МОУ СОШ №51

Информатика:

Quick Basic

Выполнил: Додан В.В.

Новосибирск, 2005

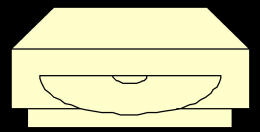
Аннотация.

Цель данного проекта- помочь начинающим изучить язык программирования Quick Basic.

1.
2. Операция
3. Ввод данных с клавиатуры:
4. Ввод данных: Data/Read,
5. Вывод данных на экран (печать данных) в текстовом режиме
6. Основные арифметические
7. Функции, определяемые
8. Действия над символьными
9. Вызов
10. Ветвление (условный переход,
11. Множественный
12. Организация
13. Обработка массивов
14. Процесс обработки данных массива

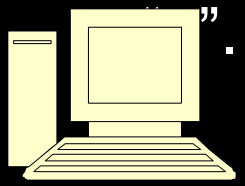
Предисловие

Язык Basic, разработанный в 1963-1964 годах в Дартмутском колледже (США), в его сов- ременных версиях является языком, позволяющим профессионально использовать все сов- ременные методы и технологию программирования. То, что Basic- системы в настоящее время входят в комплект поставки MS-DOS фирмы Microsoft говорит о том, что они очень популярны и по своим возможностям сравнимы с системами программирования на других распространённых языках (Си, Паскаль).



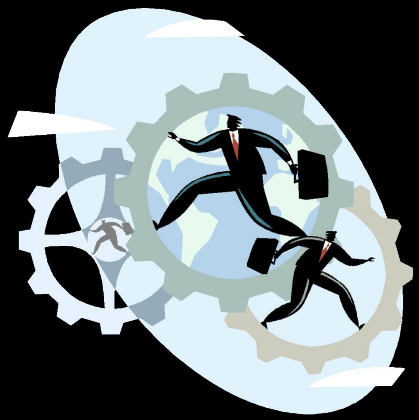
Операция присваивания.

“:=” – операции присваивания, в программе записывается как



”.

Программа	Ячейки памяти
	a
10 a=2	2
	b
20 b=a*3	6
	a
30 a=a+3	5

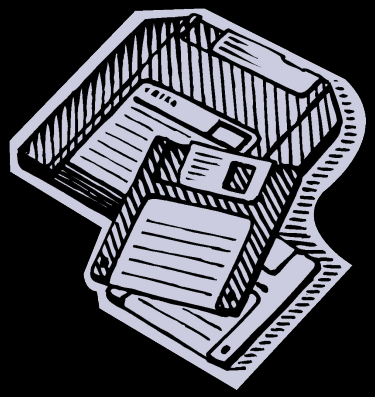


Ввод данных с клавиатуры: Input.

- Встретив команду Input, машина останавливает выполнение программы и ждёт ввода данных с клавиатуры. Набор данных заканчивается нажатием клавиши “ввод”. *Оператор*
- *Input “не умеет” считать, поэтому нельзя набирать в ответ на “?” арифметическое выражение, т.е. нельзя ввести число , набрав 3/7.*

Ввод данных: Data/Read, Restore

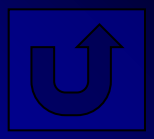
- Команда Read считывает в ячейки памяти данные, перечисленные в программе в строке Data. Команда Restore восстанавливает список Data после считывания. *Команда Read “не умеет” считать и список Data не может содержать знаков арифметических выражений (нельзя записывать 3/7)*



Данные всех строк Data образуют единый набор данных, независимо от того, в каком месте программы они расположены.

Вывод данных на экран (печать данных) в текстовом режиме: команда Print .

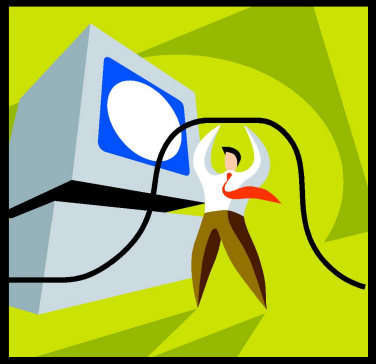
- Результаты работы можно вывести для просмотра на экран, распечатать на принтере, сохранить для дальнейшего использования в файле на магнитном диске. Вывод информации на экран может осуществляться в текстовом или графическом режимах.
- При работе в текстовом режиме на экран можно вывести любые символы (буквы, цифры, знаки препинания, специальные символы). Если начать заполнять экран символами, то можно увидеть, что они выстроятся ровными рядами и столбцами.



Locate номер строки, номер столбца – позволяет начать вывод с любой позиции экрана. Команда Print выводит информацию на экран, LPrint- на принтер.

Основные арифметические операции

(В порядке приоритета)
^ - возведение в степень
* - умножение
/ - деление
\ - целочисленное деление
+ - сложение
– - вычитание



- Кроме чисел и имён ячеек памяти, содержащих числа, в записи арифметических действий могут участвовать функции. Различают функции “встроенные” и функции, определяемые пользователем.
- “Встроенные” – функции, действия которых уже описаны в языке.
- Определяемые пользователем – функции, действие которых программист описывает в программе сам.

Функции, определяемые пользователем

- Описав такую функцию, к ней можно обратиться для выполнения с конкретными значениями. Строка с командой описания функции ничего не вычисляет и не выполняет, она определяет имя новой функции (имя должно начинаться с букв FN) и задаёт правила её выполнения (описывая аргументы и действия над ними).

Действия над символьными данными (над цепочками литер)

- Над символьными данными можно проводить операцию конкатенации и операцию сравнения. Символы можно сравнивать. В программе записывается сравнение символьных данных, компьютер сравнивает соответствующие коды.

Вызов подпрограмм

- GOSUB номер строки (метка)

- Номер первой строки подпрограммы. Последняя команда подпрограммы – RETURN. Оператор RETURN возвращает управление программой в строку, следующую за строкой:
- GOSUB номер (метка)
- вызвавшей подпрограмму.

Ветвление (условный переход, выбор)

Полная альтернатива (предусмотрены команды в ветви “ДА” и ветви “НЕТ”)

- IF условие THEN Номер строки, с которой начинается описание блока команд ветви “ДА” ELSE Номер строки, с которой начинается описание блока команд ветви “НЕТ”

Между блоками “ДА” и “НЕТ” должна быть строка с
командой GOTO номер строки, с которой программа
продолжается после окончания ветвления.

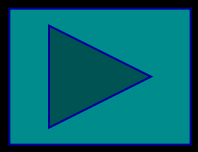
Не полная альтернатива (предусмотрены команды только в одной ветви).

IF условие **THEN** действие

Если ветвь содержит несколько команд, то удобнее условие
записывать так, чтобы команды оказались в ветви “НЕТ”,
тогда запись выбора в программе будет иметь вид:

IF условие **THEN** Номер строки, с которой продолжается программа
Описание команд ветви “НЕТ”
после окончания ветвления

Примечание.
В современных версиях (например QBasic) определены две формы записи конструкции ветвления:

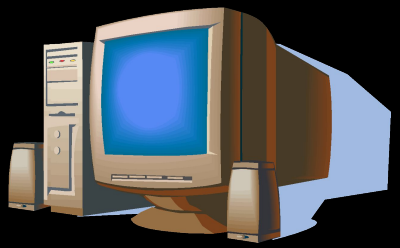


1. **Линейная форма записи** (команды **IF, THEN, ELSE** записаны в одной строке)

IF условие **GOTO** Номер строки для перехода к командам ветви “ДА” **ELSE** Номер строки или описания команд ветви “НЕТ”

IF условие **THEN** Номер строки или описания команд ветви “ДА” **ELSE** Номер строки или описания команд ветви “НЕТ”

Описание команд ветви – набор команд, разделённых при записи знаком “ : ”.



2. Блок выбора

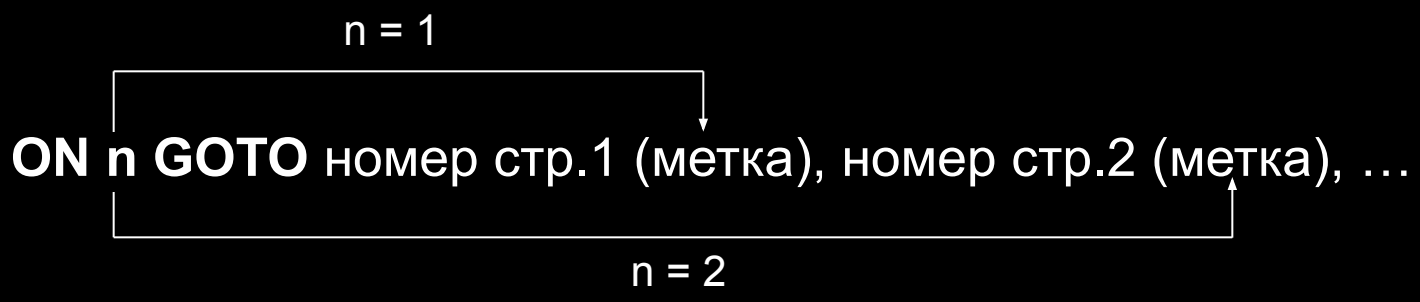
Пример.

IF условие **THEN**
    блок команд
    ветви “ДА”
ELSE
    блок команд
    ветви “НЕТ”
END IF

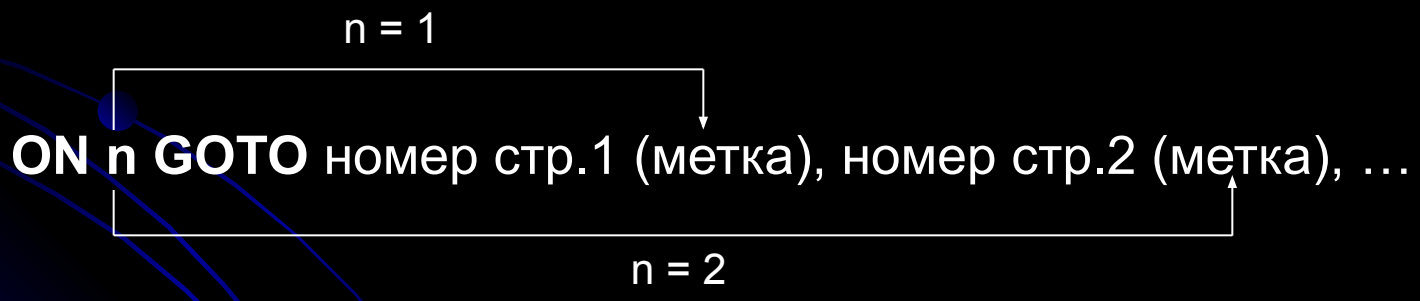
```
CLS : INPUT a
IF a = 1 THEN
  PRINT “работает ветвь ‘ДА”
  PRINT a
ELSE
  PRINT “работает ветвь ’НЕТ”
  PRINT a
END IF
END
```

Условие – это логическое выражение, которое может содержать.

Множественный выбор



- Переход к одной из указанных программных строк (меток) в зависимости от числового значения в ячейке n (число в ячейке в случае необходимости округляется до целого)



Выбор одной из подпрограмм. Номер стр.- номер первой строки подпрограммы.

Расширенные сведения о ‘выборе’:

В современных версиях языках (QBasic) определена структура множественного выбора:

SELECT CASE (переключатель)
  CASE условие 1
    1-ый блок команд
  CASE условие 2
    2-ой блок команд
  *
  *
  *
CASE ELSE
    n - ый блок команд
END SELECT

Переключателем может служить выражение значение которого определяет выбор или имя ячейки памяти, тогда выбор определяет содержимое этой ячейки.

Условие **CASE** может быть записано в одной из трёх форм:

1) **CASE** константа, константа …
2) **CASE IS** знак отношения – константа
3) **CASE** константа 1 **TO** константа 2

Сначала определяется значение переключателя, стоящего после слов **SALECT CASE**. Затем выполняется проверка: удовлетворяет ли это значение одному из **CASE условий**.

1-я форма **CASE условия**:

блок команд выполняется, если значение будет равно одной из констант.

2-я форма **CASE условия**:

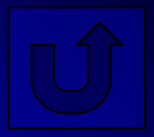
блок команд выполнится, если выполняется условие отношения между значением переключателя и константой.

3-я форма **CASE условия**:

блок команд выполнится, если выполняется соотношение:

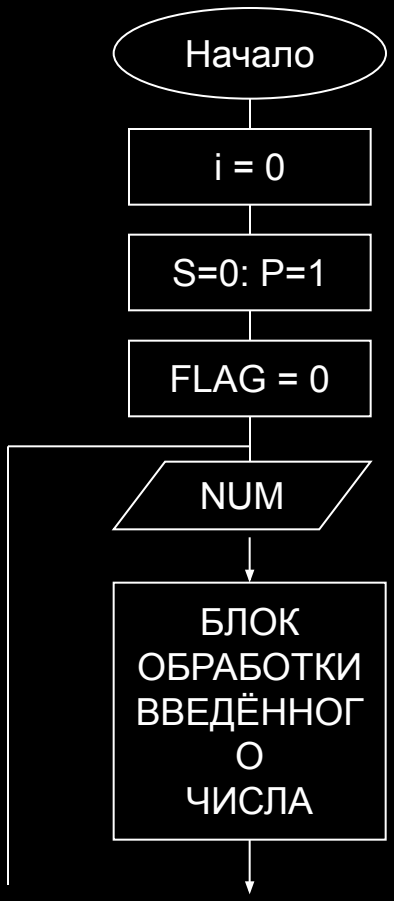
константа ≤ значение ≤ константа 2.

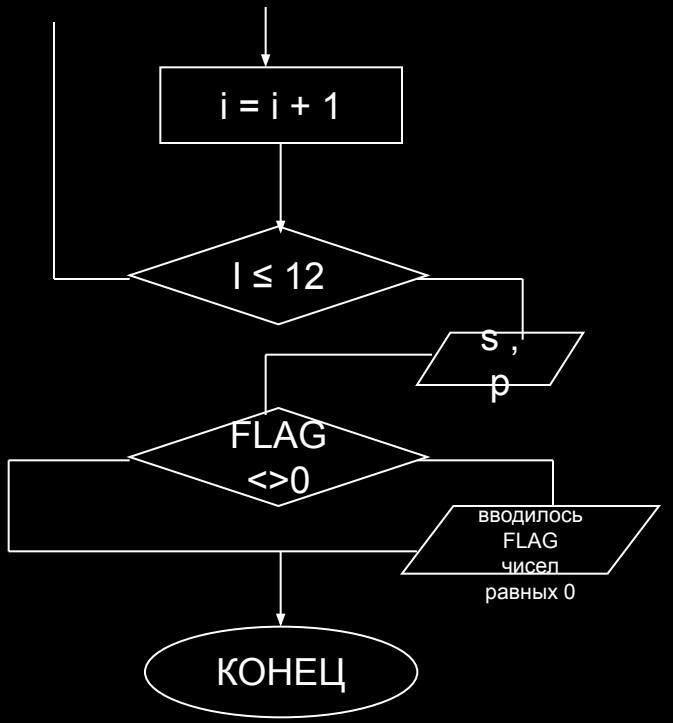
Если ни одно из **CASE условий** не выполняется, то работает блок команд, определяемый строкой **CASE ELSE**.



Организация циклов

- Любой вариант многократного выполнения набора команд, составляющих тело цикла, можно организовать с помощью конструкции ветвления.



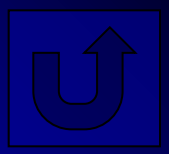


- **Ячейка “счётчик” i.** При каждом выполнении тела цикла значение ячейки увеличивается на 1, ячейка “считает”.
- **Ячейки “копилки” S и P.** Если число, поступающие в ячейку **NUM**, положительно, то оно добавляется к значению **S** – в ячейке **S** копится сумма. Если число в **NUM** отрицательно, оно меняет значение **P** – в ячейке **P** копится произведение. (Готовя ячейку под произведение, туда надо положить 1)
- **Ячейка “флаг” FLAG.** Меняет своё первоначально установленное значение, сигнализируя о каком-либо событии в программе (в данном случае о вводе нуля в ячейку **NUM**).

Для более компактной записи цикла в программе существуют
специальные управляющие конструкции организации циклов:
FOR…NEXT и **WHILE…WEND**

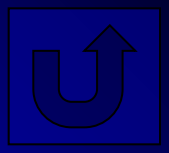
В версиях QBasic дополнительно определён универсальный оператор цикла DO…LOOP. Он имеет несколько форм записи:

DO WHILE условие **Команды тела цикла** **LOOP**	Тело цикла выполняется до тех пор, пока УСЛОВИЕ ИСТИННО (может не выполниться ни разу).
DO UNTIL условие **Команды тела цикла** **LOOP**	Тело цикла выполняется до тех пор, пока УСЛОВИЕ ЛОЖНО (может не выполниться ни разу)
DO **Команды тела цикла** **LOOP WHILE условие**	Тело цикла выполняется до тех пор, пока УСЛОВИЕ ИСТИННО (выполнится хотя бы один раз).
DO **Команды тела цикла** **LOOP UNTIL условие**	Тело цикла выполняется до тех пор, пока УСЛОВИЕ ЛОЖНО (выполнится хотя бы один раз).



Обработка массивов данных

- Если набор данных объединён одной общей задачей их обработки (перебор данных, их анализ, однотипное изменение и др.), то о таком наборе принято говорить как о массиве данных.
- При обработке такого массива вручную на бумаге данные обычно размещают в таблицах.
- Для обработки в компьютере массив данных можно разместить в форме таблицы во внутренней памяти на время работы программы или предварительно записать в отдельный от программы файл на магнитном диске и во время работы программы считывать данные из файла.



Процесс обработки массива

- Для объявления таблиц (массивов) используется оператор DIM (Dimension-размер). После слова DIM перечисляются через запятую имена массивов и рядом в скобках указывают максимальные значения (границы изменения) индексов. Компьютер нумерует ячейки таблиц начиная с 0, поэтому для размещения 50 чисел должен быть объявлен массив L(49).
- Для ввода, вывода и обработки массива используют циклы.