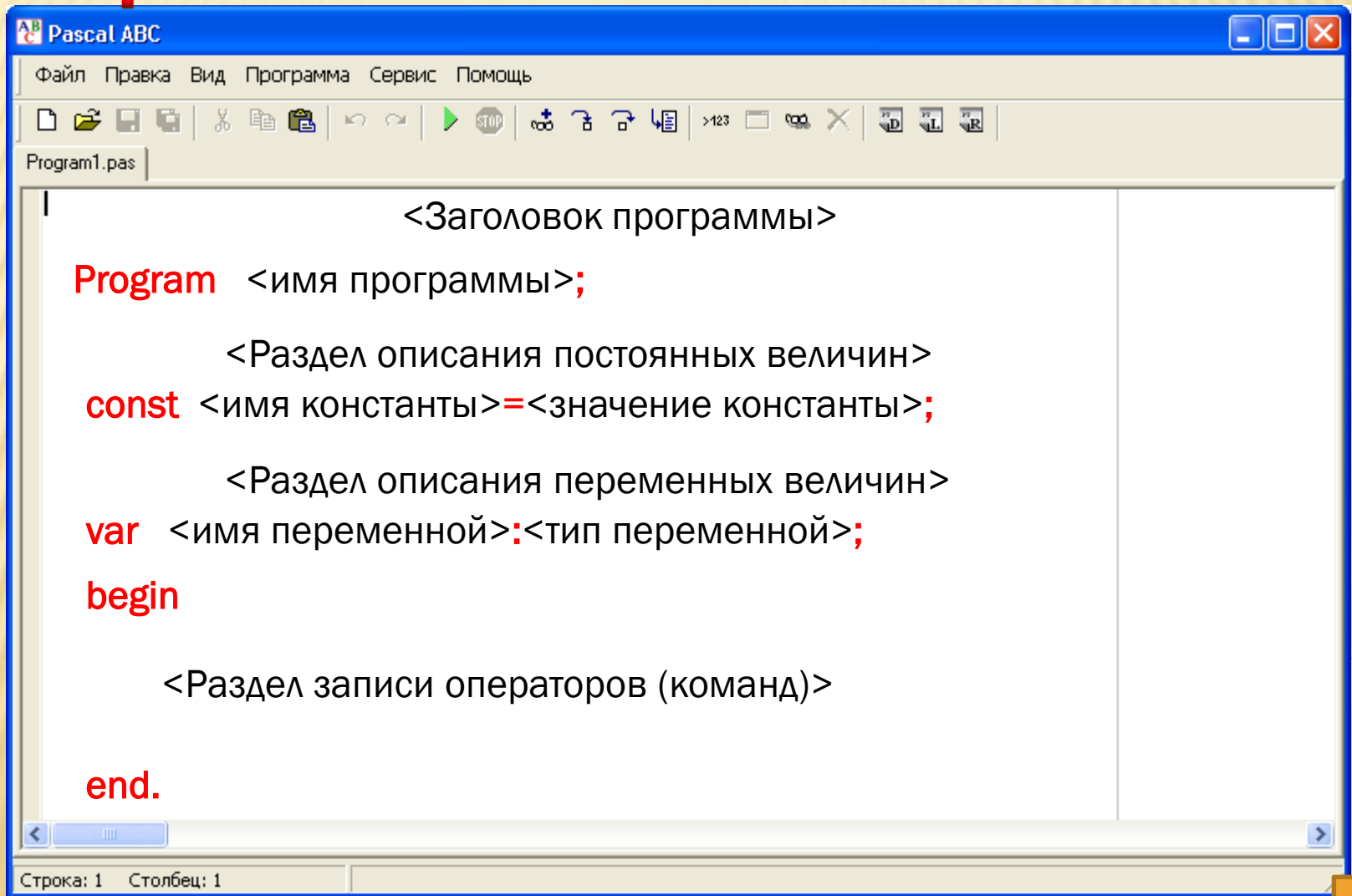


Структура составления программы на языке Паскаль



The image shows a screenshot of a Pascal ABC IDE window. The window title is "Pascal ABC". The menu bar includes "Файл", "Правка", "Вид", "Программа", "Сервис", and "Помощь". The toolbar contains various icons for file operations, editing, and execution. The main text area shows the structure of a Pascal program with the following code:

```
<Заголовок программы>  
Program <имя программы>;  
    <Раздел описания постоянных величин>  
const <имя константы>=<значение константы>;  
    <Раздел описания переменных величин>  
var <имя переменной>:<тип переменной>;  
begin  
    <Раздел записи операторов (команд)>  
end.
```

The status bar at the bottom indicates "Строка: 1" and "Столбец: 1".



Основные понятия

Идентификаторы служат в качестве имен программ, модулей, процедур, функций, типов, переменных и констант. Идентификатором считается любая последовательность латинских букв или цифр, начинающаяся с буквы. Буквой считается также символ подчеркивания "_".

Например, **a1**, **_h**, **b123** - идентификаторы, а **1a**, **ф2** - нет.

Служебные слова служат для оформления конструкций языка и не могут быть использованы в качестве имен.

Имя программы, имена постоянных и переменных являются идентификаторами!

Любое выражение имеет определенный тип и после вычисления возвращает некоторое значение. Простейшими выражениями являются переменные и константы. Более сложные выражения строятся из более простых с использованием операций, скобок, вызовов функций, индексов и приведений типов. Данные, к которым применяются операции, называются *операндами*.

В **Pascal ABC** имеются следующие операции: @, **not**, ^, *, /, **div**, **mod**, **and**, **shl**, **shr**, +, -, **or**, **xor**, =, >, <, <>, <= и >=



Типы переменных

Тип **integer** (целый). Значения этого типа занимают 4 байта и находятся в диапазоне от -2147483648 до 2147483647. Константа `MaxInt` хранит значение 2147483647.

Тип **byte** (беззнаковый целый). Значения этого типа занимают 1 байт и находятся в диапазоне от 0 до 255.

Тип **word** (беззнаковый целый). Значения этого типа занимают 2 байта и находятся в диапазоне от 0 до 65535.

Тип **char** (символьный). Значения этого типа занимают 1 байт и представляют собой символы в кодировке Windows. Стандартная функция `Chr(x)` возвращает символ с кодом `x`. Константы этого типа могут быть записаны в виде `#x`, где `x` - целое число от 0 до 255.

Тип **real** (вещественный). Значения вещественного типа занимают 8 байт, содержат 15-16 значащих цифр и по модулю не могут превосходить величины $1.7 \cdot 10^{308}$. Самое маленькое положительное число вещественного типа равно $5.0 \cdot 10^{-324}$. Константы типа `real` можно записывать как в форме с плавающей точкой, так и в экспоненциальной форме: 1.7, 0.013, 2.5e3 (2500), 1.4e-1 (0.14).



Описание переменных

Раздел описания переменных начинается со служебного слова **var**, после которого следуют строки вида

список имен переменных: тип;

Имена в списке перечисляются через запятую.
Например:

```
var  
a,b,c: integer;  
d: real;  
e,f: integer;  
ch: char;
```



Описание постоянных

Раздел описания именованных констант начинается со служебного слова **const**, после которого следуют строки вида

ИМЯ КОНСТАНТЫ = значение;

или

ИМЯ КОНСТАНТЫ : тип = значение;

Например:

```
const  
Pi = 3.14;  
Count = 10;  
Name = 'Mike';  
DigitsSet = ['0'..'9'];
```



Оператор вывода

Для вывода информации используются стандартные процедуры **write** и **writeln**.

Процедура **writeln** после вывода своих параметров осуществляет переход на следующую строку (в отличие от **write**)

Например, если *a*, *b* - целые переменные, то при выполнении операторов

```
a:=-2437; b:=13555;  
writeln(a:6,'Привет!':9);  
writeln(b:1);
```

в окно вывода будет выведен следующий текст:

```
-2437 Привет!  
13555
```

Оператор вывода

Для вещественных и комплексных значений можно также использовать формат **`:m:n`**, где *m* и *n* - целые значения. Значение *m* задает ширину поля вывода, а значение *n* *количество знаков после десятичной точки*.

Например:

```
writeln(-14.859:10:3); // ___-14.859  
writeln(-14.859:10:5); // _-14.85900  
writeln(-14.859:10:2); // _____-14.86  
writeln(-14.859:10:0); // _____-15  
writeln(-14.859:10:7); // -14.8590000  
writeln((0,1):10:1); // _(0.0,1.0)
```

(здесь символом `_` изображены пробелы).

Оператор ввода

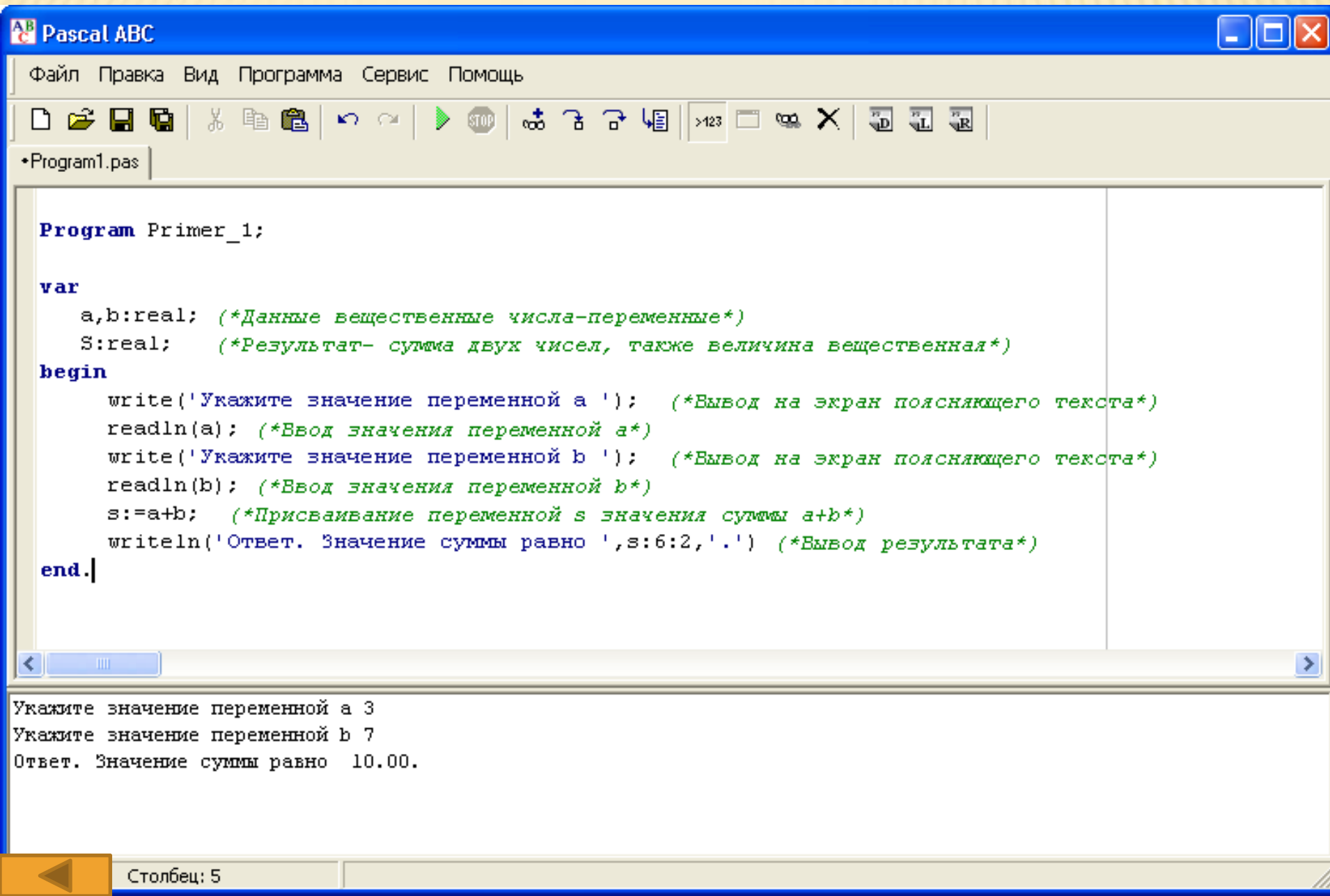
Для ввода с клавиатуры используются стандартные процедуры **read** и **readln**.

Они могут вызываться как без параметров, так и со списком параметров. Параметры в списке перечисляются через запятую и должны быть переменными простого типа (кроме перечислимого типа и интервального типа, построенного на базе перечислимого), либо типа **string**.

Процедура **readln** после ввода пропускает данные до конца текущей строки ввода.



Пример программы нахождения суммы двух вещественных чисел



```
Program Primer_1;

var
  a,b:real; (*Данные вещественные числа-переменные*)
  S:real;    (*Результат- сумма двух чисел, также величина вещественная*)
begin
  write('Укажите значение переменной a '); (*Вывод на экран поясняющего текста*)
  readln(a); (*Ввод значения переменной a*)
  write('Укажите значение переменной b '); (*Вывод на экран поясняющего текста*)
  readln(b); (*Ввод значения переменной b*)
  s:=a+b;   (*Присваивание переменной s значения суммы a+b*)
  writeln('Ответ. Значение суммы равно ',s:6:2,'.') (*Вывод результата*)
end.
```

Укажите значение переменной a 3
Укажите значение переменной b 7
Ответ. Значение суммы равно 10.00.

Столбец: 5