows(M)(Not less than 1, not more then 100): 100
Enter the number of bookcases(N)(Not less than 1, not more than 100): 20
Enter the number of books(K)(Not less than 1, not more than 100): 5
Enter the number of students(not less that 1 not more then M): 100
```

Рисунок 4. Вывод ответа к тесту (рисунок 3). Лишь часть полного ответа. Каталог создался корректно

```
        [99][15][3]: Q cd bxl
        [99][15][4]: Ohqa
    bookcase 16:
        [99][16][0]: Wgqmuaseb  ea
        [99][16][1]: U nj gmvs
        [99][16][2]: Fbhqq
        [99][16][3]: Zv tjc j spm
        [99][16][4]: Fpspb
    bookcase 17:
        [99][17][0]: C  jx wxbf
        [99][17][1]: Z oiwcpha
        [99][17][2]: Fncjrvzmyl
        [99][17][3]: Ocxg begedcue
        [99][17][4]: Usva jdn
    bookcase 18:
        [99][18][0]: Dmh g om
        [99][18][1]: Elq vb
        [99][18][2]: Jmapyxrnf
        [99][18][3]: Xzfa
        [99][18][4]: Djimui
    bookcase 19:
        [99][19][0]: Ipfey i
        [99][19][1]: Aebdvdisl
        [99][19][2]: Smls
        [99][19][3]: R x  fps fc
        [99][19][4]: Njh bpp
```

3.2. Некорректные значения

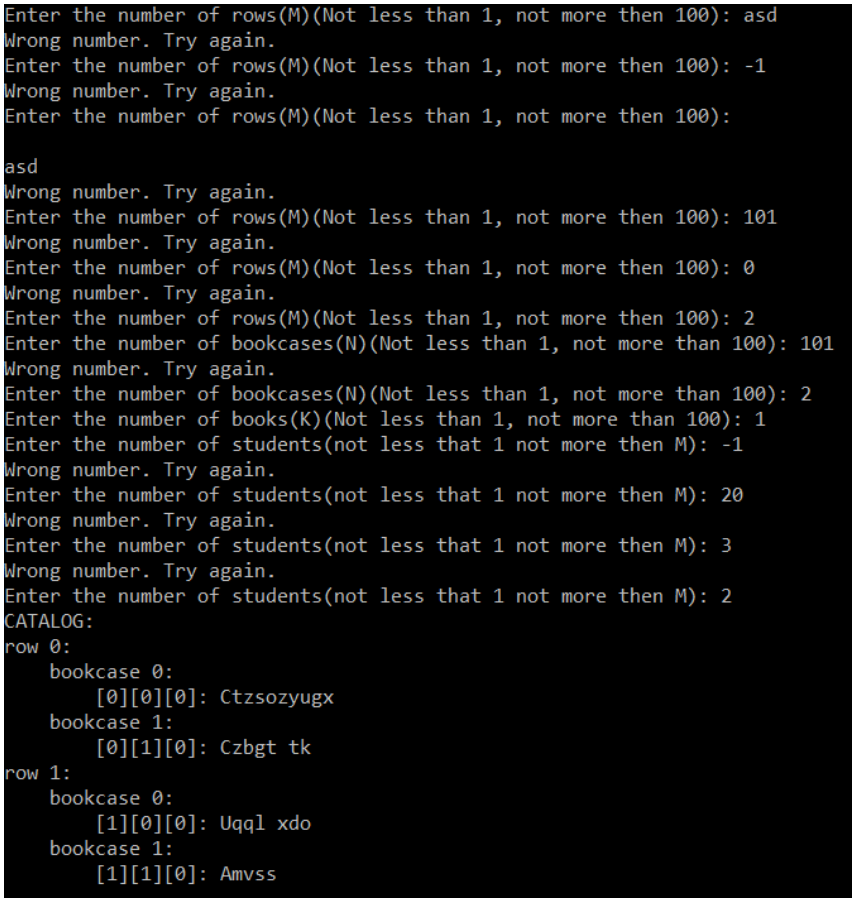

Рисунок 5. Множество попыток вводить строку или переносы строки, а также попытки выйти за допустимый диапазон значений. Программа предлагает юзеру заново ввести данные.

ПРИЛОЖЕНИЕ 1

Список литературы

1. Заголовок. [Электронный ресурс] // URL: https://solarianprogrammer.com/2012/10/17/cpp-11-async-tutorial
2. Заголовок. [Электронный ресурс] // URL: http://www.softcraft.ru/edu/comparch/

ПРИЛОЖЕНИЕ 2

Код программы

```
#include <iostream>
#include <string>
#include <vector>
#include <ctime>
#include <future>

//Попов Андрей
//БПИ197
//Вариант 16
//16. Задача об инвентаризации по рядам.После нового года в
//библиотеке университета обнаружилась пропажа каталога.После поиска и
//наказания виноватых, ректор дал указание восстановить каталог силами
//студентов.Фонд библиотека представляет собой прямоугольное помещение,
//в котором находится M рядов по N шкафов по K книг в каждом шкафу.
//Требуется создать многопоточное приложение, составляющее каталог.При
//решении задачи использовать метод «портфель задач», причем в качестве
//отдельной задачи задается составление каталога одним студентом для одного
//ряда.


/// <summary>
/// ввод целого числа, вводимое пользователем
/// </summary>
/// <param name="message">Сообщегие, котороре будет показано пользователю</param>
/// <param name="min_value">Минимальное значение вводимого числа</param>
/// <param name="max_value">Максимальное значение вводимого числа</param>
/// <returns>возвращает вводимое число</returns>
int get_int(std::string message, int min_value = INT32_MIN, int max_value = INT32_MAX) {
	while (true) {
		int integer;
		std::cout << message;
		std::cin >> integer;
		if (std::cin.fail() || min_value > integer || integer > max_value) {
			std::cin.clear();
			std::cin.ignore(INT32_MAX, '\n');
			std::cout << "Wrong number. Try again.\n";
		}
		else {
			std::cin.ignore(INT32_MAX, '\n');
			return integer;
		}
	}
```

```
}

/// <summary>
/// Возвращает случайное целое число включая обе границы
/// </summary>
/// <param name="min_value">левая граница</param>
/// <param name="max_value">правая граница</param>
/// <returns>Возвращает случайное целое число включая обе границы</returns>
int next_int(int min_value, int max_value) {
	return min_value + (rand() % (max_value - min_value + 1));
}

/// <summary>
/// создаёт случайное название книги
/// </summary>
/// <param name="min_length">минимальная длина</param>
/// <param name="max_length">максимальная длина</param>
/// <returns>случайное название книги</returns>
std::string get_random_title(int min_length = 3, int max_length = 30) {
	std::string str;
	str += (char)next_int('A', 'Z');
	for (int i = 0; i < next_int(min_length, max_length) - 1; i++) {
		if (next_int(1, 1000) < 175) //обусловлено приблизительной частотой пробела в
текстах на английском языке
			str += " ";
		else
			str += (char)next_int('a', 'z');
	}
	return str;

}

/// <summary>
/// Создаёт ряд в каталоге
/// </summary>
/// <param name="rows">каталог</param>
/// <param name="row">конкретный ряд этого каталога</param>
/// <param name="M"></param>
/// <param name="N"></param>
/// <param name="K"></param>
void make_row(std::string*** rows, int row, int M, int N, int K) {
	for (int j = 0; j < N; j++) {
		for (int k = 0; k < K; k++)
			rows[row][j][k] = get_random_title();
	}
```

```
}

/// <summary>
/// Символизирует работу студента, заполняющего последовательно ряды, если к ним ещё никто не приступал
/// </summary>
/// <param name="rows">каталаг</param>
/// <param name="started">массив булов, определящий, начал ли кто-то работу на каким-либо рядом</param>
/// <param name="M"></param>
/// <param name="N"></param>
/// <param name="K"></param>
/// <param name="student_number">уникальный номер студента</param>
/// <returns></returns>
int student_thread(std::string*** rows, bool* started, int M, int N, int K, int student_number) {
	srand(time(NULL));
	for (int i = 0; i < M; i++) //реализация "портфеля зада" - поиск рядов, к которым ещё никто не приступал и заполнение этих рядов.
		if (!started[i]) {
			started[i] = true;
			//std::cout << "student " << student_number << " fills in the row " << i << '\n'; //для проверок, что потоки работают как надо
			make_row(rows, i, M, N, K);
			//std::cout << "student " << student_number << " finished filling the row " << i << '\n';

		}
	return 0;
}

/// <summary>
/// Выписывает все книги каталога
/// </summary>
/// <param name="rows">каталог</param>
/// <param name="M"></param>
/// <param name="N"></param>
/// <param name="K"></param>
void write_answer(std::string*** rows, int M, int N, int K) {
	std::cout << "CATALOG:\n";
	for (int i = 0; i < M; i++) {
		std::cout << "row " << i << ":\n";
		for (int j = 0; j < N; j++) {
			std::cout << "    bookcase " << j << ":\n";
			for (int k = 0; k < K; k++)
				std::cout << "       [" << i << "][" << j << "][" << k << "]: " << rows[i][j][k] << '\n';
```

```
                }
        }

}


int main() {
        int M = get_int("Enter the number of rows(M)(Not less than 1, not more then 100): ", 1,
100); // ввод входных данных
        int N = get_int("Enter the number of bookcases(N)(Not less than 1, not more than 100): ", 1,
100);
        int K = get_int("Enter the number of books(K)(Not less than 1, not more than 100): ", 1,
100);
        int students_count = get_int("Enter the number of students(not less that 1 not more then
M): ", 1, M);
        std::string*** rows = new std::string * *[M]; //генерация каталога
        for (int i = 0; i < M; i++) {
                rows[i] = new std::string * [N];
                for (int j = 0; j < N; j++) {
                        rows[i][j] = new std::string[K];
                }
        }
        bool* started = new bool[M];
        for (int i = 0; i < M; i++)
                started[i] = false;
        std::vector<std::future<int>> students; //вектор потоков
        for (int i = 0; i < students_count; i++)
                students.push_back(async(student_thread, rows, started, M, N, K, i)); //запуск
вектора потоков
        for (std::future<int>& student : students) //ожидание пока потоки не завершат работу
                student.get();
        write_answer(rows, M, N, K); //вывод ответа на задачу
}
```