1. Общие сведения об ЭВМ и преобразовании информации в них.

Понятия вычислительной математики, информации, и алгоритмической обработки тесно связаны между собой. Это следует из определения в энциклопедии кибернетики - ”машина - физическая система (устройство или комплекс устройств), предназначенная для автоматизации процесса алгоритмической обработки информации и вычислений”.

Информация определяет многие процессы, происходящие в вычисли­тельной машине. (ЭВМ)-машина получает информацию, запоминает ее, обрабатывает по заданным алгоритмам и направляет пользователю или в другие системы обработки. Общение с ЭВМ предполагает наличие опре­деленных аппаратных средств и соответствующего программного обеспе­чения для ввода, обработки и вывода информации, ориентированной на ведение диалога в системе “человек - ЭВМ”. Большинство диалоговых сис­тем содержат функциональные программы, обеспечивающие стандартные применения ЭВМ. Это текстовые редакторы, отладчики, сервисные сред­ства просмотра и корректировки содержимого памяти. Важное значение имеют интенсивно развивающиеся средства машинной графики.

1.1. Современные компьютеры (ПК )

Основными компонентами компьютера являются процессор, память, устройства ввода и вывода ( ВВ ). С помощью устройства ввода про­грамма и исходные данные попадают в память. Программа содержит по­следовательность инструкций, которую выполняет процессор. Результаты выполнения программы поступают в устройство вывода. Обычно в каче­стве устройства ввода используют клавиатуру, “мышь”; в качестве устрой­ства вывода - дисплей или принтер.

Наряду с перечисленными устройствами ВВ широко в настоящее время применяются дисководы и магнитофоны - устройства, осуществляющие запись и считывание информации с магнитных носителей (дисков и лент). Развитие лазерной технологии привело к созданию компакт-дисков CD-ROM с емкостью памяти в несколько Гбайт. Ближайшее будущее оптиче­ских дисков связывают с DVD-дисками, приходящими на смену накопите­лям CD-ROM и CD-R. Первые устройства памяти на дисках DBD-RAM ожидаются в начале 1998 года и их емкость составит около 2,6 Гбайт.

На эволюционном пути развития оптических дисков ценной альтернативой может стать магнитооптическая технология ( МО ), способная бросить вызов технологии изменения фазы DVD-RAM.

Процессор ( ЦП ) -это сердце ПК. В настоящее время наилучшим признается процессор класса Pentium Pro - он обеспечивает оптимальную производительность при работе с 32-разрядными операционными системами (ОС) и приложениями. Скорость работы процессора тем больше, чем выше тактовая частота - количество элементарных операций выполняемые им в одну секунду. Для Ptntium Pro тактовая частота на конец 1996 г. достигла 200 MНz и более.

Очень важным элементом ПК является оперативная память, откуда процессор берет программы и исходные данные для обработки, в нее он записывает полученные результаты. Следует помнить, что данные и результаты сохраняются в ней только при включенном компьютере. Для компьютеров с тактовой частотой более 25 Мгц, с целью быстрого доступа к оперативной памяти, предусмотрена КЭШ-память или “сверхоперативная память” небольшого обьема. Добавление КЭШ-памяти приводит к значительно большему увеличению производительности процессора.

Особое место в конструкции ПК занимает Системная плата, на которой кроме ЦП размещаются наборы микросхем, шины и системы BIOS ( базовая система ввода-вывода ). Если ЦП - сердце ПК, то набор микросхем служит его электрокардиостимулятором. Наборы микросхем управляют обменом с памятью, шиной ввода-вывода и во многих случаях встроенными контролерами жесткого диска с интерфейсом. Они обрабатывают также запросы прерывания и прямого доступа в память. Наиболее популярными до сих пор пользуются наборы, изготавливаемые фирмой Intel. Системная BIOS представляет собой часть программного кода, как провило хранящегося в микросхеме программируемой постоянной памяти ( ППЗУ ), которая устанавливается во всех ПК. BIOS выполняет самотестирование при включении питания ( Power-on self-test, POST ) при включении ПК и управляет прерываниями от клавиатуры и передачами через порты.

Монитор - наиболее интенсивно используемая часть компьютера. Правильно выбранный монитор сможет повысить продуктивность работы, а неподходящий монитор способен вызывать головные боли, усталость и чрезмерное напряжение зрения. Краткий обзор основных устройств ПК можно закончить перечислением требований к нему и блоков, выполнение которых и наличие необходимо для подключения к INTERNET. Самый простой способ подготовки пользователя к присоединению к INTERNET - это присвоение каждому ПК адреса IP ( Internet Protocol ). При создании хорошей схемы сети INTERNET можно ограничиться четырьмя видами наиболее распространенных приложений INTERNET - для просмотра WEB, для реализации электронной почты, телеконференций Usenet и доступа FTP ( протокола передачи файлов ). В ПК должны быть установлены быстродействующий ЦП, чтобы максимальна ускорить процесс визуализации документов в формате HTML (язык разметки гипертекста ), 16-Мбайт ОЗУ для выполнения современных программ просмотра, богатство функций которых постоянно растет, и диск большой емкости для приложений, загрузки информации из сети и буферизации страниц HTML. К этому следует также добавить и аппаратный графический акселератор, средства для 16-бит представления цвета, звуковую плату и громкоговорители для работы с насыщенными графикой страницами Web и всеми новыми технологиями мультимедиа Web, требуется и накопитель CD-ROM для некоторых инструментальных средств поиска INTERNET. Конкретная конфигурация зависит от вида прикладных программ, от способов связи пользователей и требуемых уровней производительности и безопасности.

1.2. Представление информации в ЭВМ.

Обьектом передачи и преобразования в ЭВМ является дискретная информация. Носителем информации в ЭВМ является электрический сигнал, у которого меняется какой-либо параметр - частота, амплитуда. Обычно для записи этих сигналов используют магнитные ленты, диски или магнитооптические диски. В процессе ввода, хранения, вывода и обработки информации в ЭВМ осуществляется ее многократное преобразование из одной формы представления в другую. С каждой формой представления информации связаны различные алфавиты. Процесс преобразования информации из одного алфавита в другой называется кодированием. Обратное преобразование информации называют декодированием. Преимущественное распространение получило двоичное кодирование, алфавит которого состоит из двух цифр - 0 и 1.

Прямой перевод чисел из десятичной системы счисления (СС ) в двоичную и наоборот громоздок и в ЭВМ не применяется. Для этих целей используют промежуточную (комбинированную) - двоично-десятичную СС, в которой каждая цифра десятичного числа заменяется 4-разрядным двоичным эквивалентом, т.н. тетрадой. Такой перевод чисел не представляет труда, т.к. достаточно запомнить только двоичные эквиваленты соответствующих десятичных цифр: 0-000, 1-0001, 2-0010, 3-0011, 4-0100, 5-0101, 6-0110, 7-0111, 8-1000, 9-1001.

Например, десятичное число N10= 1988 в двоично-десятичной СС будет иметь вид

1 9 8 8

Nz-10= 0001 1001 1000 1000 (1.1)

По этому же правилу преобразуются и дробные числа.

Перевод чисел из двоично - десятичной в десятичную СС осуществляется в обратном порядке. Подобные преобразования чисел называют методом прямого замещения. Наряду с двоично-десятичной в ЭВМ могут применяться и другие СС - восьмеричная, шестнадцатеричная, двоично -восьмеричная, двоично-шестнадцатеричная. Главными достоинствами этих СС являются компактность записи чисел и простота представления их в двоичном коде. Применение этих СС оказалось удобным для выполнения в ЭВМ ускоренных сдвигов двоичных чисел сразу на четыре и восемь разрядов, необходимых для выполнения операций умножения и деления. Кроме того, они обеспечивают удобную форму представления данных в ЭВМ.

Полуслово=2 байта

|  |  |
| --- | --- |
| 10101101 | 10111101 |

0...........7 8...........15

В настоящее время в современных ЭВМ для кодирования алфавитов применяется байтовый алфавит. В нем в качестве основной единицы представления данных применяется байт, состоящий из восьми двоичных разрядов (бит). Бит - двоичная цифра. В байтовом алфавите байт является минимальной единицей информации, адресуемой и обрабатываемой в ЭВМ, в связи с чем в ЭВМ применяется единая иерархическая структура организации данных, которая включает полуслово, слово двойное слово и поле (рис.1.1)

Байт

|  |
| --- |
| 01101001 |

01234567

Слово = 4 байта

|  |  |  |  |
| --- | --- | --- | --- |
| 01110101 | 11001001 | 10001100 | 10110100 |

0............7 8...........15 16..........23 24.........31

Рис.1.1. Форматы данных.

Двойное слово состоит из 8 байт. Поле имеет переменную длину от 1 до 256 байт.

Знаковый разряд Цифровые разряды

а)

|

Условное место запятой

Знаковый разряд Цифровые разряды

| | | б)

Порядок Мантисса

Рис.1.2. Представление чисел :

а) в естественной форме, б) в полулогарифмической форме

Запись числа в полулогарифмической форме (с плавающей запятой) позволяет значительно расширить диапазон представляемых чисел по сравнению с записью числа с фиксированной запятой при одинаковом числе двоичных разрядов.

С помощью всей совокупности кодовых комбинаций, которые для байта в пределе равны 28, можно закодировать помимо чисел прописные и строчные буквы русского и латинского алфавитов, знаки препинания, математические знаки основные и вспомогательные символы, необходимые для контроля и управления ЭВМ.

Числа в ЭВМ могут быть представлены в двух формах: естественной, которая также носит название формы с фиксированной запятой или точкой, и полулогарифмической, которая еще называется формой с плавающей запятой или точкой.

Для выполнения вычислений с числами в естественной форме необходимо, чтобы они были всегда меньше единицы. Делается это с помощью масштабных коэффициентов. Для записи знака числа отводится один дополнительный разряд (рис.1.2,а). Если число положительное, то в этот разряд записывается 0, если отрицательное, то 1.

Число в нормальной форме имеет вид:

A = npM, (1.2)

где n -основание СС, p -порядок числа, M -мантисса.

В числе, записанном в полулогарифмической форме, положение для записи которого в нем выделяется несколько разрядов (рис.1.2,б). Мантисса должна быть по модулю меньше единицы и первая цифра должна быть значащей. Например, A=110,101001 в соответствии с (1.2) представляется

А = 110,101001 = 10110,110101001.

где n = 102, p = 112, M = 1101010012.

Это же число в нормальной форме будет иметь вид :

Знак 011 Знак 0110 101 001

Порядок Мантисса

Для удобства вычислений мантиссу нормализуют, то есть сдвигают ее влево до тех пор, пока первая цифра после запятой станет равной 1.

2.Общие сведения о программировании

и программном обеспечении

2.1.Понятие алгоритма.

Решение любой задачи - многоэтапный процесс, включающий постановку задачи, выбор метода ее решения и, наконец, составление алгоритма.

Алгоритм - это совокупность предписаний и правил, выполнение которых за конечное число действий преобразует исходные данные задачи в искомый результат.

Существует ряд задач, математические модели решения которых создаются на основе общеизвестных формул и логических условий. В других задачах метод решения можно получить только экспериментально. После выбора метода решения разрабатывается алгоритм. Нет одного общего способа, пользуясь которым можно бы составить любые алгоритмы. Однако алгоритм составляется в той форме, которая наиболее понятна человеку. Рассмотрим, например, запись алгоритма в операторной форме для вычисления корней квадратного уравнения ax2+bx+c=0. Корни этого уравнения находятся по формуле

 (2.1)

Порядок вычислений здесь зависит от знака подкоренного выражения D=b2-4ac. Если D 0,то корни действительные, если D<0-корни мнимые. Выделяем следующие этапы:

1. Вычисление D=b2-4ac

2. Проверка условия D0 или D<0

3.Вычисление действительных корней, если D0

4. Вычисление мнимых корней, если D<0.

В этом процессе решения задачи можно выделить три арифметических этапа (1,3,4) и один логический (2). Кроме этапов непосредственного расчета, вычислительный процесс включает и другие этапы, обусловленные спецификой обработки данных компьютером - ввод исходных данных, перевод их из десятичной в двоичную систему счисления и обратно, вывод результатов расчета, останов ЭВМ или переход к решению другой задачи.

Операторная форма-это запись алгоритма в виде условных буквенно-цифровых символов-операторов, графических знаков-стрелок и знаков препинания. Для некоторых основных и вспомогательных операторов приняты следующие условные обозначения:

А - арифметический оператор

Р - логический оператор

П - оператор переноса

Я - оператор останова

Ф - оператор формирования.

Запись алгоритма решения на ЭВМ квадратного уравнения в операторной форме имеет следующий вид:

ПААРА; ААПЯ, (2.2)

где П-оператор ввода исходных данных,

А-оператор перевода из десятичной СС в двоичную,

А-оператор расчета D=b- 4ac,

Р-логический оператор, проверяющий условие D0 и D<0,

А-оператор расчета действительных корней,

А-оператор расчета мнимых корней,

А-оператор перевода результатов из двоичной СС в десятичную,

П-оператор вывода результатов на печать,

Я-оператор останова ЭВМ.

Алгоритмы, имеющие в своем составе хотя бы один логический оператор, называются разветвляющимися. Алгоритмы, не имеющие логических операторов, называются линейными. Если бы целью решения приведенного уравнения было бы отыскание действительных его корней, то процесс вычислений описывался линейным алгоритмом

ПААААПЯ (2.3)

Наиболее наглядная и чаще всего употребляемая форма представления алгоритмов с помощью различных геометрических фигур - блок - схемная. Все блоки обозначаются номерами. Внутри блоков приводятся данные, раскрывающие содержание выполняемых действий: начало и конец алгоритма, ввод-вывод данных, различные процессы обработки данных, логические и другие операции. Все геометрические фигуры в блок-схеме алгоритма должны изображаться в соответствии с ГОСТ 19002-80 и ГОСТ 19003-80.

Блок-схему алгоритма рисуют сверху вниз блок за блоком. Блоки соединяются между собой прямыми линиями со стрелками на конце или без них, показывающими направление процесса обработки данных. Каждое такое направление называется линией потока или ветвью алгоритма. Направление ветвей, идущих сверху вниз или слева направо, принять за основное и стрелками не обозначается. Каждая ветвь исходит и подходит обычно к середине блока. Блок-схема не только достаточно наглядно описывает алгоритм, но и является основой для составления программы. Следует иметь в виду, что один и тот же алгоритм может использоваться при решении различных по содержанию задач, если они были предварительно классифицированы по виду математических моделей - каждый класс математических моделей имеет свой метод решения.

На рис. 2.2 приведена блок-схема решения квадратного уравнения, решение которого имеет вид (2.1).

1

Начало

2

Определить

a, b, c

3



D>0 4 D<0



5 7

D=0 6

  

8 9 10

  

11

конец

Рис.1.2. Блок-схема решения квадратного уравнения.

2.2. Основные понятия программирования

Программа - это представление алгоритма с помощью специальных символов, записанных в форме, удобной для обработки компьютером. Вопросы, связанные с методами разработки программ, привели к созданию программирования.

Программирование - это раздел прикладной математики, разрабатывающий методы использования ЭВМ для реализации алгоритмов.

Логическая схема алгоритма, составленная в операторной форме или представленная графически в виде блок-схемы - начальный этап подготовки решения любой задачи на ЭВМ. Конечный этап - запись программы в машинных кодах или на машинном языке. Машинный язык - это система представления команд и данных в конкретных ЭВМ. Система команд любого ПК содержит полный перечень операций, которые выполняются им.

Программа, написанная на машинном языке, состоит из последовательности пронумерованных строк, каждая из которых содержит код команды или код числа, причем номер строки является адресом ячейки памяти, в которой хранится код конкретной команды или конкретного числа. В такой программе первая строка обычно содержит операцию ввода исходных данных и команд в память ПК. Вычислительную часть программы завершают команды, содержащие коды операций печати результата и останова машины. В конце программы, в последних строках, обычно указывают адреса ячеек, в которые записаны исходные данные. Составление программ на машинном языке называлось ручным программированием и широко использовалось в ЭВМ первого поколения и в ряде специализированных ЭВМ или вычислительных устройствах на основе микропроцессоров.

Большая трудоемкость ручного программирования, сложность отладки задач на машинном языке привели к созданию современных алгоритмических языков высокого уровня - Бейсик, Фортран, Паскаль, С, АДА, JAVA. Машинная программа, в этом случае, получается путем перевода программы с символического языка на машинный самой ЭВМ по специальной программе, называемой транслятором. (рис. № )

1 2 3

Рис.2.2. Структурная схема транслятора :

1 - программа на символическом языке,

2 - транслятор,

3 - программа на машинном языке.

Область применения ЭВМ непрерывно расширяется, что требует дальнейшего совершенствования программного обеспечения, направленного как на создание новых средств описания алгоритмов различных задач сложной структуры, ток и на создание более эффективных языков программирования. Под языками программирования понимают формальные языки связи человека с ЭВМ, предназначенные для описания данных и алгоритмов ( программ ) их обработки в ПК.

Ниже в сл.(?) приводится описание алгоритмического языка Бейсик, реализованного практически во всех моделях ПК. В настоящее время существует несколько версий этого языка, обусловленных спецификой ЭВМ. Однако, зная одну из версий Бейсика, легко перейти к работе на другой версии - различия между ними несущественны. При составлении Бейсик - программ соблюдают четыре правила:

текст программы записывается печатными буквами и цифрами одного размера,

недопустимо использование подстрочных и надстрочных записей,

для наглядного представления фрагментов программы на экране дисплея рекомендуется в одной строке использование не более 80 символов,

каждая новая строка начинается с ее номера.

Бейсик снабжен интерпретатором, обеспечивающим быструю и эффективную разработку программ и компилятором, обуславливающим высокую эффективность машинной программы. Компилятор - это транслятор, обеспечивающий перевод программы, написанный на алгоритмическом языке, в совокупность машинных команд без их выполнения. Характерной особенностью Бейсика является непосредственное взаимодействие пользователя и ЭВМ, при котором пользователь сразу же получает ответы на все свои запросы, посланные в компьютер.

3.Алфавит и основные операторы языка Бейсик

3.1.Общие сведения и структура программы на Бейсике.

Бейсик - язык программирования высоко уровня и в нем реализованы следующие основные возможности:

-четыре типа данных : целые (от -32 768 до +32767); вещественные с простой точностью(7 десятичных цифр),вещественные с двойной точностью (16 десятичных цифр); строки символов (до 255 символов);

-многомерные массивы;

-последовательные файлы и файлы с произвольным доступом;

-воспроизведение звука;

-работа с графикой;

-форматный вывод;

-работа с таймером;

-прямое обращение к основной памяти и портам;

-использование подпрограмм на машинном языке;

-обработка ошибочных ситуаций и др.событий;

-средства отладки;

-широкий набор встроенных функций.

Система программирования Бейсика включает интерпретатор и компилятор, обеспечивающий более высокую скорость выполнения программ.

Программа на Бейсике представляет собой последовательность строк ,каждая из которых содержит номер и информационную часть.

Номер строки - целое число, в котором от 1 до 5 цифр. Оно принимает значения от 0 до 65529.Номера строк, кроме определения порядка размещения, используются и для указания строк при передаче уравнения.

Информационная часть состоит из одного или нескольких операторов(команд), в том числе и комментариев. Если строка содержит несколько операторов, то между ними ставится двоеточие. Комментарий размещается после операторов и отделяется от них апострофом(или двойными кавычками).

В Бейсике используются английские ключевые слова, значения которых отражают назначение оператора, функции или команды.

3.2. Основные элементы языка

3.2.1. Алфавит языка и комментарий

При программировании на языке Бейсик используются прописные и строчные буквы латинского алфавита, цифры от 0 до 9 и специальные знаки:

“.”-(точка), ”,”-(запятая), ”;”-(точка с запятой), ”<<“-(кавычки), ”’”-(апостроф), ”( )”-(круглые скобки), ” “-(пробел), ”$”-(знак денежной единицы ), ”&”-(коммерческое “и”), ”\”-(знак деления нацело), ”%”-(процент),

“№”-(номер), ”?”-(знак вопроса).

Знаки арифметических операций :”+”- (плюс), ”-”-(минус), ”\*”-(умножение), “/”-(деление), ”^”-(возведение в степень).

Знаки отношений “>“, ”<“,”<=“, ”>=“, ”<>“. В комментариях возможно использование букв русского алфавита.

Комментарий - любая последовательность символов. Он может быть задан либо с помощью апострофа, либо с помощью оператора REM:

REM ( комментарий ).

Если в строке записывается несколько операторов, то комментарий должен быть последним. И если в этом случае используется апостроф, то знак “:”, разделяющий операторы в строке, перед комментарием не записывается.

3.2.2. Простейшие конструкции языка

В Бейсике используются константы , переменные и массивы .Они представляют данные двух типов: числовые и строковые.

Константа - это такой обьект программы, который сохраняет в ней постоянное значение. Тип и значение константы определяются ее записью. Различают два типа числовых констант:

целочисленные и вещественные. Запись чисел близка к естественной. При записи вещественных чисел для отделения целой части от дробной используется точка вместо запятой. Очень большие или очень маленькие числа можно записывать в форме с порядком, где вместо основания степени 10 употребляется буква E.

Примеры записи:

|  |  |
| --- | --- |
| Число | Запись на языке Бейсик |
| 7  -6.3  0.37  106  -5.4\*10-8 | 7  -6.3  0.37 или .37  1E6 или 0.1E7 и т.д.  -5.4E-8 или -.54E-7 и т.д. |

При записи целых чисел к числу нужно приписать %, например 40%.

Переменная-это такой обьект программы, который в процессе выполнения программы может иметь различные значения. Для обозначения переменных используются имена, образуемые из букв, цифр и точек. Первым символом имени должна быть буква. Тип переменной может быть задан либо явно, либо с помощью оператора DEF..., либо по умолчанию. Явное задание типа переменной определяется последним символом имени переменной: знак % определяет целый тип, знак $ - строковую переменную.

Массив представляет собой последовательность переменных. Каждый массив имеет имя и размерность. Имена массивов и их тип определяются аналогично простым переменным. Для явного задания массива используется оператор DIM . Например:

30 DIM A20$(6), A2%(2,3), где A20-одномерный массив с данными строкового типа и A2-двумерный массив целого типа.

Стандартные функции. Для вычисления часто употребляемых функций в памяти ЭВМ имеется набор стандартных программ:

|  |  |  |  |
| --- | --- | --- | --- |
| Запись на  Бейсике | Математическое  определение | Запись на  Бейсике | Математическое  определение |
| ABS(x)  TAN(x)  ATN(x)  COS(x)  SIN(x) | |x|  tn x  arctn x  cos x  sin x | EXP(x)  LOG(x)  SQR(x)  SGN(x) | ex  ln x  x1/2  1, x>0  0, x=0  -1, x<0 |

В качестве аргумента (в круглых скобках ) можно употреблять любое арифметическое выражение.

Арифметические выражения соответствуют общепринятым алгебраическим выражениям, в которые входят числа, переменные, функции, соединенные знаками арифметических операций

3.3. Операторы языка Бейсик

Оператор присваивания служит для присваивания переменной значения арифметического выражения и имеет вид

LET V= арифметическое выражение (LET можно опустить).

Оператор безусловного перехода имеет вид

GOTO N, где N-номер строки.

Он используется при организации разветвлений.

Условные операторы служат для изменения порядка выполнения операторов в зависимости от какого-либо условия. Они используются для организации циклов и разветвлений и имеют вид

IF условие THEN N или

IF условие GOTO N

IF условие THEN оператор

где условие представляется как

(арифм. выражение) Q (арифм. выражение)

Q одно из отношений <, <=, >, >=, =, <>.

Если условие удовлетворяется, то осуществляется переход к строке с номером N. Если условие не удовлетворяется, то осуществляется переход к оператору, следующему за условным (см. пример 1).

Пример 3.1. Программа табулирования функции y=x2 ln x в интервале [2, 4] с шагом 0,2.

10 X=2

20 Y=X\*X\*LOG(X) : PRINT X,Y : X=X+0.2

30 IF X<=4 GOTO 20

Операторы цикла -это пара операторов FOR-NEXT. Общий вид операторов

FOR V=A0 TO An STEP H

........................................

NEXT V

где V - неиндексированная переменная - управляющая переменная цикла;

A0, An, H-начальное и конечное значения и шаг изменения переменной цикла.

Если H=1, то STEP 1 можно опустить.

Операторы между FOR и NEXT образуют тело цикла. Если при выполнении оператора FOR обнаружено, что условие завершения цикла выполнено, то управление передается оператору, следующему за оператором NEXT. Это происходит, например, при начальном значении управляющей переменной больше конечного, а приращение имеет положительное значение.

Пример 3.2. Программа вычисления суммы 200 элементов массива ALFA:

210 SUM=0

220 FOR I=1 TO 100

230 SUM=SUM+ ALFA(I)

240 NEXT I

Кроме операторов FOR-NEXT в Бейсике запускается организация цикла по условию с помощью операторов WHILE и WEND. Первым из них цикл начинается, а вторым - завершается. Работа цикла будет завершена тогда, когда логическое выражение в операторе WHILE будет “FALSE”. В этом случае после оператора WHILE управление будет передано оператору, следующему за оператором WEND.

Пример 3.3. Вычисление суммы ряда S=1+1/2+1/4+....+1/2n,пока член ряда не станет меньше 10-6.

10 SUM=1: N=1

20 WHILE 0.5^N>1E-6

30 SUM=SUM+0.5^N

40 N=N+1

50 WEND

Циклы могут быть вложенными друг в друга. Внутренние циклы должны целиком помещаться во внешних . Подряд идущие операторы NEXT могут быть заменены одним NEXT с управляющими переменными всех вложенных циклов, разделенными запятыми.

Пример 3.4. Программа формирования единичной матрицы.

20 DIM A(N,N)

30 FOR I=1 TO N : FOR J=1 TO N

40 IF I<>J THEN 60

50 LET A(I,J)=1 : GOTO 70

60 LET A(I,J)=0

70NEXT J : NEXT I

80 END

Операторы ввода-вывода. Операторы READ (переменные), DATA (числа) присутствуют в программе одновременно. Данные, подлежащие вводу, могут располагаться в одном или в нескольких операторах DATA, но в том порядке, в котором данные должны использоваться оператором READ. Оператор RESTORE обеспечивает возврат в начало списка данных, задаваемых DATA , для многократного считывания.

Пример 3.5.

10 READ A,B,C,D

20 DATA 3.4, 0.3, 10, 1.5

30 RESTORE : READ E,F

Будем иметь: A=3.4, B=0.3, C=10, D=1.5, E=3.4, F=0.3.

Оператор INPUT позволяет вводить данные с клавиатуры дисплея и имеет вид

INPUT(список переменных).

Оператор PRINT (список) выводит на экран или на принтер результаты вычисления. При выводе текста применяют кавычки или апострофы. Арифметическое выражение в списке перед выводом сначала вычисляется. Использование в качестве разделителя запятой, при выводе осуществляет последний в зонном формате - экран разделен на 5 зон по 14 позиций. Если разделитель “;”, то вывод осуществляется в плотном формате через пробел.

Символьные переменные задаются либо с помощью операторов INPUT,READ,DATA, либо оператором присваивания. Строка символов при этом заключается в кавычки. Задаются они буквой или буквой с цифрой с добавлением знака денежной единицы $, например A$=‘Волгоград’.

Пример 3.6. Вывод городов и их индексов.

10 PRINT “Город”; TAB(20); “Индекс”

20 READ A$

30 IF A$=“ ” THEN END

40 READ B$

50 PRINT A$; TAB(20); B$

60 GOTO 20

70 DATA Волгоград, 400 000, Воронеж, 394 000, “ ”

Здесь для установки позиций использована встроенная функция TAB(N), где N-целое число, номер позиции.

Подпрограммы (группа операторов) служат для выполнения одного и того же алгоритма в разных местах программы. Передача управления к подпрограмме осуществляется с помощью операторов GOSUB N, где N- номер строки и RETURN (обозначает возврат к программе).

Для прекращения выполнения программы служит оператор STOP, который в отличие от END не закрывает используемых в программе файлов и рекомендуется применять его при отладке программы.

Пример 3.7. Программа обращения матрицы методом Гаусса.

05 REM ‘ Обращение матрицы методом Гаусса

10 PRINT ‘Задайте размерность N=‘

15 INPUT N : DIM A(N,N), P(N,N)

20 DIM B(N), C(N,N), G(N), X(N)

25 FOR I=1 TO N : FOR J=1 TO N

30 PRINT ‘Введите A’ I’, ‘J’=‘ : INPUT A(I,J)

35 LET P(I,J)=A(I,J) : NEXT J : NEXT I

40 FOR J2=1 TO N : FOR I=1 TO N

50 LET B(I)=0 : NEXT I

60 LET B(J2)=1

70 FOR J3=1 TO N : FOR J4=1 TO N

80 LET A(J3,J4)=P(J3,J4) : NEXT J4 : NEXT J3

85 GOSUB 100 : FOR I=1 TO N

90 PRINT ‘X’I’, ‘J2’=‘ X(I)

95 NEXT I : NEXT J2 : STOP

100 LET N1=N-1 : FOR K=1 TO N1

110 IF ABS (A(K,K))>0 GOTO 200

120 LET K1=K+1 : FOR M=K1 TO N

130 IF ABS (A(M,K))>0 GOTO 150

140 GOTO 210

150 FOR L=1 TO N : LET V=A(K,L) :

LET A(K,L)=A(M,L)

160 LET A(M,L)=V : NEXT L : NEXT M

170 LET V=B(K) : LET B(K)=B(M) :

LET B(M)=V

200 LET G(K)=B(K)/A(K,K) : LET K1=K+1

210 FOR I=K1 TO N : LET B(I)=B(I)-A(I,K)\*G(K)

220 FOR J1=K TO N : LET J=N-J1+K :

LET C(K,J)=A(K,J)/A(K,K)

225 LET A(I,J)=A(I,J)-A(I,K)\*C(K,J)

230 NEXT J : NEXT I : NEXT K

240 LET M=N : LET X(M)=B(M)/A(M,M)

250 LET M=M-1 : LET S=0 : FOR L=M TO N1

260 LET S=S+C(M,L+1)\*X(L+1) : NEXT L

270 LET X(M)=G(M)-S : IF M>1 GOTO 250

280 RETURN : END

1. Некоторые общие и системные команды языка Бейсик

|  |  |
| --- | --- |
| Запись на Бейсике | Обозначение |
| CATALOG | Чтение каталога с ГМД. По этой  директиве на экран дисплея выводится каталог ГМД. |
| SAVE имя файла | Записывает программу из памяти на ГМД; если на ГМД уже есть файл с таким именем, он теряется и заменяется на новый. |
| RENAME файл1, файл 2 | Переименование файлов. Здесь  файл 1 - старое имя файла, файл 2 - новое имя файла. |
| DELETE имя файла | Уничтожение файлов. |
| LOCK имя файла | Защита файла на запись. Файл, защищенный от записи, в каталоге помечается (\*). |
| UNLOCK имя файла | Отменяет защиту от записи в файл. |
| LOAD имя файла | Загружает программу в памятьс ГМД. Программа и данные, ранее находящиеся в памяти,теряются. |
| RUN имя файла | Загружает и запускает программу с ГМД. Программа и данные, находящиеся в памяти ранее, теряются. |
| RUN | Запускает програмы со строки с наименьшим номером; для запуска с промежуточной точки можно указать № строки (н-р, RUN200). |
| NEW | Устанавливает начальное состояние памяти, стирает программу и данные. |
| STOP | Останавливает программу с выдачей текста строки, в которой произошел останов. |
| END | Завершает программу. |

3.5.Некоторые команды редактирования языка Бейсик

|  |  |
| --- | --- |
| Запись на Бейсике | Обозначение |
| LIST | Выдача текста программы; выдает весь текст |
| LIST 100, 200 | Выдает строки с номерами 100 и 200 |
| LIST 100, | Выдает строки с номерами не меньше 100 |
| LIST , 200 | Выдает строки с номерамине больше 200 |
| LIST 150 | Выдает строку 150 |
| REM | Комментарий до конща строки, при выполнении  игнорируется |
| VTAB Y | Установливает курсор на строку экрана с номером Y (0≤Y≤31) |
| HTAB X | Перемещает курсор на X-ю позицию текущей экранной строки (нумерация строк и позиций на экране сверху вниз  и слева направо) |
| TAB (X) | Применяется в оператоне PRINT, перемещает курсор на  X-ю позицию |
| HOME | Очищается экран, курсор перемещается в верхний левый угол экрана |
| FLASH | Включает мерцающий режим вывода текста |
| INVERSE | Включает инверсный (черный по светлому фону) режим вывода текста |
| NORMAL | Включает прямой(светлый по черному) режим вывода текста |
| SPEED=X | Задает скорость вывода мекста на экран (0≤X≤255) |

3.6.Некоторые команды графики языка Бейсик

|  |  |
| --- | --- |
| Запись на Бейсике | Обозначение |
| GR=N | Включение графики низкого разрешения (64×64), N-номер страницы размером  2 Кбайта, при этом 2≤N≤31 |
| MGR=N | Включение графики среднего разрешения (128×128), N-номер страницы размером 8 Кбайт, при этом 1≤N≤7 |
| HGR=N | Включение графики высокого разрешения. Параметры такие же, как у оператора MGR. Команды включения графических режимов очищают включенную страницу. |
| COLOR=X | Установливает цвет в последующих операторах вывода графической информации; 0≤X≤7 |
| PLOT X,Y | Помещает точку текущего цвета в X-ю позицию Y-й строки экрана |
| PLOT X0,Y0 TO X1,Y1 | Рисует прямую линию текущего цвета из точки (X0,Y0) в точку (X1,Y1). |
| TEXT=N | Устанавливает текстовый режим; N - номер текстовой страницы. Номера 0-31 соответствуют страницам размера 32×32 символа; номера от 32 до 63 - страницам размера 64×32 символа. |

Список рекомендуемой литературы

1. Лизько Ю.В., Мигунов Л.А. Вычислительная техника. - М.: Воениздат, 1989.- 272 с.
2. Абрамов С.А., Зима Е.В. Начала информатики. - М.: Наука, 1989.- 256 с.
3. Светозарова Г.И. и др. Практикум по программированию на языке БЕЙСИК. - М.: Наука, 1988.- 363 с.
4. Электронные вычислительные машины : В 8-ми кн. / Под ред. А.Я.Савельева. Кн. 2. Основы информатики. - М.: Высш. шк., 1987. -127 с.
5. Воробьёва Г.Н., Данилова А.И. Практикум по вычислительной математике. - М.: Высш. шк., 1990. - 208 с.