

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ
ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Забайкальский государственный университет»
(ФГБОУ ВО «ЗабГУ»)
Факультет энергетический
Кафедра информатики, вычислительной техники и прикладной математики

КУРСОВАЯ РАБОТА

по Теории языков программирования и методам трансляции
на тему «Программная реализация транслятора»

Курзый К. - отлично
Чернобровкин - хорошо

Выполнили ст. гр. ИВТ-18 Гурулёв П.Д., Чернобровкин Д.Р.

Проверил доцент кафедры ИВТ и ПМ Коган Е.С.

Чита
2021

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ
ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Забайкальский государственный университет»
(ФГБОУ ВО «ЗабГУ»)
Факультет энергетический
Кафедра информатики, вычислительной техники и прикладной математики

ЗАДАНИЕ
на курсовую работу


По дисциплине: Теория языков программирования и методы трансляции

Студентам: Чернобровкину Д.Р., Гурулёву П.Д.

Специальности (направления подготовки): 09.03.01 Информатика и
вычислительная техника


- 1 Тема курсовой работы: «Программная реализация транслятора»
- 2 Срок подачи студентом законченной работы: 28.12.2021
- 3 Исходные данные к работе:
 1. Описание предметной области;
 2. «Общие требования к построению и оформлению учебной текстовой документации» (МИ 01-02-2018)».

Дата выдачи задания: 15.09.2021

Руководитель курсовой работы  /Коган Е.С./
(подпись, расшифровка подписи)

Задание принял к исполнению

«15» сентября 2021 г.

Подпись студента  /Гурулёв П.Д.

Подпись студента  /Чернобровкин Д.Р.

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ
ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение
высшего образования
«Забайкальский государственный университет»
(ФГБОУ ВО «ЗабГУ»)
Факультет энергетический
Кафедра информатики, вычислительной техники и прикладной математики

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
к курсовой работе

По дисциплине Теория языков программирования и методы трансляции

На тему «Программная реализация транслятора»

Выполнили студенты группы ИВТ–18: Чернобровкин Д.Р., Гурулёв П.Д.

Руководитель работы: доцент кафедры ИВТ и ПМ Коган Е.С.

Чита
2021

РЕФЕРАТ

Пояснительная записка 23с., 8 рис., 0 табл., 8 источников, 0 прил.

КОНВЕРТОР, PASCAL, C++, АЛГОРИТМ, АЛГОРИТМИЧЕСКИЙ
ЯЗЫК, ПЕРЕМЕННАЯ, ЦИКЛ, УСЛОВИЕ.

Для достижения цели данной работы необходимо создать интерфейс взаимодействия пользователя и программы, а также реализовать программу конвертации из языка Pascal в алгоритмический язык и из алгоритмического языка на язык C++.

Приложение позволяет конвертировать код, написанный на языке Pascal в код, написанный на алгоритмическом языке, а из него в код, написанный на языке C++.

Интерфейс был создан при помощи языка C#.

СОДЕРЖАНИЕ

Введение.....	6
1. Теоретическая часть.....	7
1.1 Структура языка Pascal.....	7
1.2. Структура алгоритмического языка.....	9
1.3. Структура языка C++.....	10
1.4. Перевод из языка Pascal в алгоритмический язык и в язык C++	12
1.4.1. Функции обработки	12
1.4.2. Анализ ошибок.....	13
2. Практическая часть	14
2.1. Программная реализация	14
Заключение	21
Список использованных источников	22

ВВЕДЕНИЕ

В настоящее время существует огромное количество языков программирования. И для облегчения и ускорения перевода программы с одного языка на другой были придуманы специальные программы – трансляторы.

Транслятор — программа или техническое средство, выполняющее трансляцию программы — преобразование программы, представленной на одном из языков программирования, в программу на другом языке и, в определённом смысле, равносильную первой.

Целью данной курсовой работы является разработка конвертора с языка Pascal в алгоритмический язык, а из него на язык C++. Для достижения поставленной цели было решено разработать программное средство, способное реализовать процесс конвертации.

Интерфейс был создан при помощи языка разметки C#.

1. Теоретическая часть

Для того, чтобы реализовать конвертор из языка Pascal в алгоритмический язык и в язык C++ нам потребуется детально рассмотреть все три языка программирования, найти соответствие между ключевыми словами и конструкциями.

1.1 Структура языка Pascal

Язык Pascal – это чисто процедурный язык программирования, часто использующийся для обучения структурному программированию. Особенности языка являются строгая типизация и наличие средств структурного (процедурного) программирования. Язык предоставляет ряд встроенных структур данных: записи, массивы, файлы, множества и указатели.

Для того, чтобы описать алгоритм будущей программы используются следующие ключевые слова:

1. Program – это ключевое слово начало программы;
2. Var – это ключевое слово используется для описания переменных, которые будут использованы в программе;
3. Const – это ключевое слово, после которого идет перечисление констант;
4. Begin – обозначение начала алгоритма;
5. End – обозначение конца алгоритма.
6. Array – ключевое слово для обозначения массива.

Для правильного описания переменных в алгоритме предусмотрены ключевые слова для обозначения их типов:

1. Integer – это целочисленный тип данных;

2. Real – эта лексема используется для обозначения вещественного типа данных;

3. String – это ключевое слово требуется для описания символьных переменных;

4. Boolean – это ключевое слово говорит о том, что переменная логическая.

Циклические конструкции в алгоритмическом языке также используются, для их реализации используются такие ключевые слова:

1. Begin и end– используются для обозначения начала и конца цикла;

2. While – эта лексема используется для циклов с предусловием.

3. For – это ключевое слово используется для цикла со счётчиком.

4. Repeat, until - эта лексема используется для циклов с постусловием.

5. To, do, downto – ключевые слова являются параметрическими.

Для реализации условий в алгоритмическом языке используются ключевые слова:

1. If – используется перед условием, после которого будет происходить выполнение алгоритма;

2. Then – это ключевое слово, указывающее на действие, которое будет выполняться после невыполнения условия, указанное после «if»;

3. Else– – это ключевое слово, указывающее на действие, которое будет выполняться после невыполнения условия, указанное после «if»;

4. And, or, not, <, >, =, <=, > =, <> – логические знаки и выражений используются стандартные.

Ввод и вывод обозначается специальными словами Write, Writeln и Read, Readln соответственно.

1.2 Структура алгоритмического языка

Алгоритмический язык программирования — формальный язык, используемый для записи, реализации и изучения алгоритмов. В отличие от большинства языков программирования, алгоритмический язык не привязан к архитектуре компьютера, не содержит деталей, связанных с устройством машины.

Основные служебные слова алгоритмического языка:

1. АЛГ – название алгоритма (программы)
2. ПЕР – объявление переменных
3. АРГ – аргумент
4. РЕЗ – результат
5. НАЧ – начало алгоритма
6. КОН – конец алгоритма
7. ДАНО – исходные данные в произвольной форме
8. НАДО – цель алгоритма

Для правильного описания переменных в алгоритме предусмотрены следующие ключевые лексемы:

1. ЦЕЛ (целый)
2. ВЕЩ (вещественный)
3. СИМ (символьный)
4. ЛИТ (литера) — строка
5. ЛОГ (логический)
6. ТАБ (таблица) — для обозначения массива
7. ДЛИН (длина) — количество элементов массива

Для реализации условий и ветвления в алгоритмическом языке используются ключевые слова:

1. ЕСЛИ
2. ТО

3. ИНАЧЕ
4. ВСЕ
5. ВЫБОР
6. ПРИ
7. ЗНАЧ
8. И, ИЛИ, НЕ, ДА, НЕТ

Циклические конструкции в алгоритмическом языке также используются, для их реализации используются такие ключевые слова:

1. НЦ (начало цикла)
2. КЦ (конец цикла)
3. ПОКА
4. ДЛЯ
5. ОТ
6. ДО
7. ШАГ

Для обозначения ввода и вывода используются специальные слова
ВВОД и ВЫВОД

1.3 Структура языка C++

C++ — компилируемый, статически типизированный язык программирования общего назначения.

Поддерживает такие парадигмы программирования, как процедурное программирование, объектно-ориентированное программирование, обобщённое программирование. Язык имеет богатую стандартную библиотеку, которая включает в себя распространённые контейнеры и алгоритмы, ввод-вывод, регулярные выражения, поддержку многопоточности и другие возможности. C++ сочетает свойства как высокоуровневых, так и низкоуровневых языков. В сравнении с его

предшественником — языком С — наибольшее внимание уделено поддержке объектно-ориентированного и обобщённого программирования.

С++ широко используется для разработки программного обеспечения, являясь одним из самых популярных языков программирования. Область его применения включает создание операционных систем, разнообразных прикладных программ, драйверов устройств, приложений для встраиваемых систем, высокопроизводительных серверов, а также игр. Существует множество реализаций языка С++, как бесплатных, так и коммерческих и для различных платформ. Например, на платформе x86 это GCC, Visual C++, Intel C++ Compiler, Embarcadero (Borland) C++ Builder и другие. С++ оказал огромное влияние на другие языки программирования, в первую очередь на Java и C#.

Синтаксис С++ унаследован от языка С. Изначально одним из принципов разработки было сохранение совместимости с С. Тем не менее С++ не является в строгом смысле надмножеством С; множество программ, которые могут одинаково успешно транслироваться как компиляторами С, так и компиляторами С++, довольно велико, но не включает все возможные программы на С.

Несмотря на то, что большая часть кода С будет справедлива и для С++, С++ не является надмножеством С и не включает его в себя. Существуют некоторые различия между этими двумя языками. Например, С++ не разрешает вызывать функцию `main()` внутри программы, в то время как в С это действие правомерно. Кроме того, С++ более строг в некоторых вопросах; например, он не допускает неявное приведение типов между несвязанными типами указателей и не разрешает использовать функции, которые ещё не объявлены.

Как и в любом языке программирования в языке С++ присутствуют ключевые слова, отвечающие за описание типов переменных:

1. Int – требуется для указания на то, что переменная целочисленного типа;

2. Float – это ключевое слово, использующаяся для указания вещественного типа данных;

3. Char[] – эта лексема указывает на символьный тип (Строку);

4. Char – указывает на один символ;

5. Bool – Логическая переменная, хранящая true или false.

Условные операторы в языке C представлены ключевым словом If, далее идёт блок действий в фигурных скобках.

Для описания циклических конструкций языке C++ используются следующие лексемы:

1. While – эта лексема используется для циклов с предусловием;

2. Do...while – эта конструкция, используемая для циклов с постусловием;

3. For – это ключевое слово требуется для циклов с заданным числом повторений.

Для логических выражений в языке C++ используются символы:

1. && – это логическое И;

2. || – это логическое ИЛИ;

3. ! – это логическое НЕ.

Для ввода и вывода на экран существуют cout<< для вывода и cin>> для ввода.

1.4. Перевод из языка Pascal в алгоритмический язык и в язык C

1.4.1. Функции обработки

Лексема – это структурная единица языка, которая состоит из элементарных символов языка и не содержит в своём составе других

структурных единиц языка. Лексемами языков программирования являются идентификаторы, константы, ключевые слова языка, знаки операций и т.п.

Функции обработки представляют собой анализ лексем, найденных в тексте исходной программы лексем. Этот перечень лексем можно представить в виде списка лексем.

Каждой лексеме в списке лексем соответствует некая уникальная лексема другого языка, зависящая от типа лексемы.

Результатом работы функций обработки является перечень всех найденных в тексте исходной программы лексем.

1.4.2. Анализ ошибок

Анализ кода - один из самых надежных методов выявления дефектов. Он заключается в чтении программой исходного кода и высказывании рекомендаций по его улучшению. В процессе чтения кода выявляются ошибки или участки кода, которые являются ошибочными.

Функции обработки ошибок получают строку лексем от лексем исходного текста программы и проверяют, соответствует ли эта строка грамматике исходного языка.

В обработку ошибок входит генерация сообщений обо всех выявленных ошибках, причём достаточно внятных и полных, а кроме того, функции обработки ошибок должны уметь обрабатывать обычные, часто встречающиеся ошибки и продолжать работу с оставшейся частью программы.

2. Практическая часть

2.1. Программная реализация

Конвертация основана на соответствии ключевых слов и конструкций из данных языков программирования. При считывании слов, введенных языке Pascal, программа ищет им соответствие на алгоритмическом языке, а затем приводит синтаксис кода Pascal к синтаксису алгоритмического языка. Аналогичные действия производятся в паре «алгоритмический язык → C++».

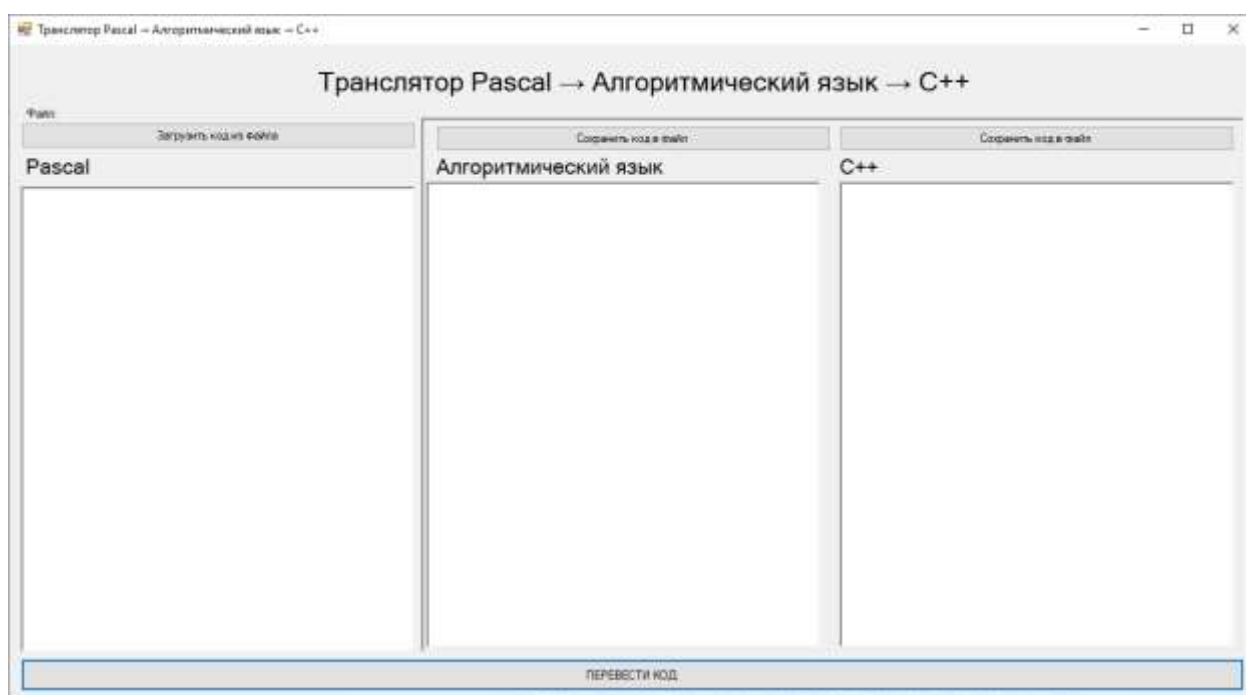


Рисунок 1 – Интерфейс программы

Левое окно позволяет вводить код на языке Pascal. Также возможно считывание кода в поле из текстового файла при нажатии на соответствующую кнопку. Окно по середине отвечает за вывод результата конвертации на алгоритмическом языке, а правое за результат конвертации в C++. Все результаты конвертации можно сохранить в текстовый файл нажав соответствующие кнопки.

Для того, чтобы программа корректно выполнила конвертацию, при написании кода на языке Pascal следует самостоятельно проверять логику работы программы. Все команды, кроме циклов и условий, должны прописываться по одному, это позволяет расширить функционал работы программы и производить конвертацию отдельных ключевых слов.

После завершения ввода требуется нажать кнопку «Перевести код», которая запустит программу конвертации.

В данном примере(рис.3) показаны возможности программы, такие как:

1. Инициализация переменных и массивов.
2. Реализация ввода/вывода.
3. Реализация цикла for.
4. Работа с массивами.
5. Реализация условного оператора if.

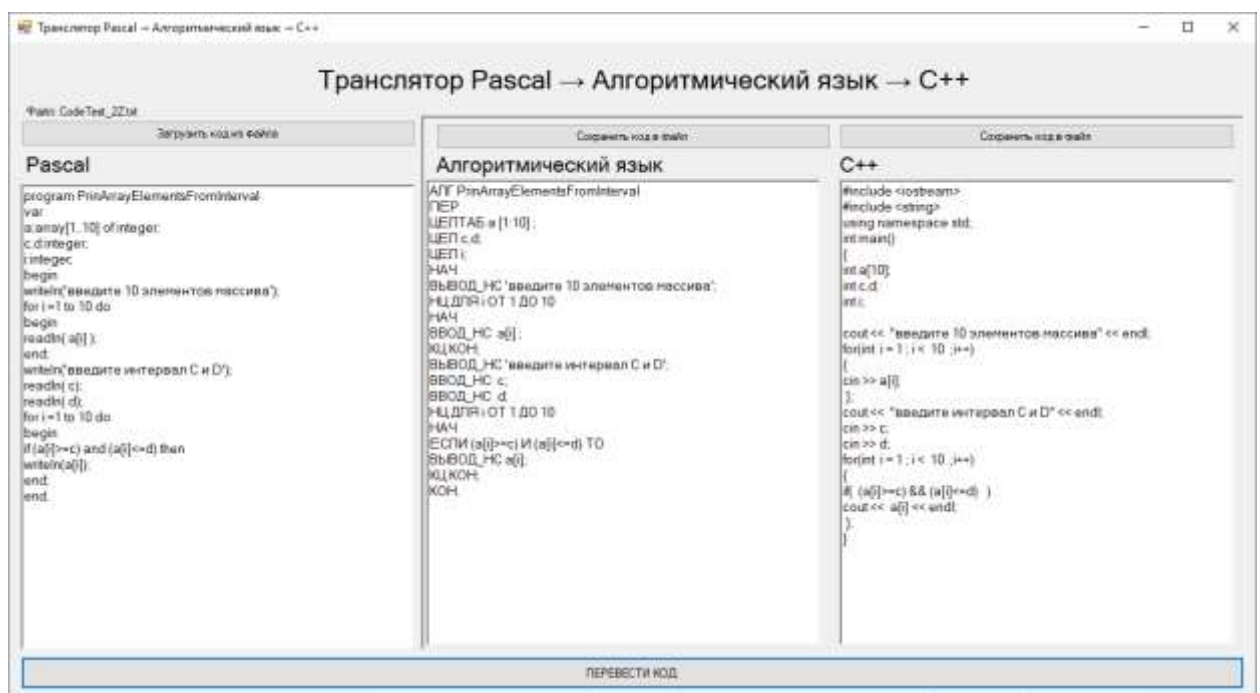


Рисунок 2 – Результат работы программы

Pascal

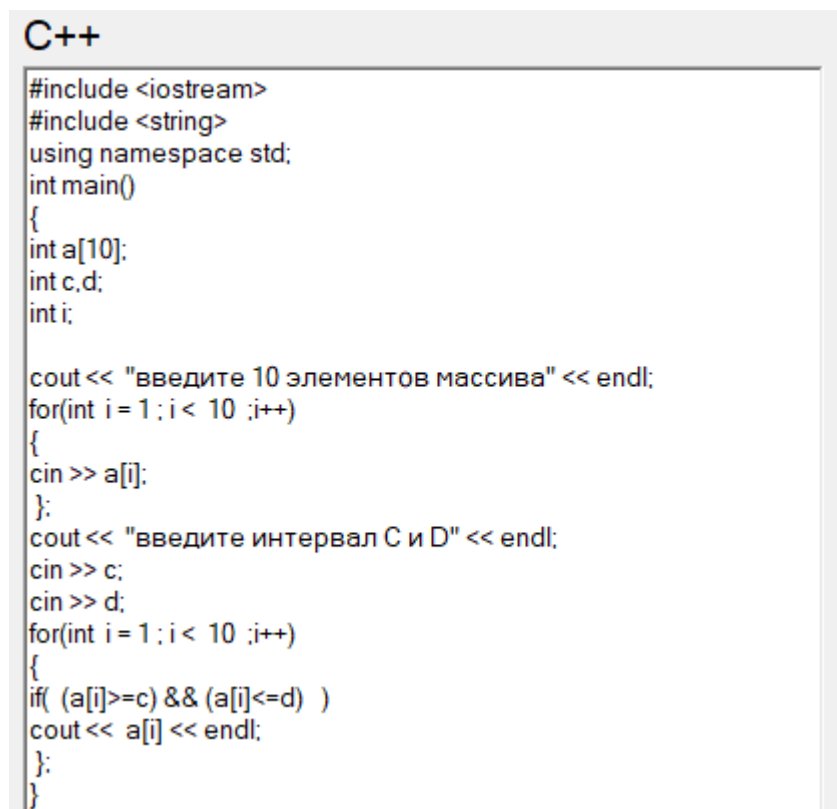
```
program PrinArrayElementsFromInterval
var
a:array[1..10] of integer;
c,d:integer;
i:integer;
begin
writeln('введите 10 элементов массива');
for i:=1 to 10 do
begin
readln( a[i] );
end;
writeln('введите интервал C и D');
readln( c);
readln( d);
for i:=1 to 10 do
begin
if (a[i]>=c) and (a[i]<=d) then
writeln(a[i]);
end;
end.
```

Рисунок 3 – Результат работы программы (изначальный код на Pascal)

Алгоритмический язык

```
АЛГ PrinArrayElementsFromInterval
ПЕР
ЦЕЛТАБ а [1:10];
ЦЕЛ с, d;
ЦЕЛ i;
НАЧ
ВЫВОД_НС 'введите 10 элементов массива';
НЦ ДЛЯ i ОТ 1 ДО 10
НАЧ
ВВОД_НС а[i];
КЦ КОН;
ВЫВОД_НС 'введите интервал C и D';
ВВОД_НС с;
ВВОД_НС d;
НЦ ДЛЯ i ОТ 1 ДО 10
НАЧ
ЕСЛИ (а[i]>=с) И (а[i]<=d) ТО
ВЫВОД_НС а[i];
КЦ КОН;
КОН.
```

Рисунок 4 – Результат работы программы (конвертация в алгоритмический язык)



```

C++
#include <iostream>
#include <string>
using namespace std;
int main()
{
    int a[10];
    int c,d;
    int i;

    cout<< "введите 10 элементов массива" << endl;
    for(int i = 1 ; i < 10 ;i++)
    {
        cin >> a[i];
    };
    cout<< "введите интервал C и D" << endl;
    cin >> c;
    cin >> d;
    for(int i = 1 ; i < 10 ;i++)
    {
        if( (a[i]>=c) && (a[i]<=d) )
        cout<< a[i] << endl;
    };
}

```

Рисунок 5 – Результат работы программы (конвертация в C++)

Также, на этапе анализа оригинального кода на Pascal, предусмотрена проверка кода на лексические и синтаксические ошибки.

Если во время лексического анализа будет обнаружен код, не содержащий в себе ни одного известного ключа кода Pascal, появится сообщение об лексической ошибке, а участок кода, содержащий ошибку, будет выделен красным цветом.

При успешной проверке на лексические ошибки следует проверка на синтаксические ошибки. Например, если в коде присваивания переменной «s:=0;» убрать «s» или «0», то появится соответствующее предупреждение об синтаксической ошибки присваивания значения переменной, а участок кода, содержащий ошибку, будет выделен красным цветом.

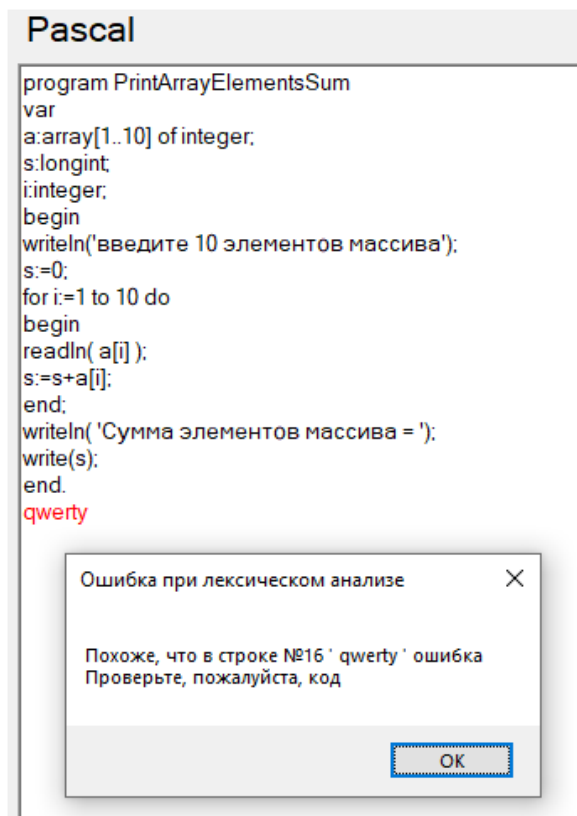


Рисунок 6 – Сообщение о лексической ошибке

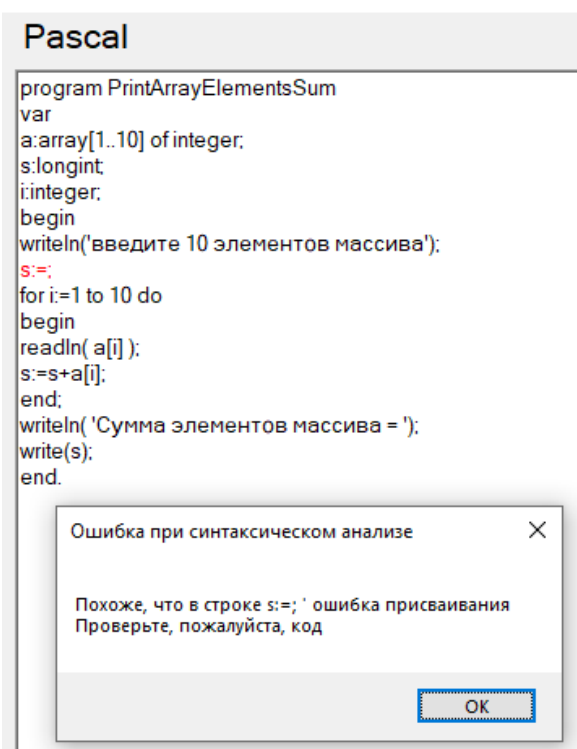


Рисунок 7 – Сообщение о синтаксической ошибке

Ниже приведены участки кода, показывающие операции конвертации:

Соответствие типов данных для переменных:

```
//типы переменных-----
if (code.Contains("integer")&&(!code.Contains("array")))
{
    NoKeys = false;
    try
    {
        code = code.Replace("integer", "ЦЕЛ");
        AlgCodeLines[line] = code;
    }
    catch
    {
        AlgCodeLines.Add(code);
    }
}
else
if (code.Contains("float") && (!code.Contains("array")))
{
    NoKeys = false;
    try
    {
        code = code.Replace("float", "Вещ");
        AlgCodeLines[line] = code;
    }
    catch
    {
        AlgCodeLines.Add(code);
    }
}
else
if (code.Contains("string") && (!code.Contains("array")))
{
    NoKeys = false;
    try
    {
        code = code.Replace("string", "СТР");
        AlgCodeLines[line] = code;
    }
    catch
    {
        AlgCodeLines.Add(code);
    }
}
```

Приведение синтаксиса вывода данных в C++:

```
//поменять синтаксис вывода
public void ChangeWrite_Algorithm(string code, int line)
{
    if(code.Contains("coutendl"))
    {
        code = code.Replace("coutendl", "");
        code = code.Replace(";", "");
        code = "cout << " + code + " << endl;";

        SiPlusCodeLines[line] = code;
    }
}
```

```

        WriteSiPlusCodeToTextBox();
    }
    else
    if(code.Contains("cout"))
    {
        code = code.Replace("cout", "");
        code = "cout << " + code;

        SiPlusCodeLines[line] = code;

        WriteSiPlusCodeToTextBox();
    }
}

```

Приведение синтаксиса для циклов в C++:

```

//ПОМЕНЯТЬ СИНТАКСИС ЦИКЛОВ
public void ChangeCycle_Alg(string code, int line)
{
    if(code.Contains("for"))
    {
        code = code.Replace("НЦ", "");
        code = code.Replace("for", "");
        code = "int" + code;
        code = code.Replace("ОТ", "=");
        code = code.Replace("ДО", "; i < ");
        code = "for(" + code + ";i++)";

        SiPlusCodeLines[line] = code;
    }
    if (code.Contains("while"))
    {
        code = code.Replace("НЦ", "");
        code = code.Replace("while", "");
        code = "while( " + code + " )";

        SiPlusCodeLines[line] = code;
    }
    if(code.Contains("}."))
    {
        code = code.Replace("}.", "");

        SiPlusCodeLines[line] = code;
    }
}

```

Таким образом была выстроена работа конвертора, основанная на соответствии трёх языков программирования.

ЗАКЛЮЧЕНИЕ

В ходе выполнения курсовой работы были получены навыки по проектированию программ, их отладке и документированию.

В результате работы была написана программа – транслятор с языка Pascal на алгоритмический язык и, из алгоритмического языка на язык C++, с использованием высокоуровневого объектно-ориентированного языка программирования C#.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Карпов, Ю.Г. Теория и технология программирования. Основы построения трансляторов: Учеб. пособие / Ю.Г Карпов Санкт-Петербург: БХВ-Петербург 2005. -272с.
2. Свердлов, С. З. Языки программирования и методы трансляции: учеб. Пособие / С.З Свердлов - Санкт-Петербург : Питер, 2007. - 638 с.
3. Яковлева Л. Л Информатика и программирование/Л. Л Яковлева Учеб. пособие. В 2 ч. Ч. 1. - Чита : ЗабГУ, 2014. - 213 с.
4. Черпаков И. В Основы программирования / И. В Черпаков Учебник и практикум - М. : Издательство Юрайт, 2017. – 219с.
5. Себеста, Роберт У Основные концепции языков программирования / пер. с англ. - 5-е изд. - Москва : Вильямс, 2001. - 672с.
6. Википедия – Свободная энциклопедия [Электронный ресурс] – URL: [https://ru.wikipedia.org/wiki/Паскаль_\(язык_программирования\)](https://ru.wikipedia.org/wiki/Паскаль_(язык_программирования)) (Дата обращения: 20.12.2021).
7. Википедия – Свободная энциклопедия [Электронный ресурс] – URL: https://ru.wikipedia.org/wiki/Алгоритмический_язык (Дата обращения: 21.12.2021).
8. Википедия – Свободная энциклопедия [Электронный ресурс] – URL: <https://ru.wikipedia.org/wiki/C%2B%2B> (Дата обращения: 22.12.2021).