МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение  
высшего образования

**«Дальневосточный федеральный университет»**

**ИНСТИТУТ МАТЕМАТИКИ И КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ  
Департамент программной инженерии и искусственного интеллекта**

РАЗРАБОТКА ТРАНСЛЯТОРА, ПЕРЕВОДЯЩЕГО ПОДМНОЖЕСТВО ЯЗЫКА C++ В ЭКВИВАЛЕНТНОЕ ПОДМНОЖЕСТВО ЯЗЫКА PascalABC.NET

**КУРСОВОЙ ПРОЕКТ**

по дисциплине «Теория вычислительных процессов и структур»

по образовательной программе подготовки бакалавров по направлению 02.03.03 "Математическое обеспечение и администрирование информационных систем"

Выполнили студенты гр.

Б9119-02.03.03техпро

\_\_\_\_\_\_\_\_\_\_\_Болычев Л.И.

\_\_\_\_\_\_\_\_\_\_\_Бондаренко Д.С.

\_\_\_\_\_\_\_\_\_\_\_Ли Е.В.

\_\_\_\_\_\_\_\_\_\_\_Косицына А.М.

Руководитель:

Ассистент департамента ПИиИИ Рогулин Р.С

\_\_\_\_\_\_\_\_\_\_\_\_ «\_\_\_» \_\_\_\_\_\_\_\_\_2022 г.

(подпись)

Защищён с оценкой \_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_

(подпись) И.О. Фамилия

«\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2022 г.

г. Владивосток

2022 г.

Оглавление

[Введение 3](#_Toc106830213)

[1 Неформальная постановка задачи 5](#_Toc106830214)

[2 Синтаксис входного языка 7](#_Toc106830215)

[3 Контекстные условия 12](#_Toc106830216)

[4 Таблица соответствия 14](#_Toc106830217)

[Заключение 17](#_Toc106830218)

[Список литературы 18](#_Toc106830219)

# Введение

В наши дни сложно представить себе программирование без использования трансляторов, компиляторов и интерпретаторов. Из-за того, что компьютер понимает лишь машинный код (то есть последовательность нулей и единиц), а писать программы в таком виде слишком неудобно и сложно, ещё на заре эры компьютерных технологий появились понятные человеку языки программирования, на которых пишутся программы, а затем преобразуются в понятный компьютеру машинный код. А для осуществления этого преобразования как раз и нужны трансляторы, компиляторы и интерпретаторы.

Рассмотрим эти понятия подробнее:

Транслятор (англ. translator — переводчик) — это программа-переводчик. Она преобразует программу, написанную на одном из языков высокого уровня, в программу, состоящую из машинных команд.

Компилятор (англ. compiler — составитель, собиратель) читает всю программу целиком, делает ее перевод и создает законченный вариант программы на машинном языке, который затем и выполняется.

Интерпретатор (англ. interpreter — истолкователь, устный переводчик) переводит и выполняет программу строка за строкой. В отличие от компилятора, интерпретатор не порождает на выходе программу на машинном языке. Распознав команду исходного языка, он тут же выполняет ее.

Таким образом, компиляторы и интерпретаторы являются разновидностью трансляторов.

Трансляторы сегодня используются практически везде, ибо желающих писать программы на машинном коде немного. Но ведь трансляцию можно осуществлять не только с высокоуровневого языка программирования на машинный код, но и вообще с любого языка программирования на любой другой. Поэтому может иметь смысл создание такого транслятора. Так, вспомнив времена, когда мы переходили с Паскаля на С++, и изучив теорию языков [1], наша команда решила создать транслятор, переводящий код на С++ в код на PascalABC.NET.

Целью курсового проекта является разработка проекта транслятора, который преобразует программу, содержащую подмножество языка C++, в программу, содержащую подмножество языка PascalABC.NET, генерируя эквивалентный исходный код.

Чтобы достигнуть цели, нужно решить определённые задачи.

Задачи курсового проекта:

1. Провести анализ основных конструкций входного языка.
2. Разработать грамматику подмножества языка C++.
3. Описать контекстные условия входного языка.
4. Описать соответствие конструкций входного и выходного языков.

# 1 Неформальная постановка задачи

Необходимо разработать проект транслятора из подмножества языка C++ в эквивалентное подмножество языка PascalABC.NET.

Подмножество языка C++ включает [2] [3]:

1. Типы данных: bool, char, short int, unsigned short int, int, unsigned int, float, double.
2. Операторы:
3. Арифметические:
   1. Унарные: ++, --;
   2. Бинарные: +, -, \*, /, %;
4. Логические:
   1. Унарные: !;
   2. Бинарные: &&, ||;
5. Сравнения: ==, !=, <, >, <=, >=;
6. Присваивания: =.
7. Операторы циклов: for, while, do … while.
8. Операторы ветвления: if, if … else.
9. Стандартные функции:

Таблица 1 - Стандартные функции языка C++

|  |  |
| --- | --- |
| **Функция** | **Описание** |
| int abs (int x)  long abs (long x) | Возвращает модуль целого числа x |
| float fabs (float x)  double fabs (double x) | Возвращает модуль числа x с плавающей точкой |
| float exp(float x)  double exp(double x) | Возвращает экспоненту от числа x |
| float sqrt(float x)  double sqrt(double x) | Возвращает значение квадратного корня от числа x |
| float log(float x)  double log(double x) | Возвращает натуральный логарифм от числа x |
| float pow(float x, float y)  double pow(double x, double y) | Возвращает значение xy |
| float sin(float x)  double sin(double x) | Возвращает синус числа x |
| float cos(float x)  double cos(double x) | Возвращает косинус числа x |
| float floor(float x)  double floor(double x) | Округляет число x до ближайшего меньшего целого числа |
| float ceil(float x)  double ceil(double x) | Округляет число x до ближайшего большего целого числа |

# 2 Синтаксис входного языка

На вход поступает цепочка, принадлежащая языку С++.

Грамматика входного языка:

G = (N, ∑, P, <программа>);

N = {<программа>, <подключение библиотеки>, <имя библиотеки>, <объявление переменной>, <объявление константы>, <предпроцессорное объявление>, <значение>, <объявление функции>, <параметры функции>, <тип данных>, <тип данных функции>, <символ>, <цифра>, <целое число>, <дробное число>, <число>, <число>, <строка>, <идентификатор>, <тело функции>, <возврат значения>, <блок кода>, <цикл>, <ветвление>, <бинарный арифметический оператор>, <унарный арифметический оператор>, <символьное значение>, <строковое значение>, <оператор сравнения>, <унарная арифметическая операция>, <арифметический операнд>, <арифметическое выражение>, <операнд сравнения>, <выражение сравнения>, <логический бинарный оператор>, <логическое значение>, <логический операнд>, <логическое выражение>, <инструкция>, <выражение>, <вызов функции>, <параметры вызова функции>, <оператор присваивания>, <присваивание>, <имя функции>, <имя переменной>, <имя константы>, <модификатор типа данных>, <тип данных переменной>, <главная функция>, <прочие символы>, <любая строка>, <тело цикла>, <оператор цикла>, <знак числа>, <имя или инициализация>, <следующая константа>}

∑ = {**a … z, A … Z, а … я, А … Я, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, (, ), {, }, [, ], +, -, \*, /, %, >, <, =, !, &, |, ; , “ , ‘, , , \_, # , @, $, ^, ~, №, :, ?**}; [4]

P: {<программа> ⟶ <подключение библиотеки> <программа> | <предпроцессорное объявление> <программа> |<объявление переменной> <программа> | <объявление функции><программа> | <объявление константы> <программа> | <главная функция>

<подключение библиотеки> ⟶ **#include** **<**<имя библиотеки>**>**

<имя библиотеки> ⟶ **cmath, iostream**

<объявление переменной> ⟶ <тип данных переменной> <имя или инициализация>

<имя или инициализация> ⟶ <имя переменной>**,** <имя или инициализация> | <имя переменной> **=** <значение>**,** <имя или инициализация> | <имя переменной>**;** | <имя переменной> **=** <значение>**;**

<объявление константы> ⟶ **const** <тип данных переменной> <следующая константа>

<следующая константа> ⟶ <имя переменной> **=** <значение>, <следующая константа> | <имя переменной> **=** <значение>**;**

<предпроцессорное объявление> ⟶ **#define** <идентификатор> <значение>

<значение> ⟶ <число> | <строковое значение> | <символьное значение> | <логическое значение>

<объявление функции> ⟶ <тип данных функции> <имя функции> **(**<параметры функции>**)** **{**<тело функции>**}** | <тип данных функции> <имя функции> **( )** **{** <тело функции>**}**

<параметры функции> ⟶ <тип данных переменной> <идентификатор>| <тип данных переменной> <идентификатор>**,**<параметры функции>

<тип данных> ⟶ **bool | char | int | float | double**

<модификатор типа данных> ⟶ **short | long | signed | unsigned**

<тип данных функции> ⟶ <модификатор типа данных> <тип данных> | <тип данных> | **void**

<тип данных переменной> ⟶ <модификатор типа данных> <тип данных> | <тип данных>

<символ> ⟶ **a |…| z | A |… | Z** [4]

<цифра>⟶ **0|1|2|3|4|5|6|7|8|9** [4]

<целое число> ⟶ <цифра><целое число>| <цифра>

<дробное число> ⟶ <целое число>.<целое число>

<число> ⟶ <целое число> | <дробное число>

<строка> ⟶ <символ><строка>|<цифра><строка>| <символ> | <цифра> |\_<строка>

<прочие символы> ⟶ **а| … | я | А | … | Я | \_ | ! | @ | # | $ | % | ^ | & | \* | ( | ) | - | + | [ | ] | { | } | ~ | № | ; | : | ? | = | < | >| . | ,** [4]

<любая строка> ⟶ <символ> <любая строка>| <цифра><любая строка>| <прочие символы><любая строка>| <символ> | <цифра> | <прочие символы>

<идентификатор> ⟶ \_<идентификатор>| <символ><идентификатор> | **\_** | <строка>

<тело функции> ⟶ <блок кода> <возврат значения> | <блок кода>

<возврат значения> ⟶ **return** <выражение>**;**| **return** <имя переменной>; | **return** <имя константы>;

<блок кода> ⟶ <инструкция><блок кода>|<инструкция>;

<цикл> ⟶ **while (**<логическое выражение>**)** **{**<тело цикла>**}** | **do {** <тело цикла> **} while (**<логическое выражение>**);** | **for** ( <инструкция>**;** <логическое выражение>**;** <инструкция>**) {** <тело цикла> **}**

<тело цикла> ⟶ <блок кода> <оператор цикла> <блок кода> | <оператор цикла> <блок кода> | <блок кода> <оператор цикла> | <блок кода> |<оператор цикла>

<оператор цикла> ⟶ **break; | continue;**

<ветвление> ⟶ **if (**<логическое выражение>**) {**<блок кода>**} | if** **(**<логическое выражение>**) {**<блок кода>**} else {**<блок кода>**} | if (**<логическое выражение>**) {**<блок кода>**} else** <ветвление>

<бинарный арифметический оператор> ⟶ **+ | - | / | \* | %**

<унарный арифметический оператор> ⟶ **++ | --**

<знак числа> ⟶ **+ | -**

<символьное значение> ⟶ **’**<символ>**’** | **’**<цифра>**’** | **‘**<бинарный арифметический оператор>**’** | **‘**<прочие символы>**’**

<строковое значение> ⟶ **“**<любая строка>**”**

<оператор сравнения> ⟶ **== | != | > | < | >= | <=**

<унарная арифметическая операция> ⟶ <идентификатор> <унарный арифметический оператор> | <унарный арифметический оператор> <идентификатор> | <знак числа> <идентификатор> | <знак числа> <число>

<арифметический операнд> ⟶ <число> | <имя переменной> | <вызов функции> | **(**<арифметическое выражение>**)** | <унарная арифметическая операция>

<арифметическое выражение> ⟶ <арифметический операнд> <бинарный арифметический оператор> <арифметический операнд>

<операнд сравнения> ⟶ <число> | <символьное значение> | <имя переменной> | <вызов функции>

<выражение сравнения> ⟶ <операнд сравнения> <оператор сравнения> <операнд сравнения>

<логический унарный оператор> ⟶ **!**

<логический бинарный оператор> ⟶ **&& | ||**

<логическое значение> ⟶ **true | false**

<логический операнд> ⟶ <логическое значение> | <выражение сравнения> | **(**<логическое выражение>**)** | <имя переменной> | <вызов функции>

<логическое выражение> ⟶ **!**<логический операнд> | <логический операнд> <логический бинарный оператор> <логический операнд> | <логический операнд>

<инструкция> ⟶ <присваивание>**;** | <объявление переменной> | <объявление константы> | <вызов функции>**;** | <выражение>**;** | <цикл> | <ветвление>

<выражение> ⟶ <арифметическое выражение> | <логическое выражение> | <унарная арифметическая операция> | <значение>

<вызов функции> ⟶ <имя функции> **()** | <имя функции> **(**<параметры вызова функции>**)**

<параметры вызова функции> ⟶ <выражение>| <выражение>, <параметры вызова функции>

<оператор присваивания> ⟶ **= | += | -= | \*= | /= | %=**

<присваивание> ⟶ <имя переменной> <оператор присваивания> <выражение>

<имя функции> ⟶ <идентификатор>

<имя переменной> ⟶ <идентификатор>

<имя константы> ⟶ <идентификатор>

<главная функция> ⟶ **int main () {** <блок кода> <возврат значения> **}** | **int main (int argc, char \*argv[] )** **{**<блок кода> <возврат значения>**}** }

# 3 Контекстные условия

Задание правил Бэкус-Науровой формы (далее Б.Н.Ф). позволяет описать схематически структуру тех последовательностей символов, которые должны быть восприняты, как программа на некотором языке программирования. Однако для таких последовательностей символов обычно существует ещё ряд дополнительных требований, которые невозможно описать с помощью Б.Н.Ф.-правил. Поэтому формируют дополнительно к Б.Н.Ф.-описаниям синтаксиса языка программирования ряд дополнительных условий. Контекстными условиями называют те условия, которые должны быть выполнены, чтобы последовательность символов рассматривалась как осмысленная программа. [5]

Рассмотрим характерные контекстные условия [2]:

* + - 1. Идентификатор должен описываться до того, как он будет использован в программе.
      2. Идентификатор должен быть уникальным: он должен описываться лишь один раз в рамках области видимости этого идентификатора.
      3. В случае коллизии идентификаторов из текущей области видимости и области видимости выше по иерархии используется идентификатор, объявленный в текущей области видимости.
      4. Требуется всегда указывать, к какому типу данных принадлежит тот или иной компонент.
      5. В большинстве функций стандартных библиотек можно использовать только те операнды, чьи типы данных совпадают с запросом использующей их функции или процедуры.
      6. Количество, тип и порядок следования формальных и фактических параметров функций и процедур должны совпадать.
      7. Объём используемой памяти ограничен лишь физическими возможностями техники.
      8. В арифметических выражениях операнды должны быть одного и того же типа, или иметь возможность быть приведёнными к одному и тому же типу.
      9. Операндами арифметических выражений могут быть только числовые константы, идентификаторы арифметических переменных, указатели арифметических функций.

# 4 Таблица соответствия

В данной таблице соответствия показаны элементы языка C++ [3] и эквивалентные им элементы из языка PascalABC.NET [6] [7].

*Таблица 2 – Таблица соответствия языков C++ и PascalABC.NET*

|  |  |
| --- | --- |
| **С++** | **PascalABC.NET** |
| **1. Структура программы** | |
| <Объявление переменных>;  <Объявление функций>;  int main(){  <Тело>;  } | <Объявление переменных>;  <Объявление функций>;  begin  <Тело>;  end. |
| **2. Типы данных** | |
| bool  short int или signed short int  unsigned short int  int или signed int  unsigned int  float  double  char | boolean  integer  word  longint  (отсутствует)  single  double  char |
| **3. Логические значения** | |
| true  false | True  False |
| **4. Математические операторы** | |
| +  -  \*  /  %  ++  -- | +  -  \*  div (целые) или / (вещественные)  mod  inc(x)  dec(x) |

*Продолжение таблицы 2*

|  |  |
| --- | --- |
| **5. Логические операторы** | |
| !  &&  || | not  and  or |
| **6. Операторы сравнения** | |
| ==  !=  >  <  >=  <= | =  <>  >  <  >=  <= |
| **7. Оператор присваивания** | |
| = | := |
| **8. Оператор условного перехода с неполным ветвлением** | |
| |  | | --- | | if (<булево выражение>) {  <Тело>;  } | | if (<Булево выражение>) then  begin  <Тело>;  end; |
| **9. Операторы условного перехода с полным ветвлением** | |
| if (<булево выражение>) {  <Тело>;  } else {  <Тело>;  } | if (<Булево выражение>) then  begin  <Тело>;  end  else  begin  <Тело>;  end; |
| if (<булево выражение>) {  <Тело>;  } else if(<Булево выражение>) {  <Тело>;  } | if (<Булево выражение>) then  begin  <Тело>;  end  else  begin |

*Продолжение таблицы 2*

|  |  |
| --- | --- |
|  | <Тело>;  end; |
| **10. Оператор цикла for** | |
| For (<идентификатор> = <число>; <булево выражение>;<выражение>) {  <Тело>;  } | for <идентификатор>:=<число> to <число> do  begin  <Тело>;  end; |
| **11. Оператор цикла while** | |
| While (<булево выражение>) {  <Тело>;  } | while <булево выражение> do  begin  <Тело>;  end; |
| **12. Процедура** | |
| void <идентификатор>(<формальные параметры>) {  <Тело>;  } | procedure <идентификатор> (формальные параметры);  begin  <Тело>;  end; |
| **13. Функция** | |
| <тип данных> <идентификатор>(<формальные параметры>) {  <Тело>;  } | function <идентификатор> (формальные параметры);  begin  <Тело>;  end; |

# Заключение

В рамках курсового проекта:

* Проведен анализ основных конструкций входного языка;
* Разработана грамматика подмножества языка C++.
* Сформулированы контекстные условия языка C++.
* Описано соответствие конструкций входного и выходного языка.

Таким образом, все поставленные задачи были успешно выполнены.

Цель курсового проекта достигнута.

# Список литературы

1. Артемьева И.Л. – Теория вычислительных процессов и структур. Часть 1. Языки и способы их задания. Учебно-методическое пособие. – Владивосток Изд-во Дальневост. Федер. ун-та, 2011. – 60 с.
2. Справочник по языку С++ | Microsoft Docs. [Электронный ресурс]. Режим доступа URL: <https://docs.microsoft.com/ru-ru/cpp/cpp/cpp-language-reference?view=msvc-160> (дата обращения 10.06.2022)
3. Справочник по C/C++. [Электронный ресурс]. Режим доступа URL: <http://mycpp.ru/cpp/scpp/> (дата обращения 11.06.2022)
4. Алфавит и лексемы языка С. [Электронный ресурс]. Режим доступа URL: <http://cpp-cpp.blogspot.com/2013/10/c.html> (дата обращения 11.06.2022)
5. Артемьева И.Л. – Теория вычислительных процессов и структур. Часть 2. Методы синтаксического анализа программ. Учебно-методическое пособие. – Владивосток Изд-во Дальневост. Федер. ун-та, 2011. – 56 с.
6. Справка PascalABC.NET. [Электронный ресурс]. Режим доступа <http://pascalabc.net/downloads/pabcnethelp/index.htm?page=LangGuide/index_lang_guide.html&ysclid=l4cik4icdw580251760> (дата обращения 12.06.2022)
7. Осипов А. – PascalABC.NET: Введение в современное программирование. [Электронный ресурс]. Режим доступа URL: <http://pascalabc.mmcs.sfedu.ru/downloads/OsipovBook/%D0%9A%D0%BD%D0%B8%D0%B3%D0%B0%D0%94%D0%BB%D1%8F%D0%A1%D0%B0%D0%B9%D1%82%D0%B0.pdf> (дата обращения 13.06.2022)