

Основы программирования на языке «BASIC»



Если вы хотите:

Общаться с ЭВМ на равных;
Достичь успеха в учебе и труде;
Выучить язык всего из 15 слов;
Просто отдохнуть и развлечься

То учите Бейсик!

Для
продолжения
нажмите
кнопку



Знакомство с программой

ЭВМ становится более доступной любому пользователю и на работе, и дома, раскрывает огромные возможности перед всеми людьми.

Язык программирования «Basic» существует в 52 вариантах. Он становится международным языком программирования. В данной презентации Вы познакомитесь с 15 основными операторами, которые позволят Вам писать программы.

В ходе работы Вам будут попадаться кнопки управления



данная кнопка позволит Вам вернуться к предыдущему слайду



эта кнопка позволит перейти к следующему слайду



эта кнопка вернет Вас к содержанию

В содержании нужно щелкнуть по интересующему Вас разделу и Вы переместитесь к нему

Желаем удачи в освоении данного курса!



Содержание



1. Оператор LET (присваивание)
2. Оператор PRINT
3. Оператор INPUT
4. Пример расчета выражений
5. Некоторые функции
6. Примеры расчетов с помощью функций
7. Примеры расчетов с помощью функций
8. Графические операторы
9. Графические операторы
10. ВЕТВЛЕНИЕ
11. Оператор безусловного перехода
12. Программа для нахождения корней квадратного уравнения
13. Оператор цикла
14. Тираж спортлото
15. Вычисление суммы простого ряда
16. Вычисление суммы ряда состоящее из дробных чисел
17. Вычисление суммы ряда с чередующимися знаками
18. Использование циклов в графике
19. Построение графиков функций
20. Построение прямой
21. Построение параболы
22. Построение тригонометрических функций
23. Массивы
24. Массивы
25. Массивы
26. Массивы
27. Виды массивов



1. Оператор LET (присваивание)



Оператор присваивания присваивает переменным некое значение.

Например: 10 LET A=7.2

То есть присвоить переменной А значение 7.2

Пример расчета арифметических выражений:

$$y=(a-bc)/(a+b)$$

Данное выражение записывается в виде строки:

30 LET Y=(A-B*C)/(A+B)

Или можно разбить на несколько строк:

40 LET Y1=A-B*C

50 LET Y=Y1/(A+B)

$$y=a^3-b^3*c^3$$

Данное выражение записывается в виде:

60 LET Y=A^3-B^2*C^5

Или

70 LET B=A^(1/3)

$$b = \sqrt[3]{a}$$

80 LET B=A^(-3)

$$b = \frac{1}{a^3}$$

90 LET B=A^(X-2*Y)

$$b = a^{x-2y}$$



2. Оператор PRINT

Print – означает операцию вывода информации на экран из памяти ЭВМ.

Например:

```
100 PRINT «Мяу-мяу»
```

```
110 END
```

Для запуска программы на исполнение набираем RUN и нажимаем ENTER. На экране появится Мяу-мяу, а строкой ниже ОК (или READY). Последним словом компьютер сообщает «Программа выполнена, жду дальнейших указаний».

С помощью данного оператора выводятся не только константы, но и арифметические выражения.

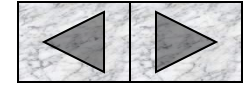
Например подсчитаем объем пирамиды с квадратным основанием:

```
120 PRINT «Объем равен», A*A*H/3, однако до строки 120 необходимо определить значение A и H. Поэтому многие предпочитают следующий тип записи:
```

```
115 LET Y=F*F*H/3
```

```
120 PRINT «Объем равен», Y (или PRINT «Y=», Y).
```

Внимание: все, что стоит под оператором PRINT в кавычках, отображается на экране в неизменном виде. Если под этим оператором стоит переменная без кавычек, то на экране отобразится ее значение.



3. Оператор INPUT

INPUT – производит операцию ввода информации с клавиатуры в оперативную память ЭВМ. За данным оператором следует список ввода. Список содержит идентификаторы, значения которых должны быть введены (элементы списка разделяются запятыми).

Например:

130 INPUT K

Если данную программу запустить на исполнение, то на экране появится знак вопроса. Пользователь должен пробить с клавиатуры число и нажать ENTER. Можно вводить сразу несколько чисел.

140 INPUT K1, K2, K3, K4 как всегда выдаст знак вопроса, а после ввода всех четырех чисел присвоит их переменным в том порядке, в каком они записаны в списке. Если же список еще не исчерпан, а клавиша ввода нажата после третьего числа, то машина поставит знак вопроса и будет ждать очередного числа.

Программисты советуют перед строкой INPUT выдавать сообщение – подсказку человеку о том, каких данных ждет от него машина:

150 PRINT «Введите коэффициент усиления трансформатора»

160 INPUT K

Или

170 PRINT «Введите дату в виде ЧЧ, ММ, ГГ»

180 INPUT B1, B2, B3.



4. Пример расчета выражений



Попробуем составить программу для вычисления по формуле:

$$y = \frac{x^3 + 2x^2 + 3x + 4}{4x^4 + 3x^3 + 2x^2 + x + 10}$$

Итак, напомним:

```
190 PRINT «Введите значение X»
```

```
200 INPUT X
```

Второй блок лучше записать в три строки:

```
210 LET Y1=X^3+2*X^2+3*X+4
```

```
220 LET Y2=4*X^4+3*X^3+2*X^2+X+10
```

```
230 LET Y=Y1/Y2
```

И, наконец, третий блок:

```
240 PRINT «Результат расчетов»
```

```
250 PRINT «X=», X
```

```
260 PRINT «Y=», Y
```

```
270 END
```



5. Некоторые функции



Функции играют важную роль в программировании.

В некоторых версиях Бейсика список функций может быть больше.

Вот основные из них:

- SIN (X) - СИНУС
- COS (X) - КОСИНУС
- TAN (X) - ТАНГЕНС
- ANT (X) - АРКТАНГЕНС
- EXP (X) - ЭКСПОНЕНТА
- Log (x) – НАТ. ЛОГАРИФМ
- SQR (X) – КОРЕНЬ КВАДРАТНЫЙ
- ABS (X) – АБСОЛЮТНАЯ ВЕЛИЧИНА
- INT (X) – ЦЕЛАЯ ЧАСТЬ
- SGN (X) – ЗНАКОВАЯ ФУНКЦИЯ
- RND (X) – ГЕНЕРАТОР СЛУЧАЙНЫХ ЧИСЕЛ
- TAB (X) - ТАБУЛЯЦИЯ



6. Примеры расчетов с помощью функций



Функция SQR(X)

```
350 LET A1=SQR(5)
360 PRINT «Корень из 5 =», A1
Или
370 PRINT «Введите число»
380 INPUT X
390 LET A1=SQR(X)
400 PRINT «Корень их»; X; «равен»; A1
```

Функция ABS(X)

```
410 LET Y=ABS(X)
```

Присваивает Y значение X в том случае, если число X положительное, и - X, если число отрицательное.

ABS (X)=|X| - будет соответствовать модулю

Функция SGN(X)

Знаковая функция которая принимает значения: Y=1 при X>0, Y=0 при X=0, Y=-1 при X<0.

```
420 LET Y=SGN(X)
```

Такая функция используется в массивах.

Функция TAB(X)

Позволяет выводить информацию не с первой позиции, а с заданной.

Например **430 PRINT TAB (15); «УРА»**

-----УРА



7. Примеры расчетов с помощью функций

Функция INT – находит целую часть от числа X. Например, если $X=4,31$, то:

```
440 LET Y1=INT(X)
```

```
450 LET Y2=INT(-X)
```

```
460 LET Y3=INT(X*X)
```

```
470 LET Y4=INT(-X*X)
```

```
480 PRINT Y1, Y2, Y3, Y4
```

Даст результат $Y1=4$, $Y2=-5$, $Y3=18$, $Y4=-19$.

С помощью данной функции можно округлять дробные числа:

```
490 LET X=INT(X+.5)
```

Функция RND (X)- генератор случайных чисел

```
500 LET T = RND(1)
```

```
510 PRINT T
```

Дадим команду на исполнение и появится число в промежутке от 0 до 1.

Для изменения интервала:

```
520 PRINT INT(RND(1)*10)
```

Эта строка выведет случайное число в интервале от 0 до 9.



8. Графические операторы

Необходимо помнить, что изображение на экране не сплошное, как может показаться на первый взгляд, а состоит из точек. Стандартное количество точек 800x600. Для начала работы с графикой представляют ось координат с нулем в верхней левой точке экрана. Ось абсцисс направлена слева на право, а ось ординат - сверху вниз.

Для очистки экрана используем команду **CLS** и нажимаем **ENTER**.

Оператор PSET (x,y) – ставит точку на экране. Для этого пишем:

530 PSET(20,40) и запускаем на исполнение. На экране появится точка. Введем еще одну строку:

540 PSET(25,40) – появится еще одна точка.

Оператор LINE(x1,y1)-(x2,y2) – строит прямые линии.

Например: **550 LINE(20,40)-(100,40)**

Можно построить прямоугольник. Для этого:

560 LINE(20,20)-(120,120)

570 LINE(120,20)-(120,120)

580 LINE(120,120)-(20,120)

590 LINE(20,120)-(20,20)

Но этот же квадрат можно построить так: **560 LINE(20,20)-(120,120), 2, B**, где «2» обозначает цвет (в данном случае – зеленый), а «B» - собственно построение квадрата.



9. Графические операторы

Оператор CIRCLE(x12,y1),r – позволяет строить окружности.

Например:

580 CIRCLE(100,100),50,2

Запустив данную команду на исполнение командой RUN мы увидим окружность радиусом в 50 точек и зеленой линией

Используя комбинации окружности и линий можно рисовать простейшие рисунки. Например:

590 CIRCLE(50,50),10,2

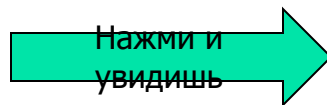
600 LINE(50,55)-(50,95),2

610 LINE(50,95)-(60,125),2

620 LINE(50,95)-(40,125),2

630 LINE(50,65)-(30,80),2

640 LINE(50,65)-(70,80),2



Для того, чтобы начертить часть окружности в оператор необходимо включить параметры, определяющие соответственно начальный и конечный углы, в пределах которых вычерчивается дуга.

Числа, указывающие значение углов (в радианах) указываются после параметра определяющего цвет.

Для пересчета в радианы воспользуйтесь формулой $X2 = X1 * 3,14159 / 180$

Например: **CIRCLE(80,100),50,2,1.5708,3.1415**

Появится дуга.

Можно превратить круг в эллипс. Для этого введем коэффициент сжатия от 1/260 до 260. Если какие-то параметры отсутствуют, то их место необходимо отметить запятой. Например:

CIRCLE(80,100),50,,,2

Будет построена окружность цвет которой совпадает с уже имеющимися фигурами и коэффициентом сжатия – 2.



10. Ветвление

В различных программах зачастую приходится сталкиваться с разветвлением условия. Для их решения нам понадобится оператор условного перехода

Например:

$$\begin{cases} \acute{o} = 2\tilde{o}^4 + 4\tilde{o}^2 - 10 & \text{ïðå } \tilde{o} > 0 \\ \acute{o} = 3\tilde{o}^3 - 6\tilde{o} & \text{ïðå } \tilde{o} < 0 \end{cases}$$

Оператор условного перехода:

Оператор состоит из трех составляющих:

IF(условие1) **THEN**(условие2)
ELSE(условие3)

IF-если, THEN- тогда ELSE – иначе.

То есть ЕСЛИ выполняется условие 1,
ТОГДА выполняется условие 2, ИНАЧЕ
выполняется условие3.

Вернемся к решению примера:

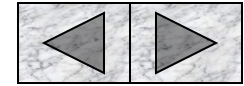
```
650 PRINT «Введите x»  
660 INPUT X  
670 IF X>0 THEN Y=2*X^4+4*X^2--10 ELSE  
      Y=3*X^X-6*X  
680 PRINT «Y=»,Y
```

Или так:

```
690 PRINT «Введите x»  
700 INPUT X  
710 IF X>0 THEN Y=2*X^4+4*X^2-10  
720 IF X>0 THEN Y=3*X^X-6*X  
730 PRINT «Y=»,Y
```

В данном случае не используется **ELSE**, что увеличивает возможности оператора условного перехода.

11. Оператор безусловного перехода



GOTO – оператор безусловного перехода, в отличие от оператора условного перехода не имеет никаких условий.

С его помощью можно переходить со строки на строку, перепрыгивать в начало, середину или конец документа.

Например:

```
800 PRINT «Мяу-мяу»
```

```
810 GOTO 800
```

В данном случае компьютер напечатает на экране «Мяу-мяу» перейдет к строке 810 которая пошлет его на строку 800 и программа зациклится.

Для конкретного определения цикла можно использовать:

```
820 LET M=0
```

```
830 LET M=M+1
```

```
840 IF M=20 THEN GOTO 870
```

```
850 PRINT «Мяу-мяу»
```

```
860 GOTO 830
```

```
870 END
```

В данном случае мы заставили компьютер повторить цикл 20 раз.



12. Программа для нахождения корней квадратного уравнения

Для решения данной задачи нам надо вспомнить квадратное уравнение:

$$ax^2+bx+c=0$$

Вводим:

```
900 PRINT «Введите a, b, c»
```

```
910 INPUT A, B, C
```

Теперь вспомним, чему равен дискриминант квадратного уравнения:

$D=b^2-4ac$, на Бейсике это записывается так:

```
920 D=B^2-4*A*C
```

Необходимо вспомнить, что если D – отрицательное число, то уравнение не имеет решения, если $D=0$, то уравнение имеет одно решения, и иначе у него два корня.

```
930 IF D<0 THEN GOTO 980
```

```
940 X1=-B+(SQR(D))/(2*A)
```

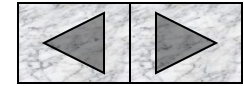
```
950 X2=-B+(-SQR(D))/(2*A)
```

```
960 PRINT "X1=", X1,"X2=",X2
```

```
970GOTO 990
```

```
980 PRINT «Уравнение корней не имеет»
```

```
990END
```



13. Оператор цикла

С помощью оператора цикла организуется многократное выполнение некоторых операций. В Бейсике используют следующие операторы:

FOR...TO...STEP...NEXT

Операторы FOR и NEXT используются всегда вместе. Вот как будет выглядеть наш цикл «мяукающей» программы:

```
1000 FOR K=1 TO 20
1010 PRINT «Мяу-мяу»
1020 NEXT K
```

FOR означает ДЛЯ, **TO** означает ДО. Цикл будет повторяться до тех пор, пока не будет выполнено условие $K=20$.

Параметр цикла можно задать по разному: A, C, X.

Пример: необходимо найти значение функции $y=2x-3x-7$ при x от 1 до 30.

```
1030 FOR X=1 TO 30
1040 LET Y=2*X-3*X-7
1050 PRINT X, Y
1060 NEXT X
```

Выполнив программу получим данные.

Если нужно поменять шаг, то:

```
1030 FOR X=-10 TO 20
```

Оператор **STEP** – это шаг. Переменная может изменяться не только на 1, но и на 0,1, 1000 и т.д. Для этого вводим шаг

```
1030 FOR X=-10 TO 20 STEP .1
```

Этот шаг будет находить значения с шагом 0,1



14. Тираж спортлото

Для случайного вывода нескольких чисел вспоминаем ранее изученные функции:

```
1070 FOR K=1 TO 20  
1080 LET T=INT (RND(1)*100)  
1090 PRINT T;  
1100 NEXT K
```

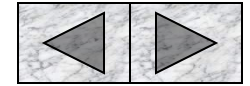
Выборка при запуске программы будет производиться в интервале от 0 до 99 в количестве 20 чисел.

Если хотим изменить параметры, например 5 из 36, то:

```
1110 FOR K=1 TO 5  
1120 LET T=INT (RND(1)*37)  
1130 PRINT «Выпало число», T;  
1100 NEXT K
```

Вот и получился примитивный лототрон

15. Вычисление суммы простого ряда



Если нам надо вычислить сумму ряда $1+2+3+4+5...+20$, то можно использовать оператор цикла:

```
1150 FOR K=1 TO 20
```

```
1160 S=S+I
```

```
1170 NEXT I
```

```
1180 PRINT «S=», S
```

Если ряд будет заканчиваться любым числом, то:

```
1190 INPUT N
```

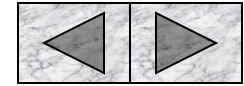
```
1200 FOR K=1 TO N
```

```
1210 S=S+I
```

```
1220 NEXT I
```

```
1230 PRINT «S=», S
```

В данной программе можно поменять знаки на «+», «-», «/».



16. Вычисление суммы ряда состоящее из дробных чисел



Допустим наш ряд это:

$$\frac{1}{2} + \frac{1}{4} + \frac{1}{6} + \dots + \frac{1}{N}$$

Обратите внимание, что здесь
знаменатель меняется с шагом 2.

```
1240 INPUT N
```

```
1250 FOR I=1 TO N STEP 2
```

```
1260 S=S+1/I
```

```
1270 NEXT I
```

```
1280 PRINT «S=», S
```

Попробуем усложнить задание:

Допустим наш ряд это: $\frac{1}{3} * \frac{4}{6} * \frac{7}{9} * \dots * \frac{19}{21}$

Обратите внимание, что здесь

зависимость между числителем и
знаменателем выражается в

формуле

$$\frac{i}{i+2}$$

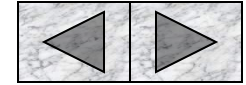
Т.е. Числитель отличается от
знаменателя на 2 и изменяется с
шагом 3.

```
1290 S=1
```

```
1300 FOR I=1 TO 19 STEP 3
```

```
1310 S=S+I/(I+2)
```

```
1320 NEXT I
```



17. Вычисление суммы ряда с чередующимися знаками



Программа для вычисления суммы ряда у которого знаки чередуются, изменяются с «+» на «-» и наоборот не многим отличается от предыдущей. Допустим ряд имеет вид:

$$\frac{2}{5} - \frac{4}{9} + \frac{6}{13} - \frac{8}{17} + \dots - \frac{20}{41}$$

Для изменения знака необходимо умножить число на -1 и знак «-» изменится на «+» и наоборот. Выведем формулу, насчитывающую любое слагаемое ряда. В данном случае это:

$$\frac{i}{2i+1}$$

```
1330 K=1
```

```
1340 FOR I=1 TO 20 STEP 2
```

```
1350 S=S+I/(2*I+1)*K
```

```
1360 K=K*(-1)
```

```
1370 NEXT I
```

Благодаря строке 1360, знак действия будет меняться с каждым витком цикла.



18. Использование циклов в графике



Построение линий

Если необходимо отобразить линию, но без оператора LINE, то можно использовать оператор цикла:

```
1700 FOR K=1 TO 100
```

```
1710 PSET (K,20)
```

```
1720 NEXT K
```

Получается отрезок состоящий из 100 точек.

Если хотим получить пунктирную линию, то:

```
1730 FOR K=1 TO 100 STEP 2
```

```
1740 PSET (K,20)
```

```
1750 NEXT K
```



19. Построение графиков функций



Построение координатной плоскости

Отображенная нами координатная плоскость может быть использована для построения любого графика функции, изменяются будут только значения на осях ординат и абсцисс.

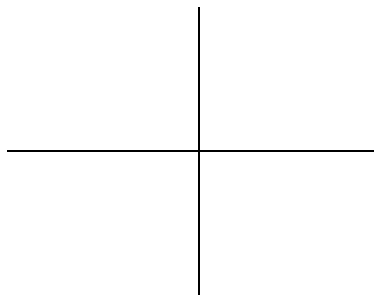
Первоначально строим оси:

```
1800 CLS
```

```
1810 LINE (110,10)-(110,210),15
```

```
1820 LINE (10,110)-(210,110),15
```

Получится



Теперь этот крест разобьем на части с помощью цикла:

```
1830 FOR X=10 TO 210 STEP 10
```

```
1840 LINE (X,108)-(X,112),15
```

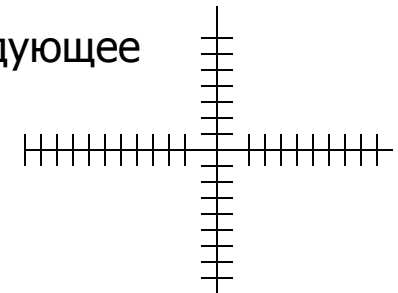
Тем самым мы разобьем ось X на отрезки. Для оси Y делаем тот же цикл, но LINE (108,Y)-(112,Y),15

Для упрощения цикла можно поменять переменные местами:

```
1850 LINE (108,X)-(112,X),15
```

```
1860 NEXT X
```

Получится следующее





20. Построение прямой



После построения координатной плоскости попробуем отобразить на ней графическое отображение уравнения прямой.

$Y=4x-5$, при x от 0 до 10.

Для построения графика воспользуемся схемой прописанной в построении графиков функций.

```
1870 FOR X=0 TO 10
```

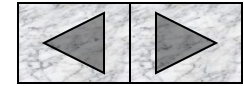
```
1880 Y=4*X-5
```

Необходимо привязать начало координат к точке (110,110):

```
1890 PSET (X+110,Y+110),15
```

```
1900 NEXT X
```

ЭВМ построит график функции с центром оси координат именно там, где нам хотелось. Для увеличения количества точек в прямой, можно вставить шаг 0,2-0,5.



21. Построение параболы

Для построения параболы мы используем тот же принцип, что и для построения прямой. Самая распространенная квадратичная парабола имеет вид:

$$y=x^2$$

Составим программу для построения графика этой функции на промежутке от -50 до 50:

```
1910 FOR X=-50 TO 50 STEP .1  
1920 Y=X^2  
1930 PSET (X+110,Y+110),15  
1940 NEXT X
```

Получится так

Нажми

Обратите внимание, что оси параболы слишком прижаты к оси y . Это и будет, т.к. координатная плоскость разбита на отрезки, состоящие из 10 точек. Исходя из этого, для отображения параболы, имеющий более привычный вид, необходимо уменьшить значение функции в 20 раз. Поэтому построим параболу на промежутке x от -50 до 50 для функции:

$$y = \frac{x^2}{20}$$

```
1910 FOR X=-50 TO 50 STEP .1  
1920 Y=X^2/20  
1930 PSET (X+110,Y+110),15  
1940 NEXT X
```

Получим

Нажми



22. Построение тригонометрических функций



Для построения тригонометрических функций мы будем использовать ту же координатную плоскость. Построим график функции $y=\sin(x)$

На нашей координатной плоскости:

```
1950 FOR X=-100 TO 100 STEP .1  
1960 Y=SIN(X)  
1970 PSET (X+110,Y+110),15  
1980 NEXT X
```

Получится изображение:

Нажми

В силу того, что у нас небольшое количество точек, необходимо увеличить амплитуду колебания в 50 раз. Т.е. Построить график функции

$$y = 50 \sin\left(\frac{x}{10}\right)$$

Но при этом обозначить значение верхней точки параболы на координатной плоскости как 1.

```
1950 FOR X=-100 TO 100 STEP .1  
1960 Y=50*SIN(X/10)  
1970 PSET (X+110,Y+110),15  
1980 NEXT X
```

Получится:

Нажми



23. Массивы

При составлении программ на Бейсике возникают некоторые трудности. Одна из них, это обозначение переменных, близких по смыслу. В математике для этого вводят индексы $k_1, k_2, k_3, \dots, k_{12}$. В Бейсике нет индексов в привычной нам форме. Но зато разрешается вместо k_1 записать $K(1)$, т. е. указывать индексы в той же строке, но в скобках. Чтобы ЭВМ могла бы выделить для индексированных переменных место в памяти, и чтобы одни переменные не занимали место других, нужно специально указать в программе название этой переменной и максимально возможное значение индекса. Это делается с помощью оператора DIM следующим образом:

1400 DIM K(12)



24. Массивы

Новый для нас оператор DIM (от DIMENSION - размерность, объем, протяженность) содержит указание ЭВМ выделить, зарезервировать, отвести в оперативной памяти место под индексированную переменную, и тогда в памяти выделяется совокупность ячеек длиной 12 или, что то же самое, массив K из 12 элементов. (Обратите внимание: здесь используется новое понятие - массив, очень употребляемый в программировании термин.) Число 12 в строке 1400 означает, что в программе можно использовать не более 12 индексированных переменных под именем K - величин $K(1), \dots, K(12)$. И после того как эти переменные узаконены, с элементами массива можно обращаться, как с обычными переменными без индексов - использовать в арифметических выражениях, засылать в них числа, выводить содержимое на экран, принтер и т. д.

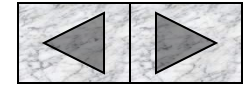


25. Массивы

Допустим, что вам необходимо вывести на экран таблицу, и в ячейках этой таблицы должны быть определенные данные. Таблицу эту следует разместить в массиве, состоящем из столькох строк, сколько вам необходимо. Такой массив, состоящий из нескольких строк и столбцов, называют двумерным массивом (а введенный ранее, в строке 1400 - одномерным). Информацию о размере двумерного массива надо принимать во внимание при работе. Она записывается также с помощью оператора размерности:

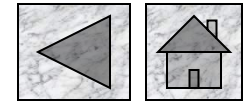
1500 DIM B(4,10)

В таком случае, число строк указывается первым, а число столбцов (длина строк - размер по горизонтали) - вторым (всегда!) и отделяется запятой.



26. Массивы

Введя в программу строку описания массива, мы можем работать с отдельными его элементами, которые обозначаются, например, так: $V(2,5)$ или $V(1,10)$. Это означает, что здесь упоминаются два элемента: первый $V(2,5)$ находится во 2-ой строке и 5-ом столбце, а второй $V(1,10)$ - в 1-ой строке и 10-м столбце. С этими элементами также можно производить все операции, которые осуществимы для обычных переменных. Очень важно не путать номер строки и номер столбца. Номер строки записывается всегда первым, а номер столбца следует за ним и отделяется запятой. И еще. Числа в скобках в операторе DIM и в обозначении элемента массива имеют разное значение. В первом случае - это наибольшее, максимально возможное количество элементов в строке и столбце, если эти же числа (кстати, они всегда целые и положительные) встречаются в идентификаторе переменной, то означают конкретное значение - номер строки и номер столбца, на пересечении которых в двумерной таблице находится данный элемент.



27. Виды массивов

Одномерные массивы

А теперь составим программу для отображения сначала одномерного массива:

```
1510 DIM M(10)
1520 FOR I=1 TO 10
1530 M(I)=INT(RND(1)*10)
1540 PRINT M(I)
1550 NEXT I
```

Программа выводит одномерный массив, состоящий из случайных чисел в интервале от 0 до 9.

Двумерные массивы

Для составления двумерного массива используем два цикла: для столбцов и для строк:

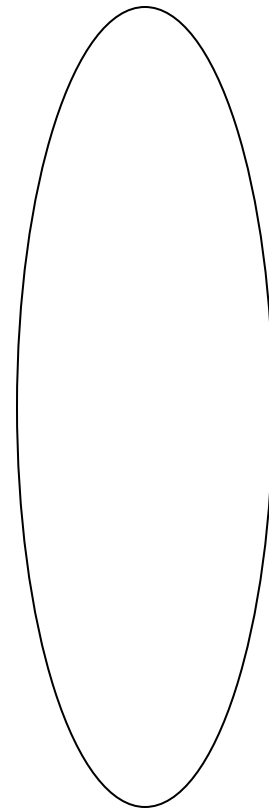
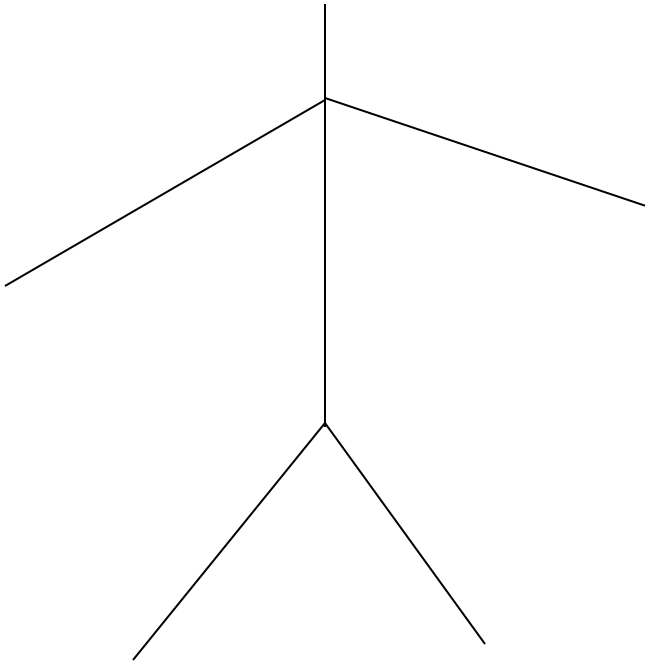
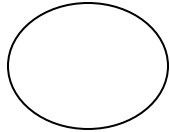
```
1560 DIM A(6,6)
1570 FOR I=1 TO 6
1580 FOR J=1 TO 6
1590 A(I,J)=0
1600 PRINT A(I,J)
```

Теперь закрываем циклы. Сначала по правилам закрывается внутренний цикл (J), а затем внешний (I):

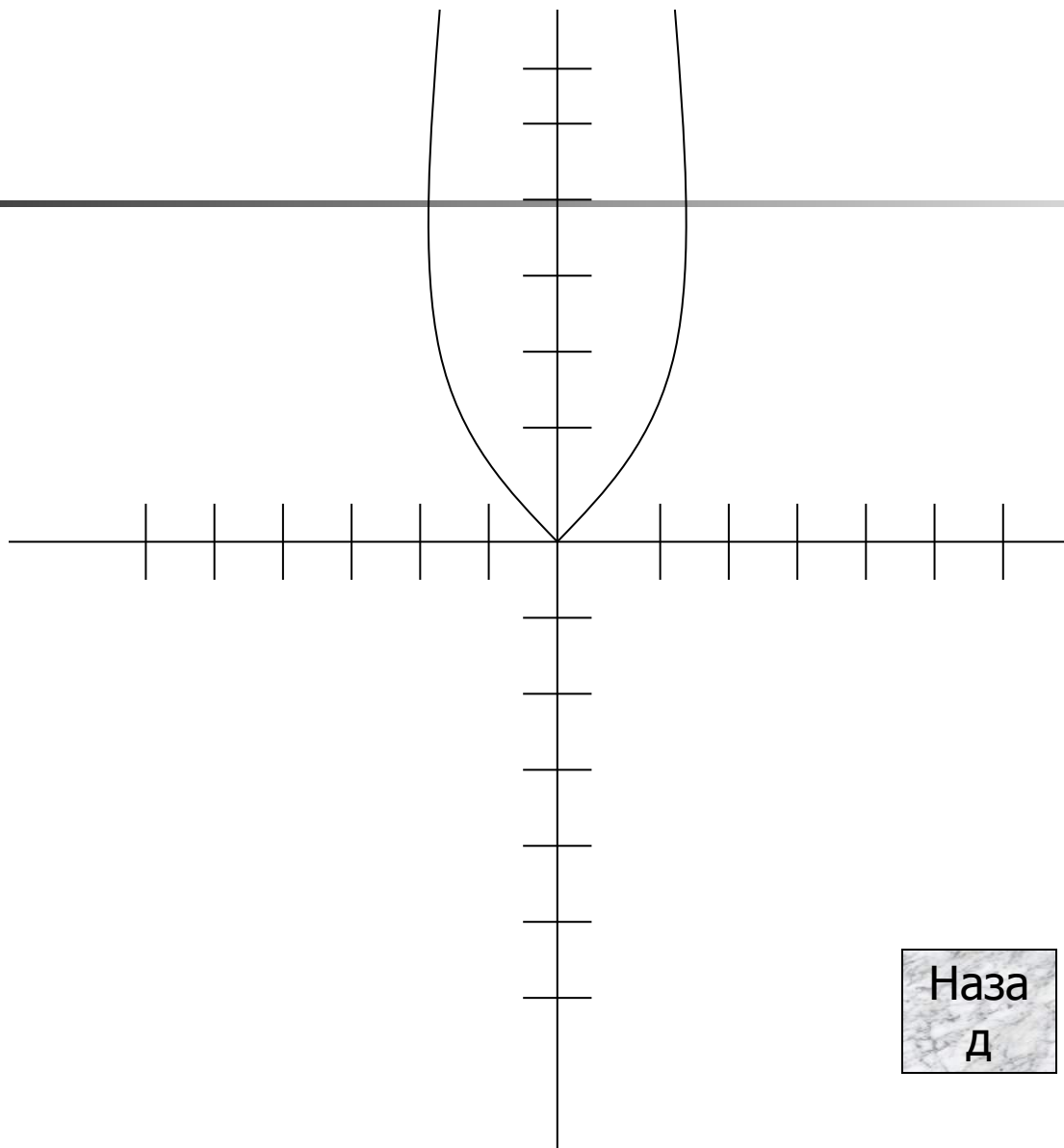
```
1610 NEXT I
1620 PRINT
1630 NEXT J
```

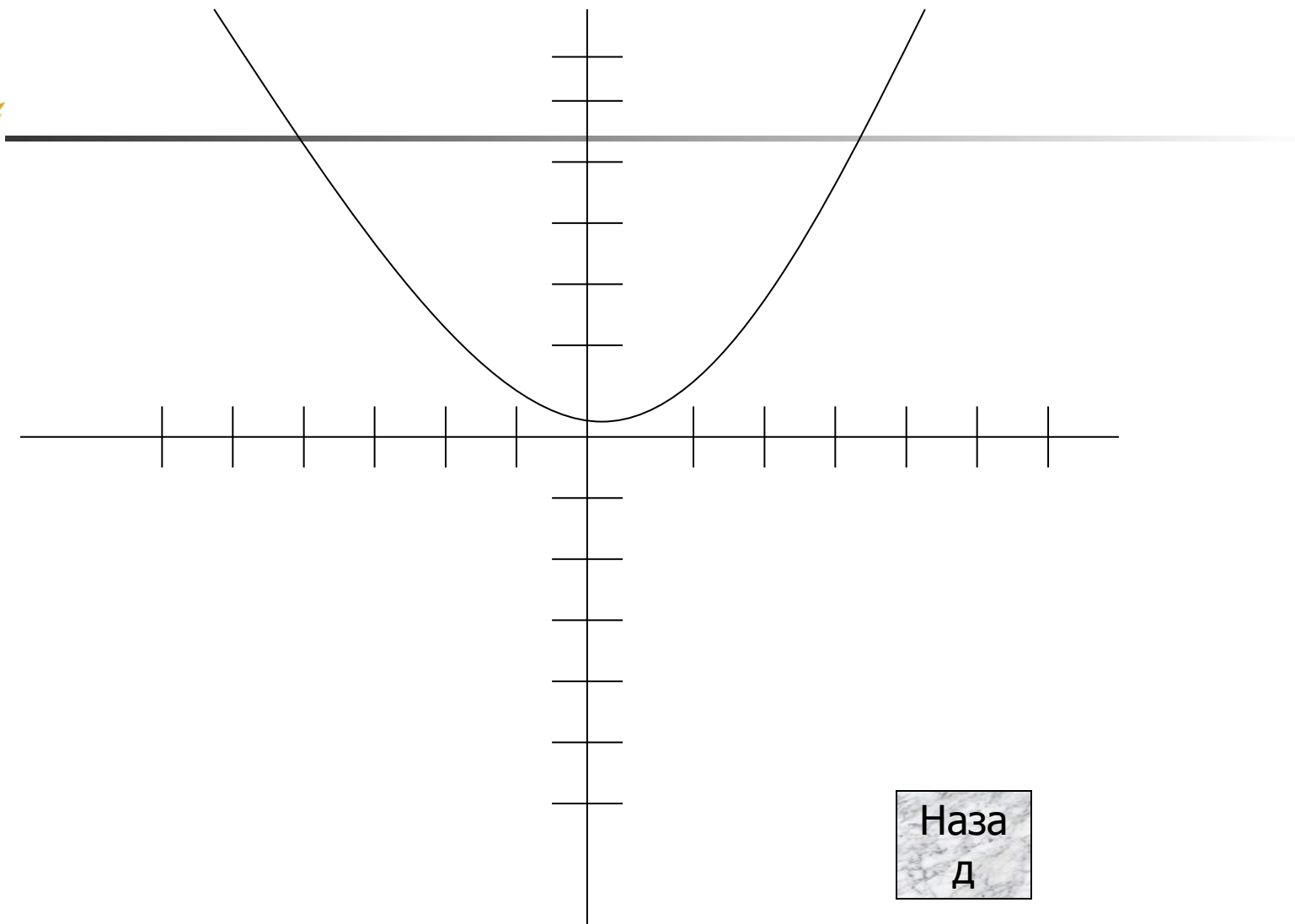
Строка 1620 нужна для того, чтобы вывести массив в виде таблицы

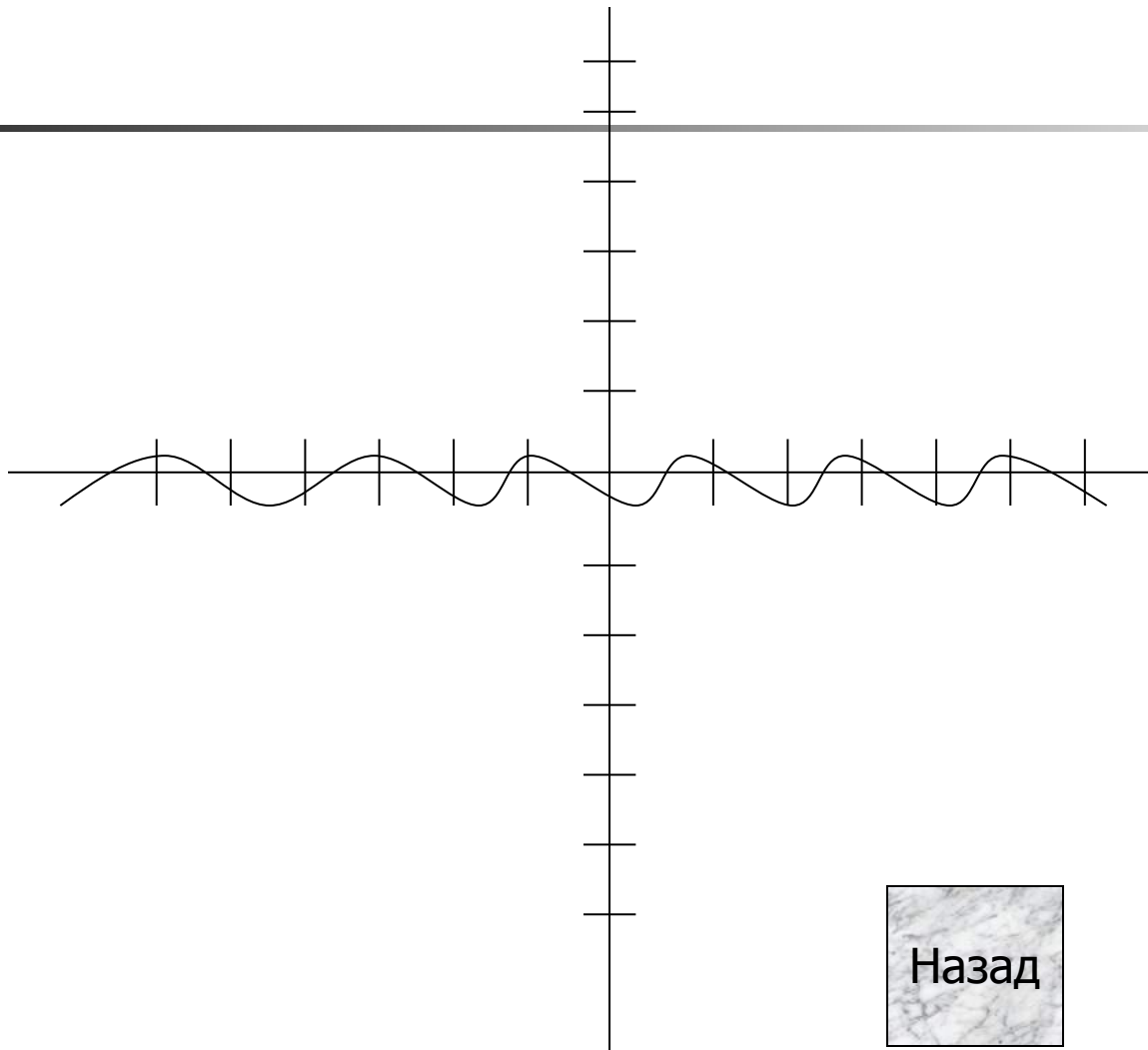
Пример построения фигур с помощью оператора



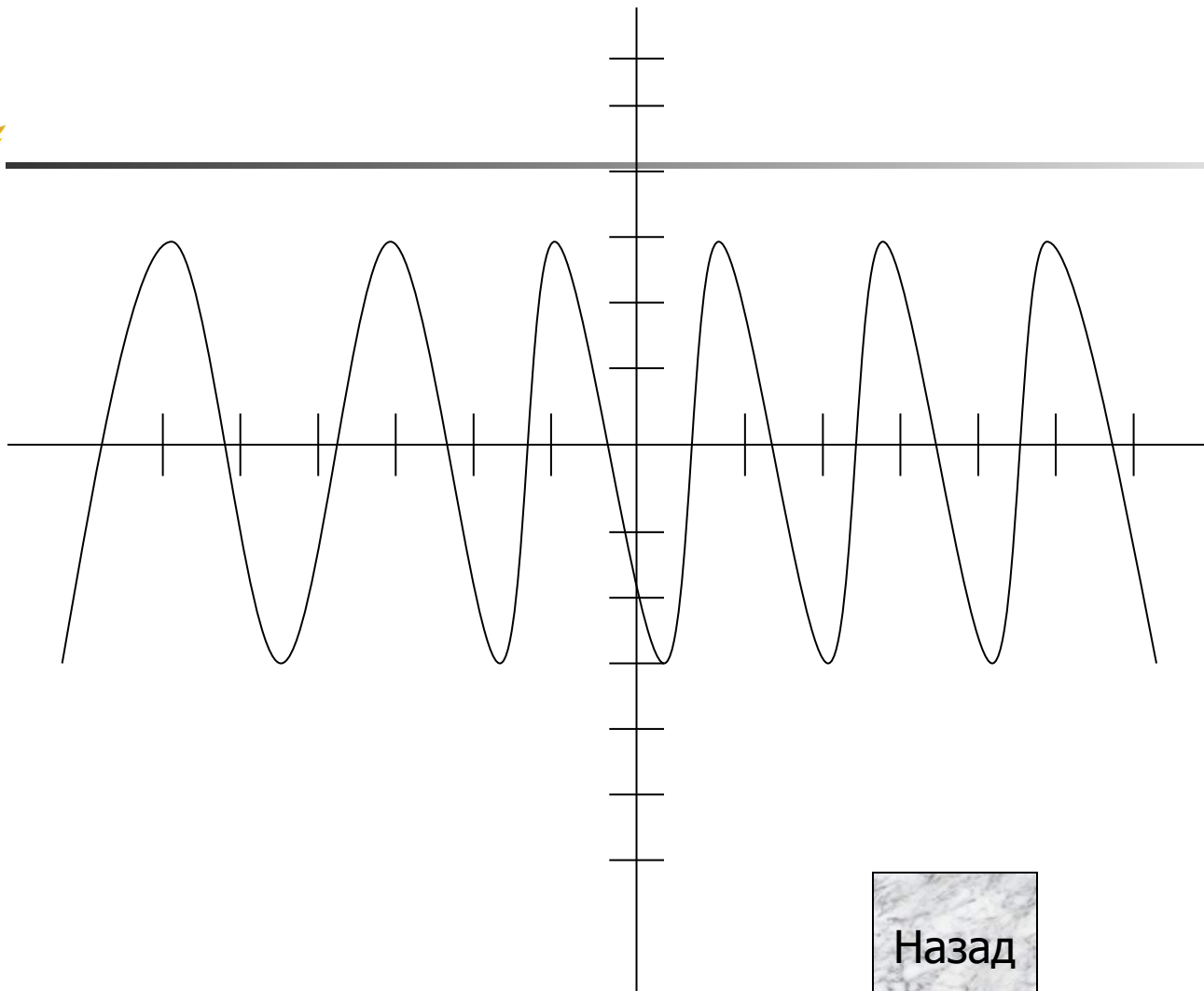
Назад







Назад



Назад