МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования «БЕЛОРУССКИЙ

ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет информационных технологий

Кафедра программной инженерии

Специальность 6-05-0612-01 Программная инженерия

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**К КУРСОВОМУ ПРОЕКТУ НА ТЕМУ:**

«Разработка компилятора FIA-2024»

Выполнил студент Филипюк И.А.

(Ф.И.О.)

Руководитель проекта ст.преп. Наркевич А.С.

Заведующий кафедрой к.т.н., доц. Смелов В.В.

Консультанты ст.преп. Наркевич А.С.

Нормоконтролер ст.преп. Наркевич А.С.

Курсовой проект защищен с оценкой

Минск 2024

Оглавление

1. **Спецификация языка программирования**

В данном разделе описывается спецификация языка программирования FIA-2024.

**1.1.Характеристика языка программирования**

Язык программирования FIA-2024 – компилируемый высокоуровневый, универсальный язык программирования со статической типизацией. Язык строго типизирован и из парадигм программирования данный язык поддерживает только функциональное программирование.

**1.2.Определение алфавита языка программирования**

Совокупность символов, разрешенных для использования языком программирования формирует его алфавит. Язык FIA-2024 использует стандартную кодировку символов Windows-1251. В нём разрешены строчные и прописные латинские буквы , цифры от 0 до 9 , символы пробела и новой строки, сепараторы, круглые, фигурные и квадратные скобки и т.д.

В таблице 1.1 описан алфавит языка с помощью формы Бэкуса-Наура.

<Строчные латинские буквы>::=a|b|c|d|e|f|g|h|i|j|k|l|m|n|o|p|q|r|s|t|u|v|w|x|y|z

<Прописные латинские буквы>::= A|B|C|D|E|F|G|H|I|J|K|L|M|N|O|P|Q|R|S|T|U|V|W|X|Y|Z

<Цифры>::=1|2|3|4|5|6|7|8|9

<Символы-сепараторы>::=,|{|}|(|)|;|пробел|’ ’|” “|+|-|\*|/|=|\_|:

<Символы операции сдвига>::=<<|>>

Таблица 1.1

**1.3 Применяемые сепараторы!**

Символы-сепараторы, описанные в таблице 1.1 служат для разделения операций, слов и конструкций языка. Ниже описаны назначение и применение сепараторов, описанных в таблице 1.1:

* {…} – Фигурные скобки. Используются для заключения в них блоков программного кода.
* (…) – Круглые скобки. Используются для заключения в них параметров функций, а также допускаются в математических операциях для отделения операндов.
* Пробел – Используется для разделения лексем. Допускается везде, кроме идентификаторов и ключевых слов.
* , – Запятая. Используется для разделения параметров функций.
* ; – Точка с запятой. Используется для разделения конструкций языка
* +,-,\*,/ - Знаки математических операций. Допускаются только в математических операциях, между идентификаторами или числовыми константами.
* = - Одинарное равно. Используется в математических операциях в качестве символа операции присвоения.
* == - Двойное равно. Используется как символ равенства.
* ! – Символ отрицания. Допускается в логических или тернарных операторах.
* \_ - Нижнее подчёркивание. Используется для логического разделения слов в идентификаторах
* : - Двоеточие. Используется для разделения условия и действия, выполняющегося при соответствии условию в тернарных операторах и операторах цикла.

**1.4 Применяемые кодировки**

Язык FIA-2024 использует кодировку Windows-1251 для написания исходного кода.

**1.5 Типы данных**

Язык FIA-2024 поддерживает как примитивные, так и пользовательские типы данных. Они описаны ниже в таблице 1.2.

|  |  |
| --- | --- |
| Тип данных | Описание типа данных |
| Беззнаковый  целочисленный тип данных unt | Примитивный тип данных. Используется для работы с целочисленными беззнаковыми значениями. В памяти занимает 4 байта.  При попытке инициализации значением больше максимального, инициализируется максимальным. При попытке инициализации значением меньше минимального, инициализируется нулём.  Максимальное значение: 2147483647. Минимальное значение: 0.  Инициализация по умолчанию: значение 0. |
| Символьный тип данных sym | Примитивный тип данных. Используется для работы с символом, который в памяти занимает 1 байт. Инициализация по умолчанию: символ конца строки “\0”. |
| Строковый тип данных str | Примитивный тип данных. Используется для работы с символьными строками. В памяти (!) байт. Инициализация по умолчанию: символ конца строки “\0”. |
| Массив arr | Пользовательский тип данных. Используется для работы с группой значений одного из допустимых в языке типов данных. В памяти занимает (!) байт. Инициализация по умолчанию: Каждый из элементов массива инициализируется значением по умолчанию согласно типу данных значений, содержащихся в массиве. |

Таблица 1.2

**1.6 Преобразование типов данных**

Символьный тип данных sym может быть преобразован в беззнаковый целочисленный тип данных unt при условии, что преобразуемое значение является символьным представлением числа.

**1.7 Идентификаторы**

Максимальная длина идентификатора – 10 символов. Максимальная длина идентификатора функции – 15 символов. Из сепараторов при написании идентификаторов допускается только символ ‘\_’(нижнее подчёркивание). При попытке инициализации идентификатора значением с длиной больше допустимой она будет усекаться. Идентификаторы не могут повторяться.

**1.8 Литералы**

Литералы языка программирования – последовательность символов, используемая для именования неизменяемых значений в коде.

В языке FIA-2024 допускаются только целочисленные и логические литералы.

Более подробно они описаны в таблице 1.3:

|  |  |
| --- | --- |
| Литерал | Описание |
| Целочисленный | Могут состоять только из цифр без дробной части.  Могут состоять только из цифр [0..9] без дробной части при выводе.  При выходе за пределы допустимости значение будет преобразовано в максимальное для данного типа.  Восьмеричная система счисления: 8x[0-7]\* |
| Логический | Состоит из ключевого слова T или F. Также может записываться как 1 или 0 соответственно. |

Таблица 1.3 – Литералы языка FIA-2024

**1.9 Объявление данных**

Для объявления данных используется ключевые слова create и global (совместно с create). Данные могут объявляться в любом месте программы. Если данные объявлены внутри функции, то и видны они будут только внутри этой функции. Если же данные объявлены за пределами какой-либо функции, то переменная будет видна в любой точке программы(обязательно использование ключевого слова global. В противном случае будет выдана соответствующая ошибка).

Правила объявления:

* Каждая переменная должна быть обязательно объявлена до её использования.
* Объявление переменной обязательно начинается с ключевого слова create, после чего идёт тип данных (один из допустимых для языка FIA-2024) и идентификатор.
* При объявлении глобальной переменной после ключевого слова create о1бязательно идёт ключевое слово global.
* Не глобальные переменные не могут быть объявлены вне функции.
* Глобальные переменные не могут быть объявлены внутри функции.
* Объявление функции обязательно начинается с ключевого слова function, после которого идет тип данных возвращаемого значения, а в круглых скобках указываются параметры функции( или не указывается ничего, если функция не принимает параметров).

**1.10 Инициализация данных**

Переменные можно инициализировать значениями только во время объявления и после него. При инициализации переменной до момента её объявления будет выдана соответствующая ошибка.

Для инициализации переменной значением используется символ-сепаратор = (одинарное равно). Синтаксис: create <тип данных> = <значение>.

Переменные можно инициализировать только тем типом данных, которым они объявлены. Исключением является присвоение символьному типу данных целочисленного значения и наоборот(при условии, что целочисленному типу данных присваивается символьное значение, соответствующее числу).

**1.11 Инструкции языка**

Все инструкции, поддерживаемые языком FIA-2024 описаны в таблице1.4:

|  |  |  |
| --- | --- | --- |
| Операция | Описание | Пример |
| Объявление переменной | create <тип данных> <идентификатор>; | create unt number; |
| Инициализация переменной | <идентификатор> = <значение>  Create <тип данных> <идентификатор> = <значение>; | number = 10;  create unt number = 10; |
| Объявление функции | create <тип данных> function <идентификатор> (<параметры через ,>){<тело функции>}; | create unt function output(unt a, unt b){}; |
| Выход из функции | return <значение или идентификатор>; | return a; |
| Оператор цикла | until <условие выхода их цикла>{<тело цикла>}; | until x<y{x++}; |
| Вывод данных | write <данные>;  writeline <данные>; | write symbol;  writeline symbol; |
| Подключение библиотек | using <название>; | using standardlib; |

Таблица 1.4 – Инструкции языка FIA-2024

**1.12 Операции языка**

Язык FIA-2024 поддерживает два типа операций: математические и сдвиговые. Они описаны в таблице 1.5:

|  |  |  |  |
| --- | --- | --- | --- |
| Операция | Описание | Количество операндов | Пример |
| **+** | Арифметическая операция сложения. Коммутативная, ассоциативная, не дистрибутивная | 2 | Num1+Num2 |
| **-** | Арифметическая операция вычитания. Коммутативная, ассоциативная, не дистрибутивная | 2 | Num1-Num2 |
| **/** | Арифметическая операция деления. Не коммутативная, не ассоциативная, дистрибутивная | 2 | Num1/Num2 |
| **\*** | Арифметическая операция умножения. Не коммутативная, не ассоциативная, дистрибутивная | 2 | Num1\*Num2 |
| **<<** | Операция сдвига влево. Левоассоциативная, не коммутативная, не дистрибутивная | 2 | Num1<<2 |
| **>>** | Операция сдвига вправо. Правоассоциативная, не коммутативная, не ассоциативная, не дистрибутивная | 2 | Num1>>2 |
| () | Приоритетность операций. Ассоциативная, коммутативная, дистрибутивная. | 1 | (Num1+Num2) |

Таблица 1.5 – Операции языка FIA-2024

**1.13 Выражения и их вычисление**

Под выражением понимается некоторая комбинация литералов, переменных, констант и операций (как арифметических, так и сдвиговых), дающая в результате некоторое значение.

Правила написания выражений в языке FIA-2024:

* Выражения рассматриваются только справа налево.
* В общем случае приоритет имеют операции умножения и деления. Для изменения приоритета операций используются оператор () (круглые скобки).
* Любое выражение должно присваиваться переменной для хранения вычисляемого в выражении значения.
* Запись любого выражения должна заканчиваться точкой с запятой (;).

**1.14 Конструкции языка**

Язык программирования FIA-2024 использует несколько стандартных конструкций. Все они описаны в таблице 1.6:

|  |  |  |
| --- | --- | --- |
| Конструкция | Описание | Пример |
| Точка входа в программу | Main(){<тело функции>}; | Main(){  }; |
| Блок кода | {…}; | {  Num3 = Num1+Num2;  }; |
| Функция | create <тип данных> function <идентификатор> (<параметры>){<тело функции>}; | create unt function announce(sym announcement){  write announcement;  }; |
| Цикл | until <условие выхода из цикла>{<тело цикла>}; | until i<10{  i++;  }; |

Таблица 1.6 – Конструкции языка FIA-2024

Исходный код, написанный на языке FIA-2024, обязательно представляет собой главную функцию Main (точка входа в программу). Все остальные функции должны быть обязательно объявлены до объявления главной функции. Главная функция обязательно должна возвращать значение.

**1.15 Область видимости идентификаторов**

Языком FIA-2024 рассматриваются два типа видимости для идентификаторов: глобальный и локальный. Все они и их характеристики описаны ниже в таблице 1.7.

|  |  |
| --- | --- |
| Область видимости | Описание |
| Глобальная | Переменные, объявленные с использованием ключевого слова global и обязательно вне функций и любых блоков кода. Переменные, объявленные таким образом видны в любой точке программы. |
| На уровне функции | На уровне функции видны переменные с глобальной областью видимости, переменные, передающиеся в функцию в качестве параметров и переменные, объявленные непосредственно в самой функции. Переменные с такой областью видимости видны только в самой функции. |
| На уровне блока кода | Переменные, объявленные в конструкциях тернарных операторов и операторов цикла. Такие переменные видны только внутри оператора, где они объявлены. |

Таблица 1.7 – Область видимости идентификаторов языка FIA-2024

**1.16 Семантические проверки**

Язык FIA-2024 осуществляет семантические проверки во всех местах, где могут возникнуть ошибки компиляции. А именно, проверяется следующее:

- единственное объявление главной функции;

- объявление главной функции после всех остальных функций и глобальных переменных;

- преобразование типов данных;

- объявление и инициализация идентификаторов;

- правильность постановки символов, закрывающих блоки кода или функции (фигурные скобки и точки с запятой);

- правильность написания выражений;

- правильность передачи данных в функцию и возврата данных функцией;

**1.17 Распределение оперативной памяти на этапе выполнения**

Используется стековая модель памяти для размещения переменных.

**1.18 Стандартная библиотека и её состав**

Стандартная библиотека языка FIA-2024 включает в себя следующие функции:

|  |  |
| --- | --- |
| Функция | Описание |
| get\_date() | Функция для определения текущей даты. Возвращает массив из трёх элементов, где первый элемент – текущий день, второй элемент – текущий месяц, третий элемент – текущий год. Функция не принимает параметров. |
| get\_time() | Функция для определения текущего времени с точностью до секунд. Возвращает массив из трёх элементов, где первый элемент – текущий час, второй – минута, третий – секунда. Возвращаемые данные соответствуют времени на момент вызова функции. Функция не принимает параметров. |

Таблица 1.8 – Функции стандартной библиотеки языка FIA-2024

**1.19 Ввод и вывод данных**

Для ввода и вывода данных в языке FIA-2024 используются стандартные операторы, описанные в таблице 1.9.

|  |  |  |
| --- | --- | --- |
| Оператор | Описание | Пример |
| Get | Оператор для ввода одного символа (если вводится символ) или любой цифры и числа.  Переменная, в которую будет записано введённое значение, должна быть обязательно объявлена до ввода значения с помощью оператора Get.  Синтаксис: Get <идентификатор>;  В случае использования оператора с большим количеством символов будет выдана соответствующая ошибка. | create sym symbol;  Get symbol;  create unt number;  Get number; |
| Getline | Оператор для ввода строки. В случае ввода цифры или числа вместо символьной строки введённые данные будут преобразованы в строку.  Синтаксис: Getline <идентификатор>; | create str line;  Getline line; |
| Write | Оператор для вывода данных . Выводит данные без перехода на следующую строку.  Синтаксис: Write <идентификатор или значение>; | Create unt num = 15;  Write num;  Write 15; |
| Writeline | Оператора для вывода данных с переходом на следующую строку.  Синтаксис: Writeline <идентификатор или значение>; | Create str line = “I love c++”;  Writeline line;  Writeline “I love Java”; |

Таблица 1.9 – Операторы ввода и вывода данных языка FIA-2024

**1.20 Точка входа**

Точкой входа является главная функция, именуемая Main. Точка входа обязательно должна быть в исходном коде, а также обязательно должна быть объявлена после всех функций и глобальных переменных.

**1.21 Препроцессор**

Язык FIA-2024 использует препроцессор только для анализа директивы using, используемой для подключения библиотек.

**1.22 Соглашения о вызовах**

Соглашение о вызовах в языке FIA-2024 называется stcagr (standard call agreement), по которому все параметры функции передаются через стек и память очищается вызываемым кодом.

**1.23 Объектный код**

Исходный код наязыке FIA-2024 компилируется в собственный байт-код, который, в свою очередь, переводится в Ассемблер.

**1.24 Классификация сообщений транслятора**

Все ошибки, выводимые транслятором разделены по смысловым диапазонам и описаны в таблице 1.10.

|  |  |
| --- | --- |
| Диапазон ошибок | Описание |
| 0-99 | Системные ошибки |
| 100-199 | Ошибки параметров |
| 200-249 | Ошибки открытия, чтения и закрытия файлов |
| 250-399 | Лексические и семантические ошибки |
| 400-499 | Структурные ошибки |
| 500 - 600 | Семантические ошибки |

Таблица 1.10 – Классификация сообщений транслятора

**1.25 Контрольный пример**