## Министерство науки и высшего образования Российской Федерации

## ФГБОУ ВО «Алтайский государственный университет»

## Институт математики и информационных технологий

## Кафедра информатики

## Курс «Теория алгоритмических языков и трансляторов»

Отчет о лабораторной работе №4

## Проектирование лексики языка программирования

Выполнил:

Сердюк Егор Иванович  
группа 474б

Барнаул 2020

Вариант 20

# Задача

В каждом задании описывается некоторый очень усеченный вариант языка

программирования С++. В задании указывается:

1. структура программы,

2. типы данных, которые могут использоваться в программе,

3. допустимые операции над этими данными,

4. операторы,

5. операции и операнды, из которых строятся выражения,

6. все виды констант, которые могут использоваться в выражениях.

Особенно следует отметить следующие факты:

1. во всех заданиях предполагается использование составного и пустого

оператора,

2. всегда разрешается описание глобальных данных,

3. все перечисленные элементы языка должны использоваться в программе

(например, если разрешается описание функций, то, безусловно, в перечень

операторов Вам необходимо включить вызовы функций).

Программа: главная программа языка С++. Допускается описание массивов в конструкции typedef. Разрешается описание глобальных данных.

Типы данных: int, double.

Операции: арифметические и сравнения.

Операторы: присваивания и switch, пустой и составной оператор. Операнды: простые переменные, элементы структур и константы. Константы: целые в 10 c/c, символьные, целые в 16 с/c.

# Таблица приоритетов операция

|  |  |  |
| --- | --- | --- |
| **1** | +   − | Унарный плюс и минус |
| **2** | \*   /   % | Умножение, деление и остаток |
| **3** | +   − | Сложение и вычитание |
| **4** | <   <= | Операции сравнения < и ≤ |
| >   >= | Операции сравнения > и ≥ |
| ==   != | Операции сравнения = и ≠ |

1. Минимальный уровень у всех знаков сравнения
2. Следующий уровень приоритетов у аддитивных операция
3. Максимальный уровень у мультипликативных операций, здесь же унарный + и –

Соответственно

<выражение>

<слагаемое>

<множитель>

# Построение

Gj :

<программа> -> <программа><описание>| ε

<описание> -> <описание данных>| <массив>| <главная функция>

<главная функция> -> void main()<составной оператор>

<описание данных> -> <тип><список переменных>;

<список переменных> -> <список переменных>, < переменная> | < переменная>

<переменная> -> <идентификатор> | <идентификатор> = <выражение>

< массив > -> typedef<тип> <идентификатор> <размер> | typedef <тип> <идентификатор> <размер> = {список элементов массива}

<размер> -> [<целая константа 10 c/c>]

<тип> -> int | double

<составной оператор> -> {<операторы и описания>}

<операторы и описания> -> <операторы и описания><описание данных> | <операторы и описания><операторы> | ε

<операторы> -> <присваивание>|<switch>|<составной оператор>| ;

<switch> -> switch(<выражение>)<switch тело>

<switch тело> -> {<case инструкция> <default инструкция> <case инструкция>}|;

<case инструкция> -> <case инструкция> case <константа>: <операторы и описания>| ε

<default инструкция> -> default: <операторы и описания>| ε

<присваивание> -> <переменная> = <выражение>;

<выражение> -> < выражение > < < слагаемое >|

< выражение > <= < слагаемое >|

< выражение > > < слагаемое >|

< выражение > >= < слагаемое >|

< выражение > == < слагаемое >|

< выражение > != < слагаемое >|

< слагаемое > -> < слагаемое > + < множитель >|

< слагаемое > - < множитель >|

< множитель >

< множитель > -> < множитель > \* < эл. выр. >|

< множитель > / < эл. выр. >|

< эл. выр. > | +< эл. выр. >| -< эл. выр. >

< эл. выр. > -> <идентификатор>|<константа>| <элемент массива>|(<выражение>)

<элемент массива> -> <идентификатор><размер>

<идентификатор> -> <буква> <окончание>

<окончание> -> <окончание> <буква>|

<окончание> <цифра>| ε

<константа> -> <конст. целая 10 с/с>|<конст. целая 16 с/с>|<конст. символьная>

<конст. символьная> -> ‘<символ>’

<символ> -> <буква>|<цифра>|…

<конст. целая 10 с/c> -> <конст. целая 10 с/с> <цифра>|<цифра>

<конст. целая 16 с/с> -> 0x<число в 16 с/с>

<число в 16 с/с> -> <число в 16 с/с><цифра>|<число в 16 с/с><16 с/с цифра>|<16 с/с цифра>

<цифра> -> 0|1|2|3|4|5|6|7|8|9

<буква> -> A | B | … | Z | a | b | … | z

<16 с/с цифра> -> 0 | 1 | … | 9 | A | B | … | F | a | b | … | f

Лексический уровень:

1)Идентификаторы;

2)Константы: целые в 10 c/c, символьные, целые в 16 с/c;

3)Специальные знаки: точка, точка с запятой, запятая, круглые и фигурные скобки;

4)Знаки операций >, <, >=, <=, ==, !=, +, -, \*, %;

5)Ключевые слова: switch, typedef;

Синтаксический уровень:

1)Ключевые слова: switch, typedef;

|  |  |  |
| --- | --- | --- |
| Лексические единицы языка | Тип лексемы | Символ-ограничитель |
| **Ключевые слова** |  |  |
| void | TVoid = 1 | Не буква, не цифра |
| int | TInt = 2 | Не буква, не цифра |
| double  switch  case  default | TDouble = 3  TSwitch = 4  TCase = 5  TDefault = 6 | Не буква, не цифра  Не буква, не цифра  Не буква, не цифра  Не буква, не цифра |
| **Идентификаторы** | TIdent = 20 | не буква, не цифра |
| Константа целая в 10 с/с | TConstInt10 = 30 | Не цифра |
| Константа целая в 16 с/с  Константа символьная  **Специальные знаки**  **;**  **,**  **[**  **]**  **{**  **}**  **(**  **)**  **Знаки операций**  \*  /  +  -  <  <=  >  >=  ==  !=  = | TConstInt16 = 31  TConstChar = 32  TTZpt = 40  TZpt = 41  TLKSk = 42  TPKSk = 43  TLFSk = 44  TPFSk = 45  TLSk = 46  TPSk = 47  TMult = 50  TDiv = 51  TPlus = 52  TMinus = 53  TSmall = 54 TSmallEqual = 55 TLarg = 56 TLargEqual = 57 TLogEqual = 58 TNEqual = 59  TEqual = 60 | Не цифра, не буква a-f, A-F  Любой символ  Любой символ  Любой символ  Любой символ  Любой символ  Любой символ  Любой символ  Любой символ  Любой символ  Любой символ  Любой символ  Любой символ  Любой символ  Не знак =  Любой символ  Не знак =  Любой символ  Любой символ  Любой символ  Не знак = |
| **Конец исходного модуля**  **Ошибочный символ** | TEnd = 100  TErr = 200 |  |

