

***Подробности Write.  
Процедура вывода WriteLn.  
Курсор.***

---

Учитель информатики  
МОУ Юрцовская СОШ Егорьевский район  
Сафонова Ольга Викторовна

# Подробности Write

*Как пишем обращение к процедуре*

*Что мы видим в результате на экране*

**Write(3 + 2, 4 + 4)**

**58**

Это не одно число **58**, а два числа: **5** и **8**. «К сожалению», они расположены вплотную друг к другу. Этот «недостаток» оператора **Write** можно преодолеть.

**Write('Это' , 4 + 4 , 'кошек' )**

**Это8кошек**

Один оператор **Write** может выводить сразу несколько элементов. Эти элементы нужно отделять друг от друга запятыми. В этом примере 3 элемента: 1) текст **‘Это’** ; 2) выражение **4 + 4**; 3) текст **‘кошек’** . Все элементы выводятся в одну строку вплотную друг к другу. Если вся информация, выводимая оператором **Write**, не умещается в одну строку, то неуместившаяся часть автоматически выводится с начала следующей строки.

**! – Не путайте запятые и кавычки, находите элементы и отличайте текстовые элементы от чисел и выражений.**

# Совет.

Сначала найдите внутри скобок запятые. Рассмотрим пример:

**Write(8, 'котят', 3 \* 3, 'щенят')** **8котят9щенят**

Здесь запятых три; значит, элементов четыре. Элементы легко заметить, если представить себе, что запятые – это стены, разделяющие элементы.

**8**      **'котят'**      **3 \* 3**      **'щенят'**

Чтобы отличить текстовые элементы от прочих, обратим внимание, что они заключены в кавычки:

**'котят'**

**'щенят'**

# Совет.

Рассмотрим ещё один пример:

**Write('Это',4 + 4, 'кошек' )      Это8кошек**

Результат не зависит от количества пробелов (пропусков, пустых мест) снаружи от текстовых элементов, взятых в кавычки. Но пробелы, встретившиеся внутри кавычек, отображаются на экране:

**Write('Это',4+ 4,'ко   шек' )      Это8ко   шек**

**Write('Это   ',4+ 4,'   кошек' )      Это 8 кошек**

**Write('16+16=' , 16+16 )**

**16+16=32**

Здесь два элемента: текст **'16+16='** и выражение **16+16**. Несмотря на то, что текст похож на выражение, компьютер узнает его по кавычкам и не вычисляет, а просто воспроизводит, как записано: **'16+16='**. Любой элемент, заключенный в кавычки, Паскаль считает текстом.

**Write(3 + 2, ' ', 4 + 4)**

**5 8**

Здесь три элемента. Второй элемент – текст, состоящий из двух пробелов – **' '**. В тексте можно использовать любые символы, имеющиеся на клавиатуре.

# *Задание.*

Изобразите на листке бумаги в клетку (один символ – одна клетка), что напечатает оператор:

**Write('12', '5+1', '=', 120+21)**

# Процедура **WriteLn**.

Оператором **WriteLn** в большинстве случаев пользоваться удобнее, чем оператором **Write**. Оператор **WriteLn** читается как «**райт'лайн**», переводится как «**пиши строку**». Правила его записи и выполнения те же, что и у **Write**, с одним исключением – после его выполнения следующий оператор **Write** или **WriteLn** печатает свою информацию с начала следующей строки, а после выполнения оператора **Write** продолжает печатать в той же.

# Примеры.

*Программа*

*Что видим  
на экране*

```
BEGIN Write ('Ама'); Write ('зонка')  
END.
```

**Амазонка**

```
BEGIN Write ('Ама'); WriteLn('зонка')  
END.
```

**Амазонка**

```
BEGIN WriteLn('Ама'); Write('зонка')  
END.
```

**Ама  
зонка**

```
BEGIN WriteLn('Ама'); WriteLn('зонка')  
END.
```

**Ама  
зонка**

Всё вышесказанное можно более точно описать  
с

# Курсор.

При введении текста на экране монитора вы видите короткую светлую черточку или прямоугольное пятнышко, которые «бегут» на экране перед вводимым текстом. Так, если вы вводите с клавиатуры слово **BEGIN**, то:

- после нажатия на клавишу **B** на экране вы увидите **B\_**;
- после нажатия на клавишу **E** на экране вы увидите **BE\_**;
- после нажатия на клавишу **G** на экране вы увидите **BEG\_**  
и т.д.

Курсор предназначен для того, чтобы показывать пользователю, где на экране появится следующий символ, который он ведет с клавиатуры. Курсор точно так же бежит по экрану впереди текста и тогда, когда информация появляется на мониторе не при вводе с клавиатуры, а во время исполнения программы в результате выполнения операторов **Write** или **WriteLn** .

# Процедуры **Write** и **WriteLn**.

Разница между процедурами **Write** или **WriteLn** в том, что после выполнения **Write** курсор остается в той же строке, а после выполнения **WriteLn** курсор прыгает в начало следующей строки, а значит, и все следующие символы будут выводиться в следующей строке.

Оператор **WriteLn** можно использовать просто для перевода курсора в начало следующей строки. Для этого достаточно написать одно слово **WriteLn**, без скобок.

## Задание.

Определите без компьютера и изобразите на листке бумаги в клетку (один символ – одна клетка), что напечатает программа:

```
BEGIN
```

```
  Write(1992);
```

```
  WriteLn(' Мы начинаем!');
```

```
  WriteLn(6 * 8);
```

```
  WriteLn;
```

```
  WriteLn('Шестью шесть ', 6 * 6, ' '
```

```
Арифметика:' , (6+4) * 3)
```

```
END.
```

Выполнив задание на бумаге, выполните программу на компьютере и сверьте результаты . Совпадает ли число пробелов между символами? Измените число пробелов в разных местах последнего **WriteLn**. Как изменится картинка на экране? Почему? Добавьте рядом с пустым **WriteLn** еще один. Что случилось? Почему?

# *Используемая литература*

## **1. Лукин Н.С.**

Турбо-Паскаль 7.0 . Самоучитель для начинающих – 2-е изд., испр. И допол. – М.: «Диалог-МИФИ», 2005. – 400 с.