**Министерство науки и высшего образования Российской Федерации**
**Федеральное государственное бюджетное образовательное учреждение**
**высшего образования**
**«Московский государственный технический университет**
**имени Н.Э. Баумана**
**(национальный исследовательский университет)»**
**(МГТУ им. Н.Э. Баумана)**

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

# Отчёт

## по лабораторной работе № 5

**Название:** Многопоточная реализация конвейера

**Дисциплина:** Анализ алгоритмов

Студент ИУ7-55Б _____ Д.О. Склифасовский

(Группа) (Подпись, дата) (И.О. Фамилия)

Преподователь _____ Л.Л. Волкова

(Подпись, дата) (И.О. Фамилия)

*Москва, 2020*

**Содержание**

**Введение**

Цель работы: создать систему конвейерной обработки.

В ходе лабораторной работы требуется:

1) описать алгоритм, реализующий конвейерную обработку;
2) реализовать данный алгоритм;
3) провести тестирование алгоритма.

# 1 Аналитический раздел

В данном разделе представлены принципы конвейерной обработки и параллельных вычислений.

## 1.1 Общие сведения

Ленточный **конвейер** (англ. belt conveyor) — транспортирующее устройство непрерывного действия с рабочим органом в виде ленты.

**Конвейерное производство** – это разновидность организации современного производственного процесса [1]. Конвейер используется повсеместно. Он облегчил труд человека и повысил производительность. С его помощью к процессу изготовления продукции задействуют минимальное количество сотрудников. Такое оборудование позволяет будущему товару, самостоятельно проходить различные стадии производства. С его появлением, компании смогли повысить свои показатели, изготавливая конечную продукцию в разы быстрее.

## 1.2 Параллельное программирование

Параллельное программирование. Параллельное программирование служит для создания программ, эффективно использующих вычислительные ресурсы за счет одновременного исполнения кода на нескольких вычислительных узлах. Для создания параллельных приложений используются параллельные языки программирования и специализированные системы поддержки параллельного программирования, такие как MPI и OpenMP. Параллельное программирование является более сложным по сравнению с последовательным как в написании кода, так и в его отладки. Для облегчения процесса параллельного программирования существуют специализированные инструменты, например, отладчик TotalView.[2]

## 1.3 Вывод

В данном разделе были рассмотрены основы конвейерной обработки и технология параллельного программирования.

# 2 Конструкторский раздел

В данном разделе представлена схема алгоритмов обработки элементов линии конвейера.

## 2.1 Разработка алгоритма

На рисунке 1 изображена схема алгоритма запуска линии конвейера.

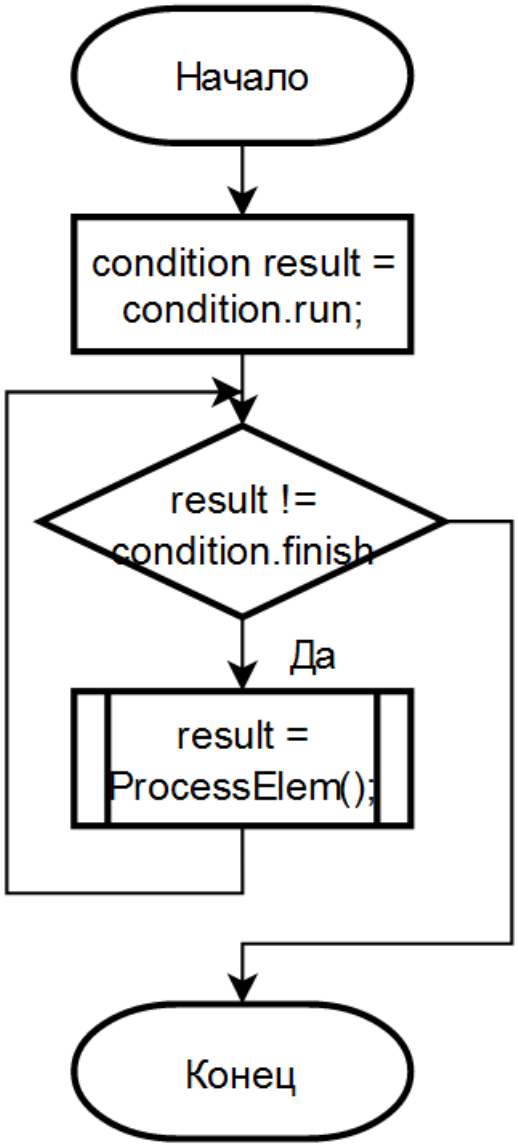


**Рисунок 1** – Схема алгоритма запуска линии конвейера

На рисунке 2 изображена схема алгоритма обработка элементов линии.

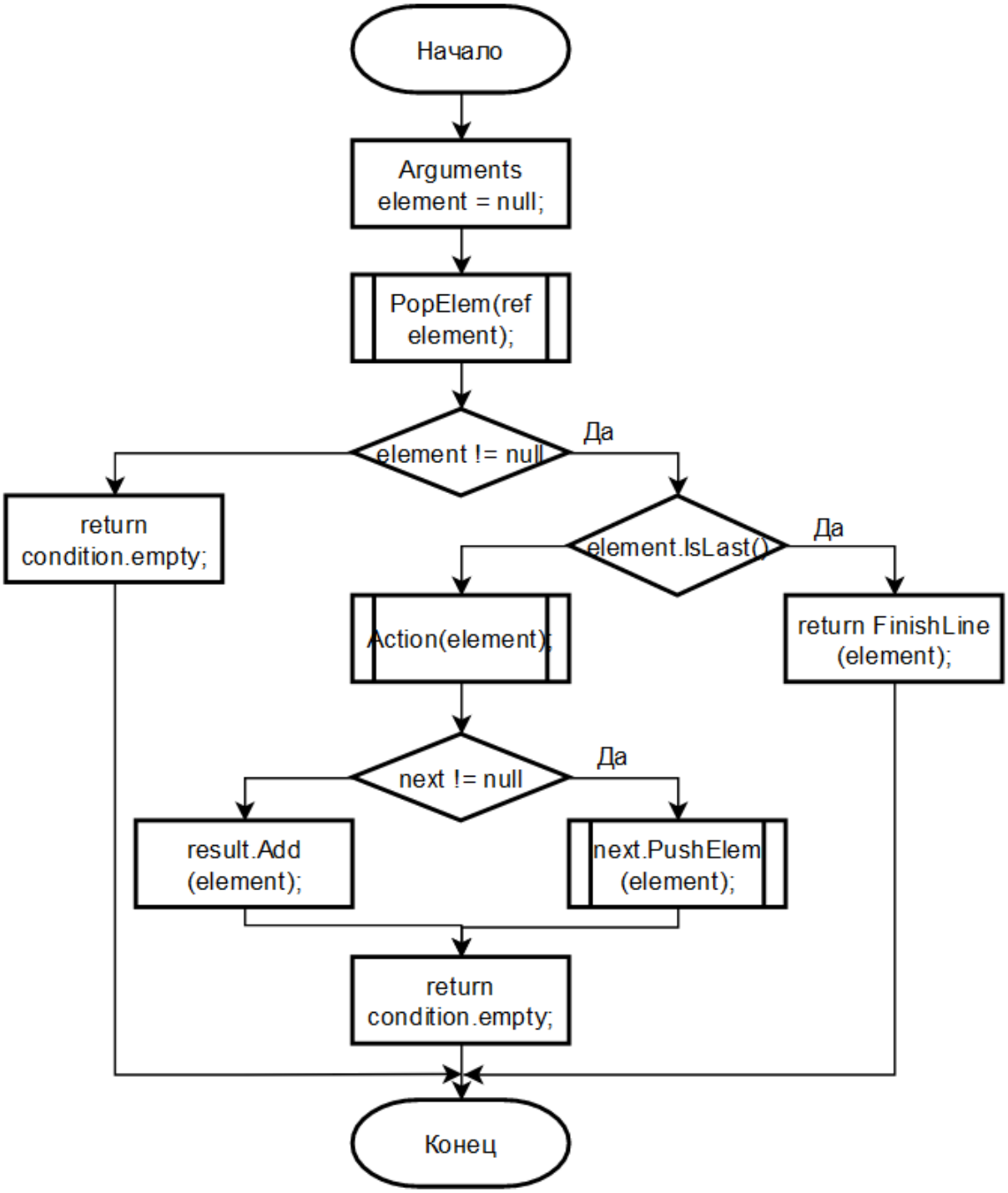


**Рисунок 2** – Схема алгоритма обработка элементов линии

## 2.2 Описание системы

Система состоит из трех линий (очередей), который последовательно соединены между собой (у каждой линии есть ссылка на следующую). Изна-

чально генерируются аргументы. У последнего из аргументов присутствует флаг, указывающий на то, что он последний. Все аргументы записываются в первую очередь, а из последней попадают в результирующий массив. Выдается время начала и конца обработки элемента.

## 2.3 Вывод

В данном разделе были рассмотрены схема алгоритмов обработки элементов линии конвейера и описана система ПО.

# 3 Технологический раздел

В данном разделе даны общие требования к программе, средства реализации и сама реализация алгоритмов.

## 3.1 Общие требования

**Требования к программе**

1) аргументы должны последовательно проходить линии в заданном порядке;

2) каждая линия должна работать в своем потоке;

3) последним элементом должен быть флаг, при поступлении которого, линия должна завершить свою работу;

4) при пустой очереди линии, она должна ожидать поступления нового элемента;

5) в результате из последней линии должен вернуться массив аргументов, у которого порядок совпадает с начальным.

## 3.2 Средства реализации

В качестве языка программирования был выбран C#, так как я знаком с данным языком программирования, имею представление о способах тестирования программы.[3]

Средой разработки Visual Studio.[4]

Для замеров процессорного времени используется функция $Stopwatch$.[5][6]

Многопоточное программирование было реализовано с помощью пространства имен System.Threading.[7]

### 3.3 Сведения о модулях программы

Программа состоит из:

1) Program.cs - главный файл программы, в котором располагается точка входа в программу;

2) Arguments.cs - файл класса Arguments;

3) Line.cs - файл класса линии конвейера;

### 3.4 Листинг кода программы

В листинге 1 реализован алгоритм запуска линии конвейера.

**Листинг 1** – Запуска линии конвейера

```
public void RunLine()
{
   condition result = condition.run;
   while (result != condition.finish)
   {
      result = ProcessElem();
   }
}
```

В листинге 2 реализован алгоритм обработки аргумента линии.

**Листинг 2** – Обработки аргумента линии

```
private condition ProcessElem()
{
   Arguments element = null;
   PopElem(ref element);
   if (element != null)
   {
```

```
7       if (element.IsLast())
8       {
9           return FinishLine(element);
10      }
11      Action(element);
12      if (next != null)
13      {
14          next.PushElem(element);
15      }
16      else if (isLast)
17      {
18          result.Add(element);
19      }
20  }
21  else
22  {
23      return condition.empty;
24  }
25  return condition.run;
26 }
```

В листинге 3 реализовано добавление элемента в очередь.

**Листинг 3** – Добавление элемента в очередь

```
1 public void PushElem(Arguments arg)
2 {
3   lock(myQueue)
4   {
5     myQueue.Enqueue(arg);
6   }
```

```
}
```

В листинге 4 реализовано взятие элемента из очереди.

**Листинг 4** – Взятие элемента из очеред

```
private void PopElem(ref Arguments arg)
{
   lock (myQueue)
   {
      if (myQueue.Count > 0)
      {
         arg = myQueue.Dequeue();
      }
   }
}
```

### 3.5 Вывод

В данном разделе были даны общие требования к программе, описаны средства реализации, были представлены сведения о модулях программы, а также реализованы алгоритмы работы линии конвейера.

# 4 Экспериментальный раздел

В данном разделе представлен результаты работы программы и показано параллельное выполнение операций.

## 4.1 Пример работы программы

На рисунке 3 изображена результат работы программы.

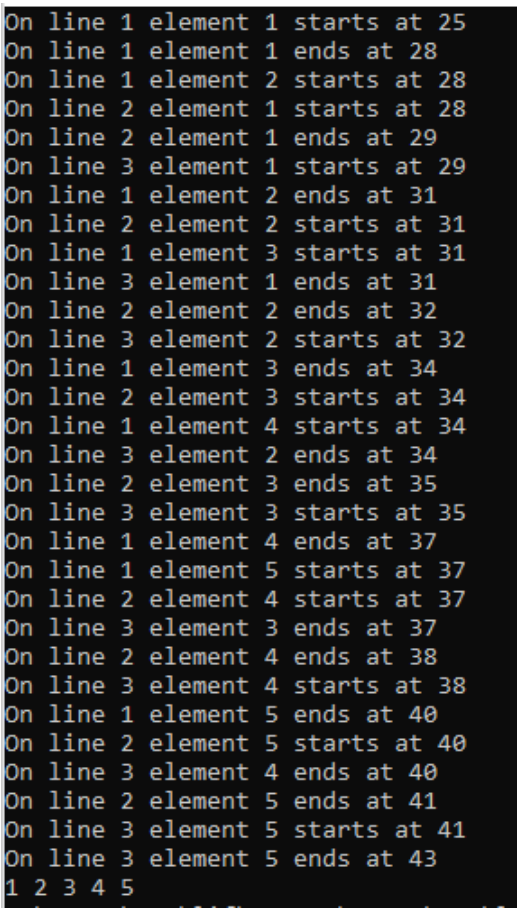


**Рисунок 3** – Результат работы программы

## 4.2 Вывод

Результаты работы программы показывают, что действия, а именно начало обработки элементов и конец, выполнялись не последовательно, а каждая линия работала параллельно. Конечный результат показывает, что последовательность элементов сохраняется.

# 5 Заключение

В ходе выполнения лабораторной работы были изучены возможности параллельных вычислений и был использован такой подход на практике. Был разработан конвейер с параллельно работающими линиями. Конвейерная обработка позволяет ускорить работу программы, если требуется обработка однотипных данных.

**Литература**


1. Конвейерное производство. -URL: http://dirsystem.ru/konvejernoe-proizvodstvo (дата обращения 19.11.2020). -Текст: электронный.

2. Параллельное программирование. -URL: https://www.viva64.com/ru/t/0038/ (дата обращения: 24.10.2020). -Текст: электронный.

3. Документация по C#. -URL: https://docs.microsoft.com/ru-ru/dotnet/csharp/ (дата обращения: 24.10.2020). -Текст: электронный.

4. Документация по семейству продуктов Visual Studio. -URL: https://docs.microsoft.com/ru-ru/visualstudio/?view=vs-2019 (дата обращения: 01.10.2020). -Текст: электронный.

5. Stopwatch Класс. -URL: https://goo.su/2e99 (дата обращения: 24.10.2020). -Текст: электронный.

6. Под капотом у Stopwatch. -URL: https://habr.com/ru/post/226279/ (дата обращения: 24.10.2020). Текст: электронный.

7. Thread Класс. -URL: https://docs.microsoft.com/ru-ru/dotnet/api/system.threading.5.0 (дата обращения: 19.11.2020). Текст: электронный.