Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра информатики

Дисциплина: Методы трансляции

**ОТЧЕТ**

к лабораторной работе №1

на тему

«Определение модели языка. Выбор инструментальной языковой среды»

Выполнил А. А. Сивый

Проверил Н. Ю. Гриценко

Минск 2024**СОДЕРЖАНИЕ**

1 Цели работы3

2 Подмножество языка программирования4

2.1 Числовые и строковые константы4

2.2 Типы переменных5

2.3 Операторы цикла5

2.4 Условные операторы6

2.5 Структуры данных6

2.6 Типы операторов7

2.7 Подпрограммы8

2.8 Библиотеки8

3 Инструментальная языковая среда9

Вывод10

Список использованных источников11

Приложение А (обязательное) Листинг программного кода12

**1 ЦЕЛИ РАБОТЫ**

1 Определить подмножество языка программирования (типы констант, переменных, операторов и функций). В подмножество как минимум должны быть включены числовые и текстовые константы, 3-4 типа переменных, операторы цикла (do…while, for), условные операторы (if…else, case)

2 Определить инструментальную языковую среду т.е. язык программирования и операционную систему для разработки.

**2 ПОДМНОЖЕСТВО ЯЗЫКА ПРОГРАММИРОВАНИЯ**

В качестве подмножества языка программирования был выбран Pascal. Pascal – это структурированный язык программирования, придуманный Никлаусом Виртом в 1968 году и названный в честь Блеза Паскаля. Он создавался с учетом целей упрощения обучения программированию и повышения ясности и поддерживаемости программного кода.

Этот язык обладает рядом характеристик, делающих его удобным для начинающих. Он обеспечивает структурированный подход программирования, что делает код более читаемым и легким для понимания. Статическая типизация помогает предотвратить множество ошибок во время выполнения программы. Он поддерживает модульное программирования, позволяя разбивать программы на отдельные модули для более легкого управления кодом.

**2.1 Числовые и строковые константы**

В языке программирования Pascal константы представляют собой значения, которые не могут изменяться во время выполнения программы. Константы могут быть как числовыми, так и строковыми.

Числовые константы в Pascal представляют собой непрерывную последовательность цифр, которая может содержать десятичные точки или экспоненту. Кроме того, числовые константы могут быть представлены в виде шестнадцатеричной (hex), восьмеричной (oct) или двоичной (bin) систем счисления, указывая префикс $ для шестнадцатеричной, 0 для восьмеричной и % для двоичной систем. На рисунке 2.1 представлен пример определения числовых констант.

Строковые константы в Pascal заключаются в одинарные или двойные кавычки. Строковая константа представляет собой последовательность символов, которая может включать буквы, цифры, символы пунктуации и специальные символы. Строковые константы также могут содержать специальные символы, такие как символ новой строки (\n), символ табуляции (\t) и т.д.

Числовые и строковые константы в языке Pascal предоставляют программистам удобный способ задания постоянных значений в программе, которые не могут быть изменены во время выполнения [1].

**2.2 Типы переменных**

В языке программирования Pascal существует разнообразие типов переменных, каждый из которых предназначен для хранения определённого вида данных. Понимание и использование различных типов переменных является ключевым аспектом разработки программного обеспечения. Переменные обычно объявляются в разделе программы, непосредственно перед основным исполняемым кодом. Объявление переменных обычно включает их идентификаторы и типы данных, которым они принадлежат.

В языке Pascal существуют следующие типы переменных:

1 Integer (целое число) – хранят целочисленные значения.

2 Real (вещественное число) – используется для представления вещественных чисел, включая числа с плавающей точкой.

3 Boolean (логическое значение) – представляет логические значения True (истина) и False (ложь). Он используется для выполнения логических операций и принятия решений в программах.

4 Char (символ) – представляет одиночный символ.

5 String (строка) – представляет последовательность символов.

Различные типы данных и их особенности позволяют эффективно структурировать данные и решать различные задачи программирования [1].

**2.3 Операторы цикла**

Операторы циклов в языке программирования Pascal предназначены для выполнения повторяющихся операций определенное количество раз или до выполнения определенного условия.

Цикл for используется для выполнения набора инструкций определённое количество раз. Он имеет начальное значение, условие продолжения выполнения и шаг итерации.

Цикл while выполняет набор инструкций, пока условие остаётся истинным.

Цикл repeat-until выполняет набор инструкций, затем проверяет условие. Если условие ложно, цикл выполняется снова.

Каждый из представленных операторов имеет свои собственные особенности и подходит для различных сценариев программирования. Использование того или иного цикла определяется условиями поставленной задачи [1].

**2.4 Условные операторы**

Условные операторы в языке программирования Pascal используются для принятия решений на основе различных условий. Они позволяют программистам направлять выполнение программы в зависимости от истинности или ложности некоторого выражения. Pascal поддерживает два основных условных оператора: if-then-else

Конструкция if-then-else позволяет выполнить одну из двух альтернативных последовательностей инструкций в зависимости от условия.

Оператор case позволяет выбирать один из нескольких вариантов для выполнения, в зависимости от значения выражения [1].

**2.5 Структуры данных**

Структуры данных играют важную роль в программировании, предоставляя способы организации и хранения данных для эффективного доступа и обработки. Язык программирования Pascal предоставляет разнообразные средства для работы с данными.

Массив (Array) в Pascal объявляется с использованием ключевого слова array с указанием нижней и верхней границы индексов, а также типа элементов массива. Элементы массива доступны по их индексам. Кроме одномерных массивов, в Pascal поддерживаются многомерные массивы, у которых каждый элемент также является массивом. Также есть возможность использовать динамические массивы, которые имею переменный размер и создаются с помощью специальных функций SetLength и New.

Set (множество) является типом данных, представляющим собой набор уникальных значений определенного типа данных. Этот тип данных позволяет объединять несколько значений в единый набор и выполнять операции с ними. Множества могут содержать или не содержать определенные значения типа, и это может быть использовано для проверки наличия или отсутствия конкретных значений в множестве. Кроме того, в Pascal для множеств предоставляются такие операции, как объединение, пересечение и разность.

Запись (record) в языке программирования Pascal представляет собой составной тип данных, который позволяет объединить несколько переменных различных типов в одну структуру. Она используется для описания объектов или структур данных, которые могут содержать несколько полей с различными типами данных. Запись объявляется с использованием ключевого слова record, после которого идет перечисление полей и их типов. Доступ к полям записи осуществляется с помощью оператора точки.

File в языке программирования представляет собой тип данных, который обеспечивает возможность работы с файлами на диске. Этот тип данных используется для создания переменных, которые могут содержать данные, считанные из файла или записанные в файл. Для работы с файлами используются различные процедуру и функции, которые позволяют открывать, закрывать, считывать и записывать данные в файлы [1].

**2.6 Типы операторов**

Pascal поддерживает широкий спектр операторов:

1 Арифметические операторы: сложение (+), вычитание (-), умножение (\*), деление (/), целочисленное деление (div), остаток от деления (mod) и возведение в степень (\*\* или ^).

2 Побитовые операторы: И (and), ИЛИ (or), НЕ (not), исключающее ИЛИ (xor), сдвиг влево (shl), сдвиг вправо (shr).

3 Операторы сравнения: равенство (=), неравенство (<>), больше (>), меньше (<), больше или равно (>=), меньше или равно (<=).

4 Операторы присваивания: присваивание (:=), инкремент (+=), декремент (-=).

5 Операторы управления выполнением программы: условный оператор (if-then-else), циклы (while, repeat-until, for), операторы перехода (break, continue, exit).

6 Операторы работы с памятью: выделение памяти (new), освобождение памяти (dispose), получение адреса переменной (@).

7 Операторы работы с типами данных: преобразование (as), проверка (is).

8 Операторы работы с множествами: объединение (+), пересечение (\*), разность (-), проверка на вхождение (in).

9 Операторы работы с файлами: открытие файла (reset, rewrite, append), закрытие файла (close), чтение из файла (read), запись в файл (write).

10 Операторы для работы со строками: конкатенация (+).

11 Индексация массивов и записей: доступ к элементом массива или полям записи ([]) [1].

**2.7 Подпрограммы**

В Pascal существует два типа подпрограмм – функции и процедуру. Они представляют собой последовательность операторов с именем, списком параметров и могут быть вызваны из различных частей программы.

Процедуры и функции называются подпрограммами для обеспечения единообразия. Процедуры выполняют последовательность операторов, но не возвращают значение. Функции, напротив, возвращают значение после своего выполнения и могут быть использованы в выражениях.

Каждая используемая подпрограмма должна быть предварительно описана в разделе описаний. Операторы подпрограммы окружаются операторными блоками begin/end и составляют тело подпрограммы. Список формальных параметров, если он есть, разделен символом «;». Каждая секция списка параметров может предваряться ключевыми словами var или const, указывающими на то, как параметры будут передаваться [1].

**2.8 Библиотеки**

Использование модулей в Pascal является важным методом для организации кода и создания модульных программ. Модуль представляет собой отдельный файл, содержащий определения типов данных, переменных, констант, процедур и функций. Он позволяет разделить программу на логические компоненты.

Модули в Pascal создаются путем создания файлов с расширением .pas, в которых размещаются определения типов данных, переменных, процедур и функций. Обычно каждый модуль имеет две секции: interface и implementation. В секции interface объявляются элементы, которые будут доступны извне модуля, а в секции implementation содержатся реализации этих элементов. Модуль можно подключить с помощью директивы uses, указав имя модуля без расширения. При компиляции программы компилятор найдет и подключит соответствующий модуль [1].

**3 ИНСТРУМЕНТАЛЬНАЯ ЯЗЫКОВАЯ СРЕДА**

В качестве языка программирования для реализации программы был выбран язык программирования C# версии 12. C# – это современный объектно-ориентированный и типобезопасный язык программирования. C# позволяет создавать разные типы безопасных и надежных приложений, выполняющихся в .NET. C# относится к широко известному семейству языков C, и покажется знакомым любому, кто работал с C, C++, Java или JavaScript.

Программы C# выполняются в .NET, виртуальной системе выполнения, вызывающей общеязыковую среду (CLR) и набор библиотек классов. Среда CLR – это реализация общеязыковой инфраструктуры языка (CLI), являющейся международным стандартом от корпорации Майкрософт.

Исходный код, написанный на C# компилируется в промежуточный язык (IL), который соответствует спецификациям CLI. Код на языке IL сохраняется в сборке, обычно с расширением .dll. Сборка содержит манифест с информацией о типах, версии, языке и региональных параметрах для этой сборке.

При выполнении программы C# сборка загружается в среду CLR. Среда CLR выполняет JIT-компиляцию из кода на языке IL в инструкции машинного языка. Среда CLR также выполняет другие операции, например, автоматическую сборку мусора, обработку исключений и управление ресурсами. Код, выполняемый средой CLR, иногда называют «управляемым кодом». «Неуправляемый код» компилируется на машинный язык, предназначенный для конкретной платформы [2].

Для написания кода выбрана интегрированная среда разработки JetBrains Rider.

В качестве операционной системы выбрана ОС Windows 10.

**ВЫВОД**

В ходе выполнения лабораторной работы было выбрано подмножество языка программирования, которое включает в себя основные элементы для создания структурированных программ, такие как числовые и текстовые константы, различные типы переменных, операторы цикла, условные операторы, различные структуры данных, функции, а также различные операторы.

Для разработки программы на данном подмножестве языка программирования была выбрана инструментальная языковая среда на основе языка C#, так как он обладает высокой производительностью и широким спектром возможностей.

**СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ**

[1] Pascal Reference guide [Электронный ресурс] – Режим доступа: <https://downloads.freepascal.org/fpc/docs-pdf> – Дата доступа: 05.02.2024

[2] Документация C# [Электронный ресурс] – Режим доступа: <https://learn.microsoft.com/ru-ru/dotnet/csharp/> – Дата доступа: 05.02.2024.

**ПРИЛОЖЕНИЕ А**

**(обязательное)**

**Листинг программного кода**

Листинг 1 – Пример простой первой программы.

program Example1;

var

number: integer;

begin

writeln('Enter number:');

readln(number);

if number > 0 then

begin

writeln('Number is positive');

end

else if number < 0 then

begin

writeln('Number is negative');

end

else

begin

writeln(‘Number is zero');

end;

for number := 1 to 5 do

begin

writeln('This is string number ', number);

end;

end.

Листинг 2 – Пример программы с использованием функций и процедур.

program Example2;

function Factorial(n: integer): integer;

begin

if n <= 1 then

Factorial := 1

else

Factorial := n \* Factorial(n - 1);

end;

procedure PrintFactorial(n: integer);

begin

writeln('Factorial of ', n, ' is equal ', Factorial(n));

end;

var

number: integer;

begin

writeln('Enter number:');

readln(number);

PrintFactorial(number);

end.

Листинг 3 – Пример программы с использованием массивов и строк.

program Example3;

const

MAX\_SIZE = 5;

type

TStringArray = array[1..MAX\_SIZE] of string;

var

strings: TStringArray;

i: integer;

begin

writeln('Enter ', MAX\_SIZE, ' strings:');

for i := 1 to MAX\_SIZE do

begin

readln(strings[i]);

end;

writeln('You entered:');

for i := 1 to MAX\_SIZE do

begin

writeln(strings[i]);

end;

end.