МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Забайкальский государственный университет»
(ФГБОУ ВО «ЗабГУ»)

Энергетический факультет
Кафедра информатики, вычислительной техники и прикладной математики

ОТЧЕТ

по производственной практике
(практика по получению профессиональных умений и опыта профессиональной деятельности)

в ООО «Агентство Джин»
(полное наименование организации)

обучающегося Долгова Александра Артемовича
(фамилия, имя, отчество)

Курс 3 Группа ИВТ-18-2

Направление подготовки 09.03.01 Информатика и вычислительная техника
(шифр, наименование)

Направленность ОП Программное обеспеченне вычислительной техники и автоматизированных систем

Руководитель практики от университета _____ старший преподаватель Палкин Г. А.
(Ученая степень, должность, Ф.И.О.)

Руководитель практики от предприятия директор организации ООО "Джин"
Дерябин А.В.
(должность, Ф.И.О.) подпись, печать

3. Оценка работы студента на практике

Заключение руководителя практики от профильной организации о работе студента

Руководитель практики
от профильной организации _____ Деревцов А.В.
(подпись) (Ф.И.О.)

4. Результаты практики

Заключение руководителя практики от кафедры о работе студента

Работа выполнена с замечаниями и недостаточной теории и практического применения результатов

Руководитель практики
от кафедры _____ /Палкин Г. А.
(подпись) (Ф.И.О.)

Оценка при защите хорошо

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Забайкальский государственный университет»
(ФГБОУ ВО «ЗабГУ»)
Факультет Энергетический
Кафедра ИВТ и ПМ

Дневник прохождения практики

по производственной практике
(практика по получению профессиональных умений и опыта профессиональной деятельности)

Студента 3-го курса, группы ИВТ-18-2 очной формы обучения

Направление подготовки (специальность) 09.03.01 Информатика и вычислительная техника
Фамилия Долгов
Имя, отчество Александр Артемович
Сроки практики 12.07.21-25.07.21

Руководитель практики от кафедры: старший преподаватель Палкин Г. А., 89243716205
(должность, звание, степень, фамилия, имя, отчество, номер телефона)

Профильная организация ООО "Агентство Дизайн"
(полное название предприятия/организации, на которое направлен студент для прохождения практики)

Руководитель от профильной организации: директор организации ООО "Дизайн" Деревцов А.В., 89144680000
(должность, фамилия, имя, отчество, номер телефона)

Печать отдела кадров профильной организации

«Утверждаю»

Зав. кафедрой_____

«____»_____ 20__ г.

1. Рабочий план проведения практики

Дата или день	Рабочий план	Отметка о выполнении
12.07-14.07	Знакомство со спецификой работы предприятия. Выявление проблем и оценка актуальности их решения.	
15.07-17.07	Постановка целей и задач работы. Проработка требований к реализуемому программному продукту.	
18.07-19.07	Обзор возможных путей решения поставленных задач. Анализ аналогов.	
20.07-22.07	Выбор средств реализации работы. Разработка интерфейса и основных алгоритмов.	
23.07-25.07	Анализ полученных результатов. Написание отчета.	

2. Индивидуальное задание на практику
(составляется руководителем практики от кафедры)

В рамках выполнения практики по получению профессиональных умений и опыта профессиональной деятельности студенту необходимо оценить существующие производственные или информационные процессы на профильном предприятии, с целью выявления актуальных задач, требующих решения путем разработки авторского программного продукта. Далее необходимо оценить рассматриваемую предметную область, выявить основные требования к будущему продукту и провести обзор аналогов. После этого выбираются оптимальный способ и средства реализации поставленной задачи, осуществляется реализация программного продукта. По результатам работы составляется отчет.

Руководитель практики
от кафедры _____ /Палкин Г. А.
(подпись) (Ф.И.О.)

Руководитель практики
от профильной организации _____ /Деревцов А.В.
(подпись) (Ф.И.О.)

РЕФЕРАТ

Пояснительная записка 26 – страницы, 20 – рисунков, 2 – приложений.

ТРАНСЛЯТОР, PASCAL, C, ЛЕКСИЧЕСКИЙ АНАЛИЗАТОР, СИНТАКСИЧЕСКИЙ АНАЛИЗАТОР, ГРАММАТИКА, ЛЕКСЕМА.

В работе описывается создание транслятора с языка Pascal на язык C. Разработка транслятора в данной работе производится на высокоуровневом объектно-ориентированном языке C#. В работе рассматриваются понятие транслятора и принципы построения транслятора. Также есть руководство пользователя.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Pascal был разработан Никлаусом Виртом в 1970; вопреки расхожему мнению, он не был исключительно учебным языком, а предназначался для практического применения. Прототипом послужил Algol Первоначально язык компилировался в байт-код, подобно языку Java. Особенностями языка являются строгая типизация и наличие средств структурного (процедурного) программирования. Pascal был одним из первых таких языков. По мнению Н. Вирта, язык должен был способствовать дисциплинированию программирования, поэтому, наряду со строгой типизацией, в Pascal сведены к минимуму возможные синтаксические неоднозначности, а сам синтаксис интуитивно понятен даже при первом знакомстве с языком. Это упрощает написание компиляторов языка. Кроме того, язык предоставлял ряд встроенных структур данных: записи, массивы, файлы, множества и указатели.

В данной работе рассматривается процесс разработки транслятора с языка программирования Pascal в язык программирования C.

1 Постановка и анализ задачи

1.1 Описание предметной области

Транслятор – это программа, которая считывает текст программы, написанной на одном языке (входном), и транслирует (переводит) его в эквивалентный текст на другом языке (выходном).

Важным пунктом в определении транслятора является эквивалентность исходной и результирующей программ. Эквивалентность этих двух программ означает совпадение их смысла с точки зрения семантики входного языка и семантики выходного языка. Без выполнения этого требования сам транслятор теряет всякий практический смысл.

Одна из важных ролей транслятора состоит в сообщении об ошибках в исходной программе, обнаруженных во время трансляции.

Процесс отображение входной программы в эквивалентную ей выходную программу состоит из двух частей: анализ и синтез.

Анализ разбивает входную программу на составные части и накладывает на них грамматическую структуру. Затем использует эту структуру для создания промежуточного представления входной программы. Если анализ обнаруживает, что входная программа неверно составлена синтаксически или семантически, он должен выдать информативные сообщения об этом, чтобы пользователь мог исправить обнаруженные ошибки. Анализ также собирает информацию об исходной программе и сохраняет её в структуре данных, называемой таблицей идентификаторов (символов), которая передаётся вместе с промежуточным представлением синтезу.

Синтез строит требуемую целевую программу на основе промежуточного представления и информации из таблицы идентификаторов.

Анализ часто называют начальной стадией, а синтез – заключительной.

Если посмотреть на процесс трансляции более детально, можно увидеть, что он представляет собой последовательность фаз, каждая из которых преобразует одно из представлений входной программы в другое. На практике некоторые фазы могут быть объединены, а межфазное промежуточное представление может не строиться явно. Таблица идентификаторов, в которой хранится информация обо всей входной программе, используется всеми фазами транслятора.

Первая фаза трансляции называется лексическим анализом или сканированием. Лексический анализатор считывает поток символов, составляющих входную программу, и группирует эти символы в значащие последовательности, называемые лексемами. Лексемы передаются последующей фазе транслятора, синтаксическому анализу.

Вторая фаза трансляции – синтаксический анализ или разбор. Во время выполнения этой фазы используются лексемы, полученные от лексического анализатора, для создания древовидного промежуточного представления программы, которое описывает грамматическую структуру потока лексем. Типичным промежуточным представлением является синтаксическое дерево, в котором каждый внутренний узел представляет операцию, а дочерние узлы – аргументы этой операции.

После синтаксического анализа исходной программы трансляторы с помощью полученного ранее синтаксического дерева генерируют код на целевом машинном языке, и трансляция завершается.

Таблица идентификаторов представляет собой структуру данных, содержащую записи для каждого имени переменной с полями для атрибутов имени. Структура данных должна быть разработана таким образом, чтобы позволять транслятору быстро находить запись для каждого имени, а также быстро сохранять данные в записи и получать их из неё.

1.2 Постановка задачи

Разрабатываемый транслятор будет иметь возможность выполнять трансляцию с Pascal в C.

При трансляции транслятор будет выполнять фазы лексического анализа, синтаксического анализа, а также генерации целевого исходного кода. Важно отметить, что на каждой из фаз транслятор должен осуществлять жёсткий контроль ошибок в анализируемом им исходном коде и по мере их нахождения уведомлять об этом пользователя.

1.3 Средства реализации

Для реализации разрабатываемого транслятора был выбран язык программирования высокого уровня C#.

1.4 Правила построения программ на языке Pascal

Программист вводит текст программы, произвольно располагая строки на экране. Отступ слева выбирает сам программист, чтобы программа была более читабельной. В одной строке допускается писать несколько операторов. Длинные операторы можно переносить на следующую строку. Перенос допускается в любом месте, где можно сделать пробел. Максимальная длина строки не должна превышать 127 символов. Из соображений наглядности, удобства просмотра и отладки программы рекомендуется длину строки ограничивать 80 символами. Программы имеют жесткую структуру.

Фрагмент программы	Содержание	Примечание
Заголовок	Program <имя программы>	Необязательный
Раздел 0	USES, описание модулей, библиотек	Разделы описаний программного блока
Раздел I	LABEL, описание меток	
Раздел 2	CONST, описание констант	
Раздел 3	TYPE, описание типов данных	
Раздел 4	VAR, описание переменных	
Раздел 5	PROCEDURE, FUNCTION - описание процедур и функций, используемых в программе	
Раздел 6	BEGIN ... END. - тело программы	Раздел исполняемых операторов

Рисунок 1 – Структура языка Pascal

Синтаксические правила построения предложений языка можно описывать следующими способами:

схемой (форматом предложения или раздела). В учебном процессе выбран именно этот способ, поскольку он наиболее понятен начинающему программисту;

синтаксической диаграммой. Этот способ детально формализует синтаксис предложения и используется разработчиками трансляторов с языка Паскаль;

порождающими правилами РАСШИРЕННЫХ БЭКУСА-НАУРА ФОРМ (РБНФ). Это весьма компактный и в то же самое время наглядный способ записи языковых конструкций. Этот способ используется в статьях и научных разработках.

Соглашение	Толкование
Угловые скобки < >	Угловые скобки заключают в себе элемент синтаксиса, который Вы должны задать. Текст внутри угловых скобок характеризует элемент, однако, не описывает синтаксис этого элемента
Квадратные скобки [ ]	Квадратные скобки в синтаксических конструкциях заключают в себе один или несколько необязательных элементов
Вертикальная черта |	Разделяет два альтернативных элемента синтаксической конструкции, один из которых нужно выбрать
Фигурные скобки { }	Фигурные скобки заключают в себе несколько элементов, разделенных '|'. Из них нужно выбрать один
Многоточие …	Многоточие показывает, что можно повторить некоторый элемент один и более раз

Рисунок 2 – Структура языка Pascal

2 Анализ данных

Данные, с которыми работает транслятор можно разделить на несколько групп. Входные данные – данные, поступающие от пользователя транслятору. Промежуточные данные – это данные, используемые транслятором во время работы. Выходные данные – данные, поступающие от транслятора пользователю.

2.1 Входные данные

В качестве входных данных для разрабатываемого транслятора выступают графические элементы и переменные в Pascal или C.

2.2 Выходные данные

В качестве выходных данных для разрабатываемого транслятора выступает файлы с исходным кодом на языке программирования C, которые он генерирует в качестве результата, выполняя трансляцию исходного кода, полученного от пользователя.

3 Программная реализация

В ходе разработки программы был разработан упрощенный вариант графического интерфейса для редактирования кода.

3.1 Лексический анализатор

Первый этап перевода называется лексическим анализом или сканированием. Лексический анализатор (сканер) считывает поток символов, составляющих исходную программу, и группирует эти символы в значимые последовательности, называемые лексемами.

Лексема – это структурная единица языка, состоящая из элементарных символов языка и не содержащая других структурных единиц языка. Лексемы языка программирования – это идентификаторы, константы, ключевые слова языка, знаки операций и т. п.

Лексический анализатор любого транслятора может быть реализован в виде детерминированного конечного автомата.

Для разрабатываемого транслятора необходимо создать два лексических анализатора, как для языка C, так и для языка Pascal.

3.2 Синтаксический анализатор

Синтаксический анализатор – это часть транслятора, которая отвечает за идентификацию и проверку синтаксических конструкций входного языка. Синтаксический анализатор получает строку токенов от лексического анализатора и проверяет, может ли эта строка токенов порождаться грамматикой входного языка. Другая функция синтаксического анализатора является генерация сообщений обо всех выявленных ошибках, причём довольно внятных и полных, и кроме того, синтаксический анализатор должен уметь обрабатывать обычные, часто встречающиеся ошибки и

продолжать работу с оставшейся частью программы. В случае корректной программы синтаксический анализатор строит дерево разбора и передаёт его следующей части транслятора для дальнейшей обработки.

Перед синтаксическим анализатором стоят две основные задачи: проверить правильность конструкций программы, которая представляется в виде уже выделенных слов входного языка, и преобразовать её в вид, удобный для дальнейшей семантической (смысловой) обработки и генерации кода. Одним из способов такого представления является дерево синтаксического разбора.

4 Техническое задание

4.1 Введение

Приложение транслятор является программой, которая считывает текст программы, написанной на одном языке (входном), и транслирует (переводит) его в эквивалентный текст на другом языке (выходном).

4.2 Требования к функциональным характеристикам

Программа должна выполнять трансляцию текста программы с языка программирования Pascal в текст программы на язык программирования C.

4.3 Требования к надёжности

В случае ошибок в работе приложения или некорректности данных, предоставляемых пользователем, программа должна вывести уведомление с информацией о действиях необходимых для исправления полученной ошибки.

4.4 Требования к информационной и программной совместимости

Для нормальной работы приложение должно функционировать на компьютере под управлением семейства Windows.

5 Руководство пользователя

Разрабатываемый в данной работе транслятор является программой с графическим интерфейсом.

Для запуска программы необходимо нажать на исполняемый файл cnvrt.sln и после запустить программу из Visual Studio.

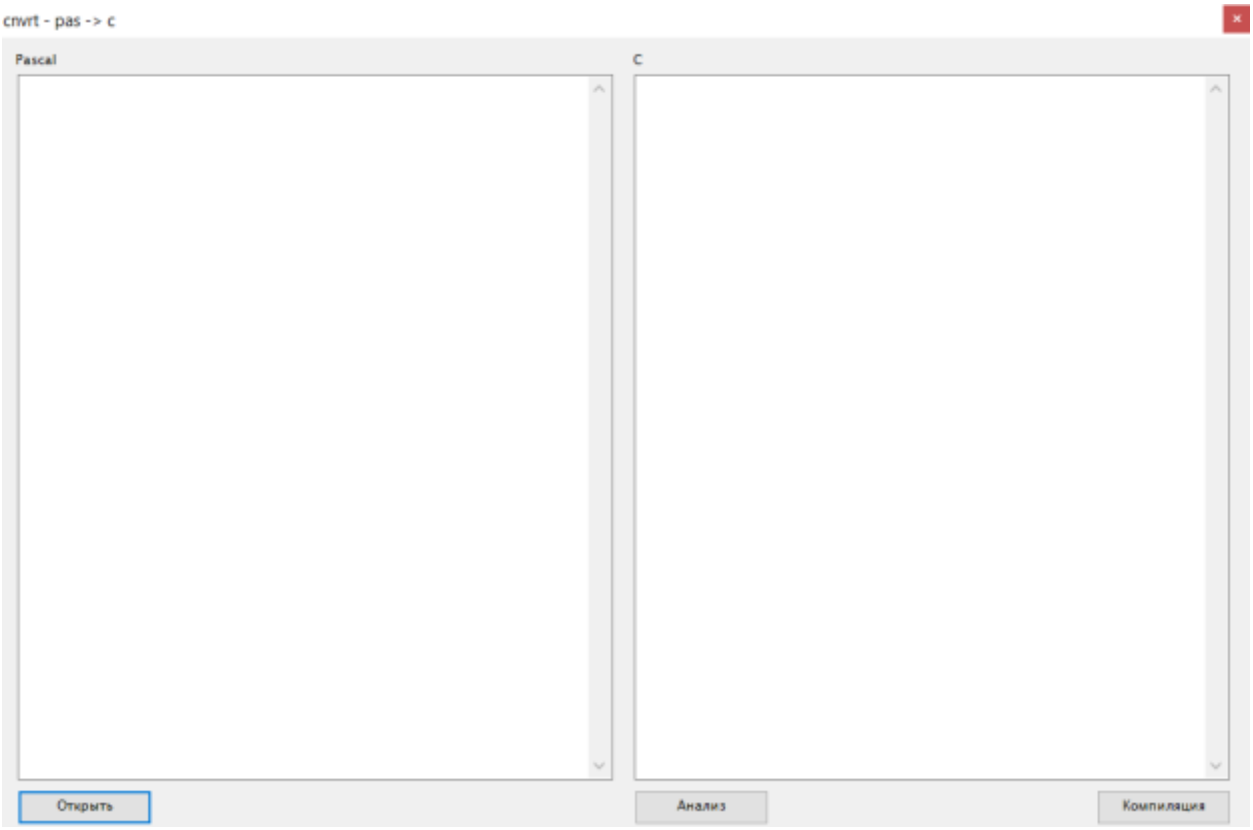

Рисунок 3 – Главное окно программы

В начале необходимо написать код программы на языке Pascal или открыть готовый файл с расширением «.pas».

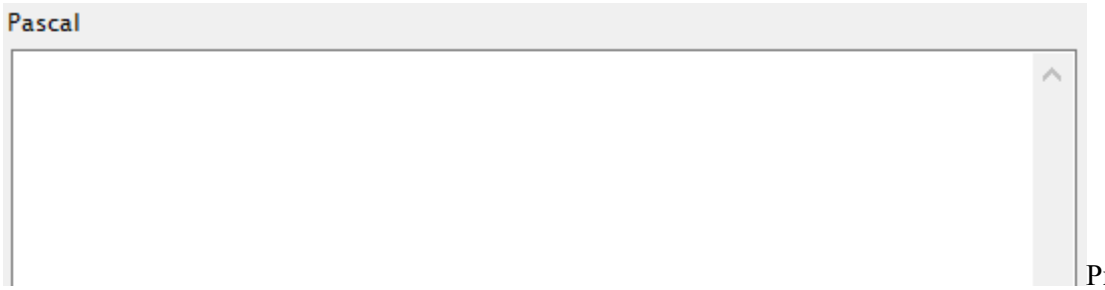

Рисунок 4 – Вставка или редактирование кода Pascal

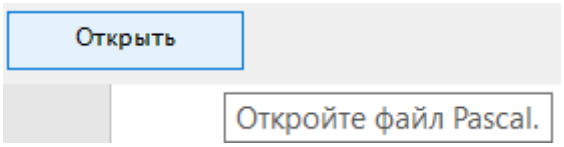

Рисунок 5 – Открытие готовых программ на языке Pascal

После того как пользователь открыл или написал код на языке Pascal, он может провести анализ программы или скомпелировать код Pascal в код C.

```
35      -       program
60      -       test
79      -       ;
48      -       var
60      -       x
78      -       ,
60      -       y
78      -       ,
60      -       z
80      -       :
52      -       integer
79      -       ;
4       -       begin
60      -       x
66      -       :=
85      -       50
79      -       ;
60      -       y
66      -       :=
85      -       30
79      -       ;
60      -       z
66      -       :=
60      -       x
27      -       mod
60      -       y
79      -       ;
13      -       end
77      -       .
13      -       sfarsit
```

Рисунок 6 – Анализ кода Pascal

Рисунок 7 – Компиляция кода

После того, как программа скомпелировала код, пользователь может скопировать его.

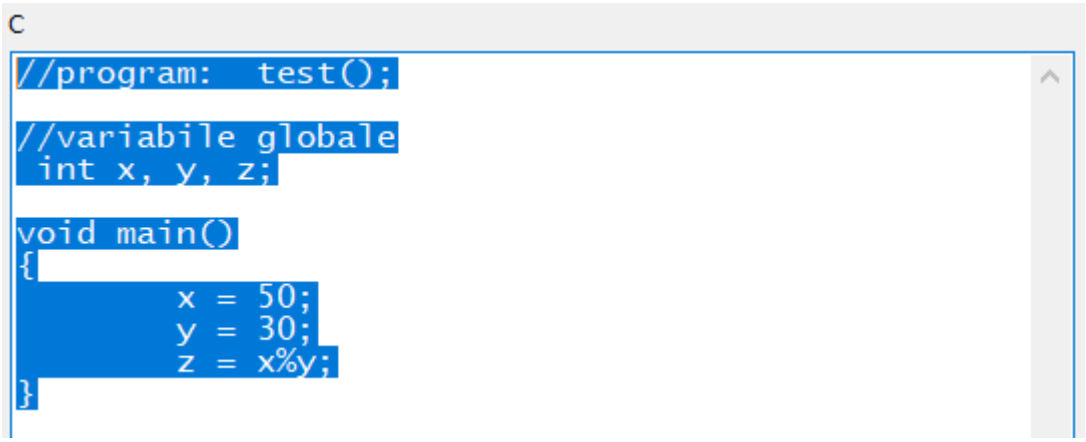

Рисунок 8 – Код на языке С

ЗАКЛЮЧЕНИЕ

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

В результате работы была написана программа – транслятор с языка программирования Pascal в язык программирования C, с использованием высокоуровневого объектно-ориентированного языка программирования C#.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Wikipedia. Транслятор [Электронный ресурс] // Электрон. текстовый документ – Режим доступа: https://ru.wikipedia.org/wiki/Транслятор свободный

2. Wikipedia. Лексический анализ [Электронный ресурс] // Электрон. текстовый документ – Режим доступа: https://ru.wikipedia.org/wiki/Лексический_анализ свободный

3. Wikipedia. Синтаксический анализатор [Электронный ресурс] // Электрон. текстовый документ – Режим доступа: https://ru.wikipedia.org/wiki/Синтаксический_анализатор свободный

4. Компиляторы. Принципы, технологии и инструментарий: учебник / Альфред. В. Ахо [и др.]; под ред. С.Н. Тригуб. – Москва: Вильямс-Издат, 2008. – С. 75 - 152.

5. Молдованова О.В. Языки программирования и методы трансляции: учеб. пособие / О.В. Молдованова. – Новосибирск: ФГОБУ ВПО «СибГУТИ», 2012. – 39 с.

6. Молдованова О.В. Языки программирования и методы трансляции: учеб. пособие / О.В. Молдованова. – Новосибирск: ФГОБУ ВПО «СибГУТИ», 2012. – С. 61 - 65.

7. Молдованова О.В. Языки программирования и методы трансляции: учеб. пособие / О.В. Молдованова. – Новосибирск: ФГОБУ ВПО «СибГУТИ», 2012. – С. 79 - 97.

ПРИЛОЖЕНИЕ

ПРИЛОЖЕНИЕ А: Пример Pascal кода

Pascal

```
program HelloWorld;
begin
  write("test");
end.
```

ПРИЛОЖЕНИЕ Б: Пример сгенерированного C кода

C

```
//program:  helloworld();

void main()
{
        printf("test ") ;
}
```