**Правительство Российской Федерации**

Федеральное государственное автономное образовательное учреждение высшего образования «Национальный исследовательский университет «Высшая школа экономики»

Факультет компьютерных наук

**Отчет к домашней работе №4**

**По дисциплине**

**“Архитектура вычислительных систем”**

Работу выполнил

Студент группы БПИ-197 \_\_\_\_\_\_\_\_\_\_\_\_\_ А. А. Синенко

подпись, дата

Работу проверил \_\_\_\_\_\_\_\_\_\_\_\_\_ А. И. Легалов

подпись, дата

Москва 2020

Содержание

[Постановка задачи 3](#_Toc56510232)

[Уточнение задачи 3](#_Toc56510233)

[Тестирование программы 3](#_Toc56510234)

[Использованная литература 4](#_Toc56510235)

[Приложение 5](#_Toc56510236)

# Постановка задачи

В цехе по заточке булавок все необходимые операции осуществляются тремя рабочими. Первый из них берет булавку и проверяет ее на предмет кривизны. Если булавка не кривая, то рабочий передает ее своему напарнику. Напарник осуществляет собственно заточку и передает заточенную булавку третьему рабочему, который осуществляет контроль качества операции. Требуется создать многопоточное приложение, моделирующее работу цеха. При решении использовать парадигму «производитель-потребитель».

# Уточнение задачи

1. Необходимо при помощи библиотек по работе с потоками в C++ написать программу, моделирующую работу цеха по производству булавок (pin).
2. Выводить действия рабочих в консоль или в файл.
3. Использовать парадигму “Производитель-потребитель” (1, стр. 1).

Реализация программы

* Для хранения потоков проинициализированы три поля: threadFisrt, threadSecond, threadThird.
* Для хранения информации о производимой булавке создана структура pin.
* Поле producted для хранения произведенных деталей.
* Методы firstWorker, secondWorker, thirdWorker для моделирования работы трех рабочих.
* В методеint main() реализовано создание потока первого рабочего (firstWorker), ожидание завершения этого потока и корректное завершение программы.

# Тестирование программы

При корректном запуске программа выводит по строке раз в полминуты с информацией о действиях, совершенных тремя потоками.

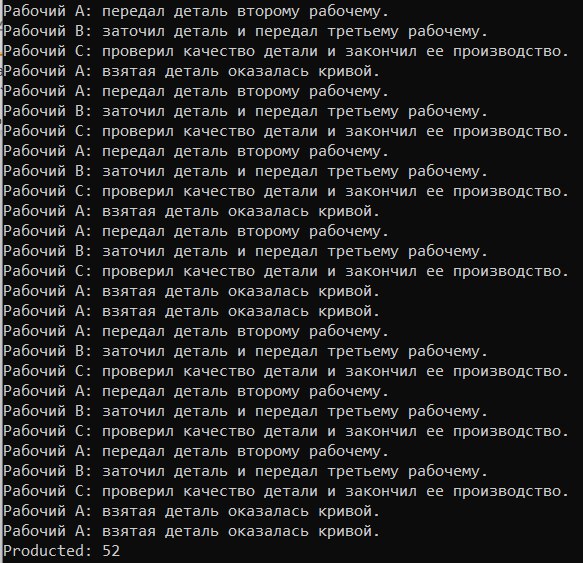


Рисунок 5 – демонстрация работы программы (последние строки).

# Использованная литература

1. https://studydocs.ru/studfiles/112/916/656121/9.doc.html (интернет источник)

# Приложение

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19  20  21  22  23  24  25  26  27  28  29  30  31  32  33  34  35  36  37  38  39  40  41  42  43  44  45  46  47  48  49  50  51  52  53  54  55  56  57  58  59  60  61  62  63  64  65  66  67  68  69  70  71  72  73  74  75  76  77  78  79  80  81  82  83  84  85  86  87  88  89  90  91  92  93  94  95  96  97  98  99  100  101 | #include <iostream>  #include <queue>  #include <stack>  #include <pthread.h>  #include <windows.h>  **using** **namespace** std;  // Потоки  **pthread\_t** threadFisrt, threadSecond, threadThird;  // Количество деталей  **const** **size\_t** COUNT = **100**;  // Структура детали  **struct** pin  {  // Кривая ли деталь  **bool** isCurved;  // Качественная ли заточка  **bool** isGoodQuality;  pin()  {  isCurved = rand() % **2**;  }  };  // Контейнер с произведенными деталями  stack<pin> producted;  // Третий рабочий  **void**\* **thirdWorker**(**void**\* param)  {  // Задержка на выполнение операции  Sleep(**500**);  pin\* takenPin = (pin\*)param;  // Проверка на качество  **if** (takenPin->isGoodQuality)  {  // Конец производства  cout << "Рабочий С: проверил качество детали и закончил ее производство." << endl;  producted.push(\*takenPin);  }  **else**  {  // Брак  cout << "Рабочий C: качество детали было низкое, поэтому она забракована." << endl;  }  **return** NULL;  }  // Второй рабочий  **void**\* **secondWorker**(**void**\* param)  {  pin\* takenPin = (pin\*)param;  // Заточка  takenPin->isGoodQuality = rand() % **2**;  Sleep(**500**);  // Передача детали третьему рабочему  cout << "Рабочий B: заточил деталь и передал третьему рабочему." << endl;  pthread\_join(threadThird, NULL);  pthread\_create(&threadThird, NULL, thirdWorker, takenPin);  **return** NULL;  }  // Первый рабочий  **void**\* **firstWorker**(**void**\* param)  {  queue<pin>\* conveer = (queue<pin>\*)param;  **while** (!conveer->empty())  {  pin takenPin = conveer->front();  conveer->pop();  // Задержка на выполнение операции  Sleep(**500**);  // Проверка на кривую деталь  **if** (!takenPin.isCurved)  {  cout << "Рабочий A: передал деталь второму рабочему." << endl;  // Передает второму рабочему  pthread\_create(&threadSecond, NULL, secondWorker, &takenPin);  pthread\_join(threadSecond, NULL);  pthread\_join(threadThird, NULL);  }  **else**  cout << "Рабочий A: взятая деталь оказалась кривой." << endl;  }  **return** NULL;  }  // Входная точка программы  **int** **main**()  {  setlocale(LC\_ALL, "Russian");  // Конвеер в виде очереди  queue<pin> conveer;  **for** (**int** i = **0**; i < COUNT; i++)  conveer.emplace(pin());  // Создание потока первого рабочего  pthread\_create(&threadFisrt, NULL, firstWorker, &conveer);  pthread\_join(threadFisrt, NULL);  // Всего произведено  cout << "Producted: " << producted.size() << endl;  // Выход из программы  **return** **0**;  } |