С. Г. Никитенко

СВОБОДНОЕ ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ

BASIC-256 для школы

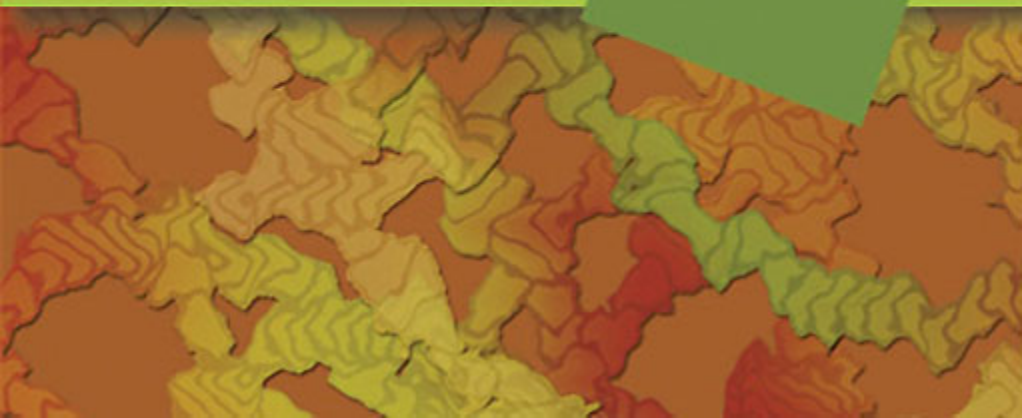
Возможность использования в различных ОС:
Windows, Linux, Mac OS X

Основы программирования в школьном курсе ИиИКТ

Типовые задачи, в том числе тесты ЕГЭ

Индивидуальные зачетные работы и проекты

ИНФОРМАТИКА И ИНФОРМАЦИОННО-КОММУНИКАЦИОННЫЕ ТЕХНОЛОГИИ



УДК 681.3.068(075.3)+800.92Basic-256
ББК 32.973.26–018.1я721
Н62

Никитенко С. Г.

Н62 Свободное программное обеспечение. BASIC-256 для школы. — СПб.: БХВ-Петербург, 2011. — 224 с.: ил. — (ИиИКТ)

ISBN 978-5-9775-0740-0

Для свободно распространяемой кроссплатформенной среды BASIC-256 рассматриваются: интерфейс пользователя, основные операторы, правила написания программ, примеры решения типовых школьных задач из курса информатики и ИКТ, методика решения заданий ЕГЭ. Подробно рассмотрены часто встречающиеся на олимпиадах различного уровня темы: ввод/вывод числовых и строковых данных посредством текстовых файлов и построение графиков функций. Приведены задания для тренировки учащихся, проведения зачетных работ и выполнения проектов. Все зачетные работы и проекты ориентированы на выполнение по строго индивидуальным заданиям. Также книга содержит справочную информацию по BASIC-256.

Для образовательных учреждений

УДК 681.3.068(075.3)+800.92Basic-256
ББК 32.973.26–018.1я721

ISBN 978-5-9775-0740-0

Оглавление

ГЛАВА 1

Начальные сведения о BASIC-256

1.1. Что такое BASIC-256?

BASIC-256 — свободная версия популярного среди учителей информатики интерпретатора языка BASIC, используемая в соответствии с правилами GNU General Public License. Авторское право на эту программу принадлежит с 2006 г. группе The BASIC-256 Team. Первоначальное название языка — KidBasic.

Разработаны версии интерпретаторов языка для ряда платформ, в частности, для Linux и для 32-битных версий Windows. В состав серии дистрибутивов ALT Linux 5.0.2 Школьный (январь 2011 г.) включен BASIC-256 версии 0.9.6.58 со встроенным справочником в переводе В. Чёрного и С. Ирюпина. В июне 2010 г. для 32-битных версий Windows, включая соответствующие версии Windows Vista и Windows 7, разработан BASIC-256 версии 0.9.6p.

Сайт группы разработчиков (англоязычный) размещен по адресу [1]. Дистрибутив BASIC-256 версии 0.9.6p можно загрузить со страницы [2]. Руководство по основным структурам данных и операторам BASIC-256 версии 0.9.5 на английском языке доступно на странице [3], а на немецком языке — на странице [4].

В BASIC-256 версии 0.9.6p имеется встроенный англоязычный справочник RU-BASIC-256 Reference 0.9.5*x*, вызываемый по клавише <F1> [5]. Справочник [5] и руководство [7] в наибольшей степени соответствуют взятым за основу в пособии версиям

0.9.6p (2010-06-10) для Windows и 0.9.6.58 (2011-01-07) для ALT Linux 5.0.2 Школьный.

1.2. Интерфейс пользователя BASIC-256

После установки русской версии BASIC-256 (например, v.0.9.6p) запуск ее выполняется из меню Windows командой **Пуск | Программы | BASIC-256 | BASIC-256**. Открывшееся окно программы (рис. 1.1) имеет обычный для приложений Windows вид. Заголовок окна имеет стандартную структуру. Меню программы содер-

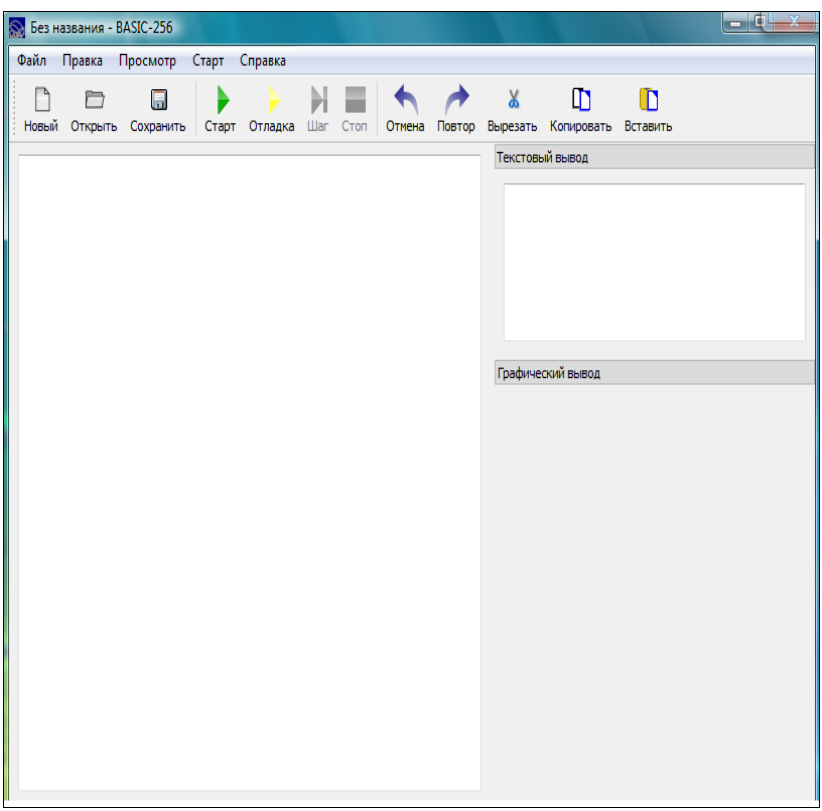


Рис. 1.1. Окно русских версий BASIC-256

жит пункты **Файл**, **Правка**, **Просмотр**, **Старт**, **Справка**. Смысл любого пункта интуитивно понятен при открытии его списка команд.

Расположенная под строкой меню панель инструментов содержит набор кнопок быстрого вызова наиболее употребительных команд: **Новый**, **Открыть**, **Сохранить**, **Старт**, **Отладка**, **Шаг**, **Стоп**, **Отмена**, **Повтор**, **Вырезать**, **Копировать**, **Вставить**. Кнопки формируют команды, позволяющие выполнять вызов программ из файлов, их сохранение, запуск, отладку и редактирование.

Рабочая область окна BASIC-256 выглядит несколько необычно по сравнению с более ранними версиями BASIC, такими как QBasic или Quick Basic. В формируемой по умолчанию рабочей области окна выделены три поля:

- слева находится основное поле — окно текста программы (окно редактора), управляемое редактором текста программы;
- справа в общем вертикальном столбце сформированы два поля (окна) исполнителя программы, управляемые отладчиком и средой исполнения:
  - верхнее поле называется **Текстовый вывод** и представляет собой экран исполнителя в текстовом режиме. На нем будут отображаться результаты вычислений и обработки данных, представляемые в виде текстовых (символьных) строк и выводимые чаще всего оператором `print`, а также сообщения об ошибках, формируемые самим интерпретатором на этапе синтаксического контроля программы пользователя;
  - нижнее поле называется **Графический вывод** и представляет собой графический экран исполнителя. На нем отображаются результаты выполнения графических операторов программы. По умолчанию графический экран имеет размеры 300×300 пикселов.

Характеристики текстового экрана (число строк, число символов в строке) зависят от типа и размера шрифта, используемого для вывода текстовой информации. Характеристики шрифта задаются специальными операторами BASIC-256 или командами меню.

Каждый из экранов очищается отдельной командой. Текстовый экран очищается обычной командой очистки текстового экрана `cls` (clear screen, очистить экран). Графический экран очищается специальной командой `clg` (clear graphic screen, очистить графический экран).

Окно текста программы присутствует на экране всегда. Экраны исполнителя (окно текста и окно графики) можно вызывать на экран или убирать с экрана командами меню **Просмотр | Окно текста** и **Просмотр | Окно графики**. По умолчанию эти функции включены — в соответствующем пункте меню стоит флажок ("галочка"). Одна подача команды включает функцию, повторная подача той же команды функцию отображения соответствующего окна на экране выключает. Отладчик позволяет при отладке программы вызвать на экран дополнительное окно для наблюдения за переменными программы — окно просмотра переменных. Оно вызывается на экран или убирается командой меню **Просмотр | Окно просмотра переменных**.

Размер шрифта, которым отображаются текст программы в окне редактора и результаты вычислений в окне текста, можно дискретно изменять командами меню **Просмотр | Размер шрифта | Мелкий / Средний / Большой / Огромный**. По умолчанию после загрузки BASIC-256 включен средний размер шрифта. Характеристики текстового экрана в режиме мелкого шрифта определяются экспериментально как 11 строк примерно по 40 символов `x`. Большинство шрифтов Windows не моноширинные, т. е. различные символы имеют различную ширину. Поэтому ширину строки в числе символов можно определить только приблизительно для некоторого символа со средней шириной. Если число выводимых строк превышает 9, в окне текста появляется вертикальная полоса прокрутки. Если выводимая строка превысит ширину окна текста, появится горизонтальная полоса прокрутки.

Запуск программы, текст которой подготовлен в окне редактора, осуществляется следующими способами: командой меню **Старт | Старт**; кнопкой **Старт** панели инструментов; клавишей <F5>.

Примечание

Клавиша <F5> используется для запуска приложений во многих реализациях BASIC.

1.3. Особенности BASIC-256 v.0.9.6p для Win32 и v.0.9.6.48 для ALT Linux 5.0.2 Школьный

Программа на языке BASIC-256 представляет собой последовательность строк кода. В каждой строке допускается один оператор или два оператора, разделенные двоеточием. Операторы в рассматриваемых версиях можно писать и строчными, и прописными латинскими буквами. То есть операторы `print x`, `PRINT x` и `Print x` будут выполняться одинаково. Функции `sin(x)`, `SIN(x)` и `Sin(x)` также будут вычисляться одинаково.

Имена переменных могут содержать только латинские буквы и цифры и должны начинаться с буквы. Причем, в именах переменных учитывается регистр: `x` и `X` — это разные переменные. Любые другие символы (символ подчеркивания, знаки препинания, знаки математических операций и другие специальные символы) в именах переменных недопустимы. Исключение составляет знак денежной единицы `$` — он (как во многих других версиях BASIC) ставится в конце имен переменных строкового типа.

В BASIC-256 используются только два типа данных: числа и символьные строки (строковые значения). Например, `z` — числовая переменная, а `z$` — строковая переменная (читается "z строковое"). Строковые значения и строковые константы пишутся в кавычках.

Оператор присваивания (так же, как и во всех версиях BASIC) — знак равенства `=`. Примеры записи операторов присваивания:

- `x = 5.47`
- `y$ = "Привет, BASIC-256"`

Оператор присваивания следует читать как "принять равным" или "присвоить значение". В правой части оператора присваивания можно записывать константу или выражение: `x = x + 5` (икс принять равным икс плюс пять).

Для чисел используется внутреннее представление в форме 32-битных двоичных чисел со знаком с плавающей запятой. В операторах и константах допускается запись чисел только в

естественной форме с десятичной точкой в качестве разделителя целой и дробной части: `2.345`; `-0.0076`.

> ***Внимание!***
>
> Запись чисел в экспоненциальной (полулогарифмической) форме в рассматриваемых версиях BASIC-256 не поддерживается.

Максимально представимые числа по абсолютной величине: `+/-1410065408`. Точность представления результатов при выводе на текстовый экран оператором `print` чисел N < 1 составляет 7 десятичных цифр, включая ноль целых с округлением до 6-го знака после запятой. Так `print 0.0000005` выведет на экран **0**, а `print 0.00000051` выведет **0.000001**. Другими словами, число |N1| <= 0.0000005 считается машинным нулем.

Из сложных структур данных в BASIC-256 определены только одномерные и двумерные массивы: числовые и строковые. Массивы описываются оператором `DIM`:

- `DIM Z(20)` — числовой массив из 20 элементов Z[0] — Z[19];
- `DIM V$(15)` — строковый массив из 15 элементов V$[0] — V$[14].

В общем случае оператор `DIM R(N)` при заданном `N` описывает `N` числовых переменных R[0] — R[N – 1], т. е. отводит для этих переменных необходимую область оперативной памяти.

Оператор `DIM U(N, M)` при заданных `N` и `M` описывает `N*M` числовых переменных U[0,0] — U[N – 1, M – 1]. Такой массив можно записать в виде прямоугольной таблицы из `N` строк по `M` чисел в каждой строке. Элемент U[i,j] при этом будет располагаться в строке i и иметь номер j слева: i = 0 — N – 1, j = 0 — M – 1.

Если размерность массива неизвестна, ее можно определить следующим образом: для одномерного массива `X(N)`: N = X[?]; для двумерного массива `U(N, M)`: N = U[?,], M = U[,?]. Каждый массив должен вводиться отдельным оператором `DIM`.

> ***Внимание!***
>
> В отличие от многих других версий, BASIC-256 не позволяет одним оператором `DIM` описать несколько массивов.

В BASIC-256 допустимы анонимные массивы — последовательности числовых или строковых констант в фигурных скобках, разделенных запятыми. Анонимные массивы можно использовать для присвоения значений обычным массивам:

```
dim Y(4): Y = {10, 20, 30, 40}
```

Приведенная последовательность операторов присвоит элементам массива Y[0] — Y[3] значения, перечисленные в списке анонимного массива: 10, 20, 30, 40. В общем случае, если в списке анонимного массива перечислены N значений, указанная последовательность операторов присвоит их элементам Y[0] — Y[N – 1].

Размерность числового или строкового массива можно переопределить в программе операторами вида `Redim x(n1)`, `Redim y$(n2)`, `Redim x1(n1, m1)`, `Redim y1$(n2, m2)`. При этом, если размерность массива увеличивается, добавленные элементы в числовом массиве инициализируются нулями, а в строковом массиве — пустыми строками. Если новая размерность массива меньше заданной ранее, "лишние" элементы теряются.

ГЛАВА 2

Основные операторы BASIC-256

2.1. Операторы обработки числовых данных

В отличие от многих других реализаций BASIC, оператор `print` в текущей версии BASIC-256 v.0.9.6p выводит на экран не список вывода, а результаты вычисления или обработки только одного выражения и имеет синтаксис `print expr`. Здесь `expr` — сокращение от expression — произвольное выражение, формирующее результат в числовой или строковой форме для вывода на текстовый экран (в текстовое окно) исполнителя. Следует заметить, что сам оператор `print` преобразует перед выводом на текстовый экран любой результат, в том числе числовой, в текстовую строку. Этот факт можно использовать для принудительного формирования списка вывода в составе `expr` в виде одной текстовой строки, как это делается в различных модификациях Visual Basic. Типовые приемы формирования списка вывода в составе `expr` в виде одной текстовой строки рассмотрены в *разд. 2.2*.

Оператор `input` обеспечивает диалоговый ввод данных в числовую или строковую переменную в следующих формах:

- `input expr, x1` — ввод числа с клавиатуры в переменную `x1`;
- `input expr, x2$` — ввод текстовой строки с клавиатуры в переменную `x2$`.

Здесь `expr` — необязательное строковое выражение, представляющее собой побуждающее сообщение для ввода. Например, первая строка приведенной ниже последовательности:

```
input "Введите целое x1 = ", x1
print x1
```

выведет на текстовый экран сообщение: **Введите целое x1 =**. Пользователь в той же строке вводит с клавиатуры нужное число и нажимает клавишу <Enter>. Переменной `x1` будет присвоено введенное число. Вторая строка выведет присвоенное `x1` значение в окно текста.

Над числовыми данными возможны следующие операции:

- сложение: `Y1 = X1 + X2`;
- вычитание: `Y2 = X1 - X2`;
- умножение: `Y3 = X1*X2`;
- деление: `Y4 = X1/X2`;
- возведение в степень: `Y5 = X1 ^ X2`;
- вычисление целого остатка `Y6` от деления целых чисел `X3` и `X4`: `Y6 = X3 % X4`;
- целочисленное деление `Y7 = X5\X6` — при этом вычисляется целая часть частного `Y7` от деления целых чисел `X5` и `X6`.

Например: команда `print 4.5^(-2.5)` выведет на экран результат вычисления: **0.026081**; команда `print 45 % 7` выведет остаток от деления 45 на 7, т. е. **3**; команда `print 45\7` выведет целую часть частного от деления 45 на 7, т. е. **6**.

Для числовых аргументов могут вычисляться следующие функции:

- `abs(x)` — вычисляет абсолютное значение выражения x, например: abs(–4.5) = 4.5;
- `x1 = ceil(x)` — вычисляет наименьшее целое x1 >= x, например: ceil(4.5) = 5; ceil(–4.5) = –4;
- `x2 = floor(x)` — вычисляет наибольшее целое x2 <= x, например: floor(4.5) = 4; floor(–4.5) = –5;

- `x3 = int(x)` — вычисляет целое число x3 путем отбрасывания дробной части числа x, например: int(4.5) = 4; int(–4.5) = –4;
- `pi`, `PI` — встроенная константа π = 3.141593;
- `cos(x)` вычисляет cos x;
- `sin(x)` вычисляет sin x;
- `tan(x)` вычисляет tg x;

> ***Внимание!***
>
> Для всех тригонометрических функций аргумент `x` задается в радианах. Аргумент `x1`, заданный в градусах, необходимо преобразовать в радианы с помощью переводного множителя 1° = pi/180, например: `y1 = sin(x1*pi/180)`. Можно также воспользоваться для преобразования аргумента `x1`, заданного в градусах, в радианы специальной функцией `radians(x1)`, например:
>
> `y1 = sin(radians(x1))`.

- функция `y = atan(x)` вычисляет arctg x, причем `y` получится в радианах.

  Если необходимо получить результат `y2` в градусах, следует использовать выражение вида `y2 = atan(x)*180/pi` или воспользоваться для преобразования угла из радианной меры в градусную функцией `degrees`: `y2 = degrees(atan(x))`.

  Аналогично для вычисления математических функций arcsin x, arcos x можно в BASIC-256 пользоваться функциями `asin(x)`, `acos(x)`;
- вычисление корней произвольной степени выполняется через операцию возведения в степень:
  - квадратный корень вычисляется так: `x1 = x^(1/2)`;
  - кубический корень: `x2 = x^(1/3)`;
  - корень произвольной степени n: `x3 = x^(1/n)`;
- функция `y1 = log(x)` вычисляет натуральный (Неперов) логарифм ln x по основанию e = 2.73: log(10) = 2.302585;
- функция `y2 = log10(x)` вычисляет десятичный логарифм lg x: log10(1000) = 3;

- логарифм числа x по произвольному основанию a вида y3 = $\log_a x$ можно вычислить, используя стандартное математическое соотношение $\log_a x = \ln x / \ln a$. В BASIC-256 для этого придется записать одно из выражений: `y3 = log(x)/log(a)` или `y3 = log10(x)/log10(a)`.

> ***Примечание***
>
> Следует заметить, что имеющаяся во всех реализациях BASIC функция `exp(x) = e^x`, обратная натуральному логарифму, в BASIC-256 отсутствует. При необходимости ее использования в программе можно ввести константу `e = 2.718282` и использовать функцию вида `e^x` вместо `exp(x)`.

Оператор `rand` формирует псевдослучайное число x в диапазоне 0 <= x < 1, распределенное по закону равной плотности. В отличие от других реализаций BASIC инициализация генератора случайных чисел (функции `rand`) в BASIC-256 выполняется не аргументом, а автоматически при каждой очередной загрузке. После каждой новой загрузки BASIC-256 будет формироваться новая последовательность случайных чисел от 0 до 0.999999. Если необходимо сформировать случайную последовательность чисел в другом диапазоне для моделирования некоторой последовательности событий, можно воспользоваться правилом: любое выражение, содержащее функцию формирования случайной последовательности `rand`, дает в результате случайную последовательность. Формирование выражений можно пояснить следующими примерами:

- бросание монеты — орлу ставится в соответствие 1, решке — 0. Случайная последовательность равновероятных 1 и 0 формируется с помощью выражения: `x = int(2*rand);`
- бросание кубика — формирование случайной последовательности целых чисел `x1` в диапазоне от 1 до 6 реализуется выражением:

```
x1 = 1 + int(6*rand);                                    (2.1)
```

- вытаскивание карты из хорошо перемешанной колоды из 36 карт — если поставить в соответствие картам от валета до туза числа от 11 до 14, то без учета масти карт процесс моде-

лируется формированием последовательности случайных целых чисел `x2` в диапазоне от 6 до 14 согласно выражению:

```
x2 = 6 + int(9*rand).
```
(2.2)

В общем случае если некоторая последовательность событий моделируется последовательностью случайных целых чисел `x3` в диапазоне от `N1` до `N2`, то можно использовать выражение вида:

```
x3 = N1 + int((N2 - N1 +1)*rand).
```
(2.3)

Оператор `pause x` — задержка на `x` секунд. Здесь `x` — десятичное число. Так, команда `pause 0.15` задерживает исполнение программы на 0.15 секунды. Этот оператор полезен при формировании подвижных изображений в графическом или текстовом окнах или при отображении результата работы программы автоматически сменяющейся с определенной задержкой последовательностью кадров, т. е. в виде слайд-фильма.

Оператор `Sound` — воспроизводит звук или последовательность звуков заданной частоты `F` и длительности `T` через системный динамик. Частота задается в герцах (Гц), длительность в миллисекундах (мс): 1 мс = 0.001 с. Команда `Sound` имеет следующие форматы:

- `Sound F1, T1` или `Sound(F1, T1)` — воспроизводит звук частоты `F1` Гц длительностью `T1` мс, например, команда `Sound(1000, 5000)` воспроизведет звук частоты 1000 Гц длительностью 5 сек. Паузу можно задать частотой 0 и нужной длительностью;
- `Sound Y`, `Sound(Y)`, `Sound {F1, T1, F2,T2, ... , Fn, Tn}` — аргументом команды может быть массив `Y` из `n` пар чисел `Fi`, `Ti`, в том числе анонимный массив.

> ***Примечание***
>
> При некотором навыке в этой форме можно записывать мелодии (табл. 2.1). Слышимость звуков зависит от полосы пропускания системного динамика. Звуки очень низких и очень высоких звуковых частот (ниже 50 Гц, выше 7–8 кГц) могут не воспроизводиться системным динамиком и будут не слышны.

В табл. 2.1 приведены латинские и русские обозначения нот и соответствующие им частоты в Гц для двух октав. Ноты проме-

Таблица 2.1. Таблица соответствия частот и нот [7]

	A_{Sharp} / B_{Flat}	B	C	C_{Sharp} / D_{Flat}	D	D_{Sharp} / E_{Flat}	E	F	F_{Sharp} / G_{Flat}	G	G_{Sharp} / A_{Flat}
ЛЯ		СИ	ДО		РЕ		МИ	ФА		СОЛЬ	
440	466	493	523	554	587	622	659	698	740	784	
220	233	247	262	277	294	311	330	349	370	392	415
								175	185	196	208

жуточных частот обозначены следующим образом: C_{Sharp} это ДО-диез; D_{Flat} это РЕ-бемоль. Длительность звучания нот определяется темпом исполнения — числом ударов метронома в минуту. Каждый удар — четверть ноты. Темп 120 — это 120 ударов метронома в минуту. Целая нота — 2 сек. Половинная нота — 1 сек. Четверть ноты — 0,5 сек.

Оператор `Volume X` или `Volume(X)` — задает громкость звука, воспроизводимого оператором `Sound`. Уровень громкости задается целым числом `X` от 0 до 10. По умолчанию `X = 5`.

Оператор `System Z$` или `System(Z$)` — позволяет из программы в среде BASIC-256 выполнить системную команду (команду операционной системы), заданную текстовой строкой `Z$`, в окне терминала. Это аналог команды Windows **Пуск | Выполнить**. Применение такой команды требует большой осторожности и высокой квалификации пользователя.

2.2. Операторы обработки строковых данных

Оператор `Asc(x$)` — код первого символа строки `x$` или символа `x$`: Asc(A) = 65; asc(blue) = 98 — код символа b.

Оператор `Chr(N)` — формирует символ по коду `N`: chr(90) = Z; chr(123) = {.

Символы стандарта ASCII с кодами от 32 до 126 приведены в табл. 2.2.

Таблица 2.2. Кодирование символов стандарта ASCII (коды от 32 до 126)

Код	Символ	Код	Символ	Код	Символ	Код	Символ	Код	Символ	Код	Символ	Код	Символ	Код	Символ
32	Пробел	44	,	56	8	68	D	80	P	92	\	104	h	116	t
33	!	45	-	57	9	69	E	81	Q	93	]	105	i	117	u
34	"	46	.	58	:	70	F	82	R	94	^	106	j	118	v
35	#	47	/	59	;	71	G	83	S	95	_	107	k	119	w
36	$	48	0	60	<	72	H	84	T	96	`	108	l	120	x
37	%	49	1	61	=	73	I	85	U	97	a	109	m	121	y
38	&	50	2	62	>	74	J	86	V	98	b	110	n	122	z
39	‘	51	3	63	?	75	K	87	W	99	c	111	o	123	{
40	)	52	4	64	@	76	L	88	X	100	d	112	p	124	|
41	(	53	5	65	A	77	M	89	Y	101	e	113	q	125	}
42	*	54	6	66	B	78	N	90	Z	102	f	114	r	126	~
43	+	55	7	67	C	79	O	91	[	103	g	115	s	127	

Следует обратить внимание на два похожих при печати символа:

- chr(39) = ’ — апостроф;
- chr(96) = ` — символ слабого ударения.

Символ ПУСТО (пустая строка) отображается командой chr(0).

Коды от 1 до 31 таблицы символов ASCII отведены для неотображаемых управляющих символов. Например, команды chr(10), chr(13) при выводе в текстовое окно оператором `print` обеспечивают переход в начало следующей строки.

Знак `+` для строковых переменных и значений выполняется как операция конкатенации (объединения) текстовых строк. Строка программы вида:

```
x$ = x1$ + x2$ + x3$
```
(2.4)

сформирует объединенную текстовую строку `x$`, состоящую из трех составляющих текстовых строк `x1$`, `x2$`, `x3$`, записанных последовательно в порядке их следования в выражении. В выражении вида (2.4) могут использоваться и числовые переменные. Они автоматически будут преобразованы в строковое представление. Выражение вида:

```
y$ = "x1 = " + x1 + " " + "x2 = " + x2,
```
(2.5)

где обрабатываются одновременно числовые и строковые значения, сформирует результирующую текстовую строку `y$`, в которую числовые значения `x1` и `x2` войдут также в форме текстовых строк. Выражение (2.5) позволяет в операторе `print` выводить в одной строке значения нескольких переменных, если объединить их в общую текстовую строку, добавив в нее необходимые разделительные пробелы и знаки препинания. Для примера — последовательность строк программы (листинг 2.1)

Листинг 2.1

```
x1 = 25
x2 = -48
print "x1 = " + x1 + ", " + "x2 = " + x2
```

выведет в текстовое окно следующий результат в виде текстовой строки: **x1 = 25, x2 = –48**.