Министерство образования Республики Беларусь
Учреждение образования «Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра информатики

Дисциплина: Методы трансляции

ОТЧЁТ
к лабораторной работе №1
на тему

ОПРЕДЕЛЕНИЕ МОДЕЛИ ЯЗЫКА. ВЫБОР ИНСТРУМЕНТОВ ЯЗЫКОВОЙ СРЕДЫ.

Выполнил: студент гр.253504 Лавренова А.С.
Проверил: ассистент кафедры информатики Гриценко Н.Ю.

Минск 2025

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Языки программирования играют важную роль в разработке программного обеспечения, предоставляя программистам средства для создания алгоритмов и работы с данными. Один из таких языков – Focal, который является ранним процедурным языком, разработанным для инженерных и научных расчетов. Несмотря на его возраст, он остаётся интересным с точки зрения изучения основ программирования и принципов построения языков.

В данной лабораторной работе проводится исследование подмножества языка Focal, включающего основные конструкции:

– типы данных;

– операторы циклов;

– условные выражения;

– функции и структуры данных.

Также рассматривается инструментальная языковая среда, в которой осуществляется разработка, включая операционную систему и аппаратную платформу.

Целью работы является формирование четкого понимания основных возможностей языка Focal, его синтаксических конструкций и способов взаимодействия с инструментальной средой. Для достижения этой цели в отчёте приводится описание подмножества языка, три примера программ, демонстрирующих его использование.

Кроме того, в рамках последующих лабораторноых работы будет разработан транслятор для подмножества языка Focal. Этот транслятор будет написан на языке программирования Kotlin, который обладает современным синтаксисом, выразительностью и мощным средствам работы со строками, коллекциями и функциональным программированием. Kotlin обладает высокой читаемостью кода, что упростит реализацию транслятора, а также предоставляет развитую экосистему инструментов, включая возможности для работы с парсингом, генерацией кода и тестированием.

Использование Kotlin в качестве основного языка разработки транслятора позволит продемонстрировать принципы построения компиляторов и интерпретаторов, включая разбор исходного кода, анализ синтаксических конструкций и генерацию выходного кода. Кроме того, благодаря возможностям Kotlin, таким как DSL (Domain-Specific Language) подходы, функциональные конструкции и мощная стандартная библиотека, процесс разработки будет более гибким и эффективным.

Понимание принципов работы языка Focal позволит лучше осознать особенности процедурных языков программирования, научиться разрабатывать алгоритмы и применять структурные элементы кода.

1 ЯЗЫК ПРОГРАММИРОВАНИЯ

FOCAL следует модели JOSS для взаимодействия с пользователем с помощью командной строки. Это позволяет вводить команды, которые будут выполнены немедленно (директивный режим). Или добавлять к ним номер строки (недирективный режим). В этом случае они добавляются в программу, если номер уникальный, или заменяют уже существующую строку с таким же номером. [1]

1.1 Операторы программы

Каждая строка в программе FOCAL должна начинаться с номера строки. Как и в JOSS, номера строк – это числа с фиксированной запятой, состоящие из двух двузначных чисел, разделенных точкой. При распечатке командой WRITE добавляются ведущие нули, так что все номера строк будут содержать пять символов, включая точку, например строка 1.10 будет напечатана как 01.10. Операторы, обращающиеся к этим строкам, не нуждаются в ведущих нулях, например, GOTO 1.10.

Число слева от точки называется «номером группы». Группы обеспечивают определённый уровень организации кода, которого не хватает в таких языках, как Fortran или BASIC.

Каждая строка должна начинаться с ключевого слова команды, следующего за номером строки. Не существует «команды по умолчанию», как в BASIC с его необязательным LET. Несколько операторов могут быть помещены в одну строку, разделённые точкой с запятой. Обычно это поведение ничем не отличается от того, если бы операторы были в отдельных строках, за исключением ситуации с циклом FOR.

1.2 Общие команды

Команда ASK (сокращённо A) берет список строк и переменных, выводит строки на экран и сохраняет введенные пользователем данные в переменных.

Команда COMMENT (сокращённо C) создаёт комментарий. Эквивалентна REM в BASIC.

Команда DO (сокращённо D) создаёт ветвь для выполнения подпрограммы. Это эквивалент BASIC команды GOSUB. На подпрограмму ссылается либо номер группы, либо номер строки. Если указан номер строки, эта строка выполняется и затем происходит возврат к оператору после DO. Если номер строки не указан, выполнение начинается с первой строки группы и продолжается, пока не будет достигнут конец группы или не встретится RETURN. RETURN требуется только для раннего возврата, в конце группы он не нужен.

Команда FOR (сокращённо F) реализует цикл for. Когда заданы три аргумента, первый – это начальное значение переменной цикла, второй – приращение, а третье – это конечное значение цикла. Если указаны два значения, первое – это начальное значение, а второе – конечное значение, а приращение устанавливается равным 1.

Команда GOTO (сокращённо G) переводит выполнение программы на указанный номер строки. Он идентичен одноименной команде в BASIC. В отличие от GO, используемой из командной строки, которая запускает программы.

Команда IF (сокращённо I) обеспечивает ветвление по условию, основанному на знаке выражения. После этого выражения команда IF может содержать список от одного до трех номеров строк. Если результат выражения меньше нуля, выполнение переходит к первому номеру; если равен нулю, на второй номер; если больше нуля, номер третьей строки. В языке отсутствуют относительные операторы, такие как «больше чем», «равно» или «меньше». Для ветвления по условию IF X>5, нужно проверить результат выражения X-5.

Команда QUIT (сокращённо Q) останавливает программу и возвращает управление среде редактирования. Эквивалентно END или STOP в BASIC.

Команда RETURN (сокращённо R) выполняет возврат из подпрограммы к месту её вызова. Использование RETURN является необязательным в последней строке, подпрограмма в любом случае возвращается из последней строки в группе.

Команда SET (сокращённо S) присваивает результат выражения указанной переменной.

Команда TYPE (сокращённо T) обеспечивает вывод одного или нескольких элементов, разделенных запятыми.

TYPE также может иметь необязательный спецификатор формата, указанный в виде %x.yz, где x – это число цифр до точки, а yz – это число цифр справа от точки. Формат по умолчанию %8.4, что означает максимум восемь цифр до и четыре справа от точки.

1.3 Переменные

Имена переменных могут начинаться с любой буквы, кроме F (F зарезервирована для функций) и могут содержать любую последовательность букв и цифр. Тем не менее, только первые два символа имеют значение. Например, следующий пример кода из FOCAL в A New Conversational Language ссылается на ту же переменную, что и DESTINATION, а затем DES.

Любая переменная может рассматриваться как массив, что позволяет использовать индексы от -2048 до 2047.

Имена переменных в FOCAL регистронезависимы. Это значит, что destination, Destination, и DESTINATION — это одна и та же переменная. [2]

1.4 Математические функции

FOCAL содержит только пять математических операций:
– сложение (+);
– вычитание (-);
– умножение (*);
– деление (/);
– экспонента (^) - экспонента преобразуется в целое число.

FOCAL-69 был необычен тем, что математические выражения могли использовать (), [] и <> взаимозаменяемо в парах для установления приоритета.

Язык не поддерживал никаких относительных операторов (например, больше, GT., > и т.д.).

1.5 Встроенные функции

В языке программирования FOCAL предусмотрен ряд встроенных функций, предназначенных для выполнения математических операций. Эти функции позволяют программисту производить вычисления без необходимости реализовывать их вручную, что значительно упрощает процесс написания кода и повышает его читаемость:

– FABS() – абсолютное значение;
– FATN() – арктангенс;
– FCOS() – косинус от аргумента в радианах;
– FEXP() – возвращает число e, возведенное в указанную степень;
– FITR() – целая часть аргумента;
– FLOG() – натуральный логарифм;
– FRAN() – случайное число;
– FSGN() – знак аргумента; FSGN(0)=1 в FOCAL-69, но FSGN(0)=0 в FOCAL-71, как и в более поздних версиях;
– FSIN() – синус от аргумента в радианах;
– FSQT() – квадратный корень.

Язык программирования FOCAL представляет собой систему, ориентированную на интерактивное взаимодействие с пользователем через командную строку, следуя модели JOSS. Его структура, включающая строгую нумерацию строк и организацию кода по группам, обеспечивает удобочитаемость и порядок в программах. Основные команды позволяют эффективно управлять вводом/выводом данных, реализацией циклов и ветвлений, а также вызовом подпрограмм.

2 ИНСТРУМЕНТАЛЬЯ ЯЗЫКОВАЯ СРЕДА

2.1 Язык программирования Kotlin

Kotlin — это современный, многопарадигменный язык программирования, разработанный компанией JetBrains. Он ориентирован на разработку безопасного, лаконичного и выразительного кода, поддерживает функциональное и объектно-ориентированное программирование. Kotlin совместим с Java и работает на платформе JVM, что делает его удобным для разработки Android-приложений, серверных решений и кроссплатформенных приложений. Язык имеет статическую типизацию, упрощенный синтаксис, поддержку асинхронного программирования и мощную систему работы с null-значениями, что снижает вероятность ошибок. [3]

2.2 Операционная система Windows

Для выполнения лабораторной работы используется операционная система Windows 11 Home. Она обеспечивает стабильную работу инструментов разработки, поддержку современных интерфейсов и совместимость с компиляторами и средами разработки для Kotlin. Windows 11 Home предлагает улучшенные функции для тестирования и отладки программ, а также удобную интеграцию с инструментами командной строки, что делает процесс разработки эффективным и удобным.

2.3 Компьютер: PC

Для разработки используется персональный компьютер, соответствующий требованиям Windows и инструментов разработки Kotlin. Это может быть как настольный ПК, так и ноутбук с достаточными вычислительными ресурсами (мощный процессор, большой объем оперативной памяти, современная графика). Хорошая аппаратная конфигурация обеспечивает быструю компиляцию, плавную работу среды разработки и комфорт при программировании.

2.4 Используемые программы

Kotlin — это следующий этап развития Java, с которой он полностью совместим. Это делает его отличным инструментом для мобильных и энтерпрайз-приложений. Visual Studio Code — идеальный выбор для создания проектов Kotlin. Он предлагает такие функции интегрированной среды разработки, как автозаполнение, выделение текста/синтаксиса, форматирование текста и выделение кода.

ЗАКЛЮЧЕНИЕ

В ходе выполнения данной лабораторной работы был рассмотрен язык программирования Kotlin, его основные конструкции, синтаксис и встроенные возможности. Изучение данного языка позволило выявить его ключевые особенности, такие как лаконичность синтаксиса, статическая типизация, поддержка функционального программирования и безопасность при работе с null-значениями. Эти аспекты делают Kotlin удобным и эффективным инструментом для разработки современных приложений.

Была также рассмотрена инструментальная языковая среда, включающая операционную систему Windows 11 и программное обеспечение, используемое для разработки. В ходе работы рассматривалась среда разработки Visual Studio Code. Благодаря встроенным инструментам и плагинам, поддерживающим Kotlin, процесс разработки был оптимизирован, что позволило эффективно изучить язык и его особенности.

Кроме того, в рамках лабораторной работы был изучен язык программирования Focal, который является ранним процедурным языком, использовавшимся для научных и инженерных расчетов. Анализ его структуры показал особенности работы с командами, нумерацией строк, математическими и встроенными функциями. Несмотря на свою устаревшую архитектуру, Focal представляет собой интересный пример языков программирования той эпохи, обеспечивая полезный исторический контекст для изучения развития процедурных языков.

Практическая часть лабораторной работы включала разработку программных примеров, демонстрирующих основные конструкции языка. Были исследованы такие элементы, как операторы, циклы, условные выражения, математические и встроенные функции. Реализация данных примеров позволила закрепить теоретические знания и получить практический опыт программирования на Focal.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

[1] Фокал для микро- и мини-компьютеров [Электронный ресурс]. – Режим доступа: https://gid.pdp-11.ru/docs/focal.pdf.

[2] Focal-8 [Электронный ресурс]. – Режим доступа: http://bitsavers.informatik.uni-stuttgart.de/pdf/dec/pdp8/focal/DEC-08-AJAB-D_FOCAL_Programming_Manual_Jan70.pdf.

[3] Kotlin в действии. – [Электронный ресурс]. – Режим доступа: https://notasi.ru/javangelion/books/kotlin/Kotlin_in_Action_2018_RU.pdf?ysclid=m6s28v58lg250309732.

ПРИЛОЖЕНИЕ А
(обязательное)
Исходный код

```
010.10   ! Программа "Калькулятор арифметических операций"
010.20   TYPE "Введите первое число: "
010.30   ACCEPT A1
010.40   TYPE "Введите второе число: "
010.50   ACCEPT B2
010.60   LET S = A1 + B2          ! Сложение
010.70   LET D = A1 - B2          ! Вычитание
010.80   LET M = A1 * B2          ! Умножение
010.90   LET Q = A1 / B2          ! Деление
010.95   ! Экспонента: A1 возводится в степень B2, результат
преобразуется к целому
010.96   LET E = A1 ^ B2
010.97   TYPE "Сумма: ", S
010.98   TYPE " Разность: ", D
010.99   TYPE " Произведение: ", M
011.00   TYPE " Частное: ", Q
011.01   TYPE " Экспонента: ", E
011.02   END

020.10   ! Программа "Тригонометрия и логарифмы"
020.20   TYPE "Введите угол (в радианах): "
020.30   ACCEPT AN
020.40   ! Вычисление синуса, косинуса и арктангенса
020.50   LET S = FSIN(AN)
020.60   LET C = FCOS(AN)
020.70   LET AT = FATN(AN)
020.80   ! Вычисление натурального логарифма и квадратного корня
(используем FLOG и FSQT)
020.90   TYPE "Введите число для логарифма и квадратного корня: "
020.95   ACCEPT X0
021.00   LET L = FLOG(X0)
021.10   LET Q = FSQT(X0)
021.20   ! Вычисление абсолютного значения и целой части
021.30   LET ABSVAL = FABS(-X0)
021.40   LET INTGR = FITR(X0)
021.50   TYPE "SIN: ", S, " COS: ", C, " ATAN: ", AT
021.60   TYPE "Логарифм: ", L, " Квадратный корень: ", Q
021.70   TYPE "Абсолютное значение: ", ABSVAL, " Целая часть: ", INTGR
021.80   END

030.10   ! Программа "Массивы и циклы"
030.20   TYPE "Генерация 10 случайных чисел и вычисление их
квадратов:"
030.30   FOR I = -2, 7        ! Использование переменной цикла (первые
два символа "I " важны)
030.40       ! Генерация случайного числа и сохранение в массив R;
допускается использование индексов от -2048 до 2047
030.50       LET R(I) = FRAN()
030.60       ! Вычисление квадрата случайного числа: здесь применяется
группировка скобками (можно использовать <>, [] или ())
030.70       LET Q = (R(I) * R(I))
```

```
030.80       TYPE "Число: ", R(I), " Квадрат: ", Q
030.90   NEXT I
031.00   ! Демонстрация использования FEXP и FSGN
031.10   TYPE "Введите число для FEXP и FSGN: "
031.20   ACCEPT NUM
031.30   LET EXPVAL = FEXP(NUM)   ! Вычисление e^(NUM) (результат
целочисленный)
031.40   LET SG = FSGN(NUM)       ! Определение знака числа (в FOCAL-
69 FSGN(0)=1)
031.50   TYPE "FEXP: ", EXPVAL, " FSGN: ", SG
031.60   END
```