

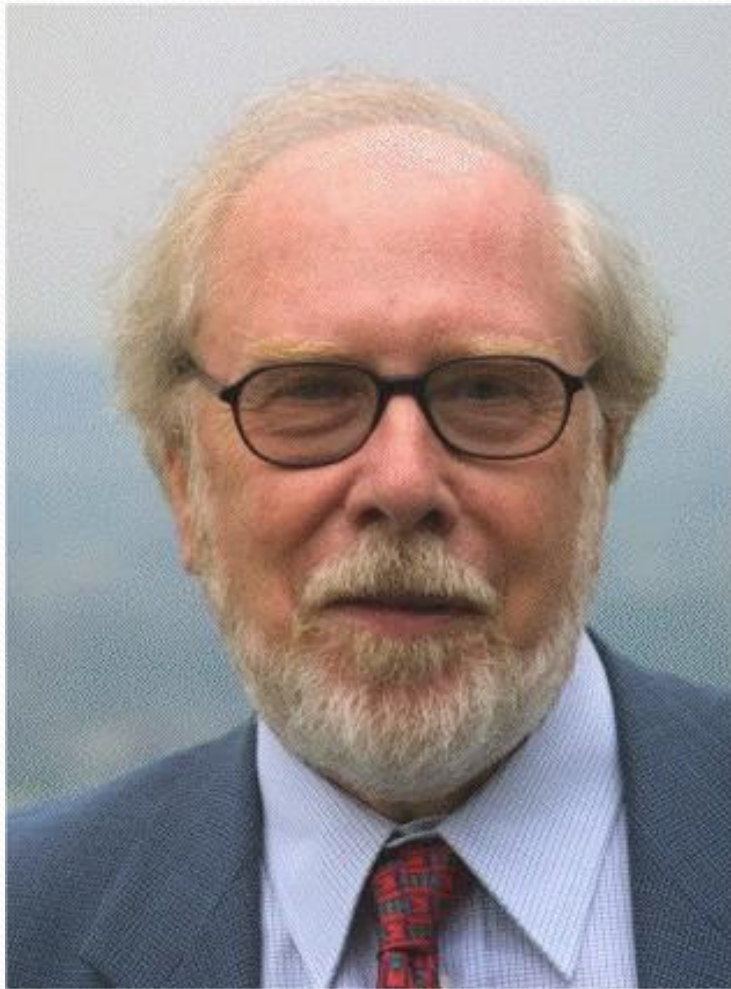
# ОСНОВЫ ПРОГРАММИРОВАНИЯ НА ЯЗЫКЕ PASCAL

# Немного истории...



Язык назван в честь выдающегося французского математика, физика, литератора и философа Блеза Паскаля, который создал первую в мире механическую машину, складывающую два числа.

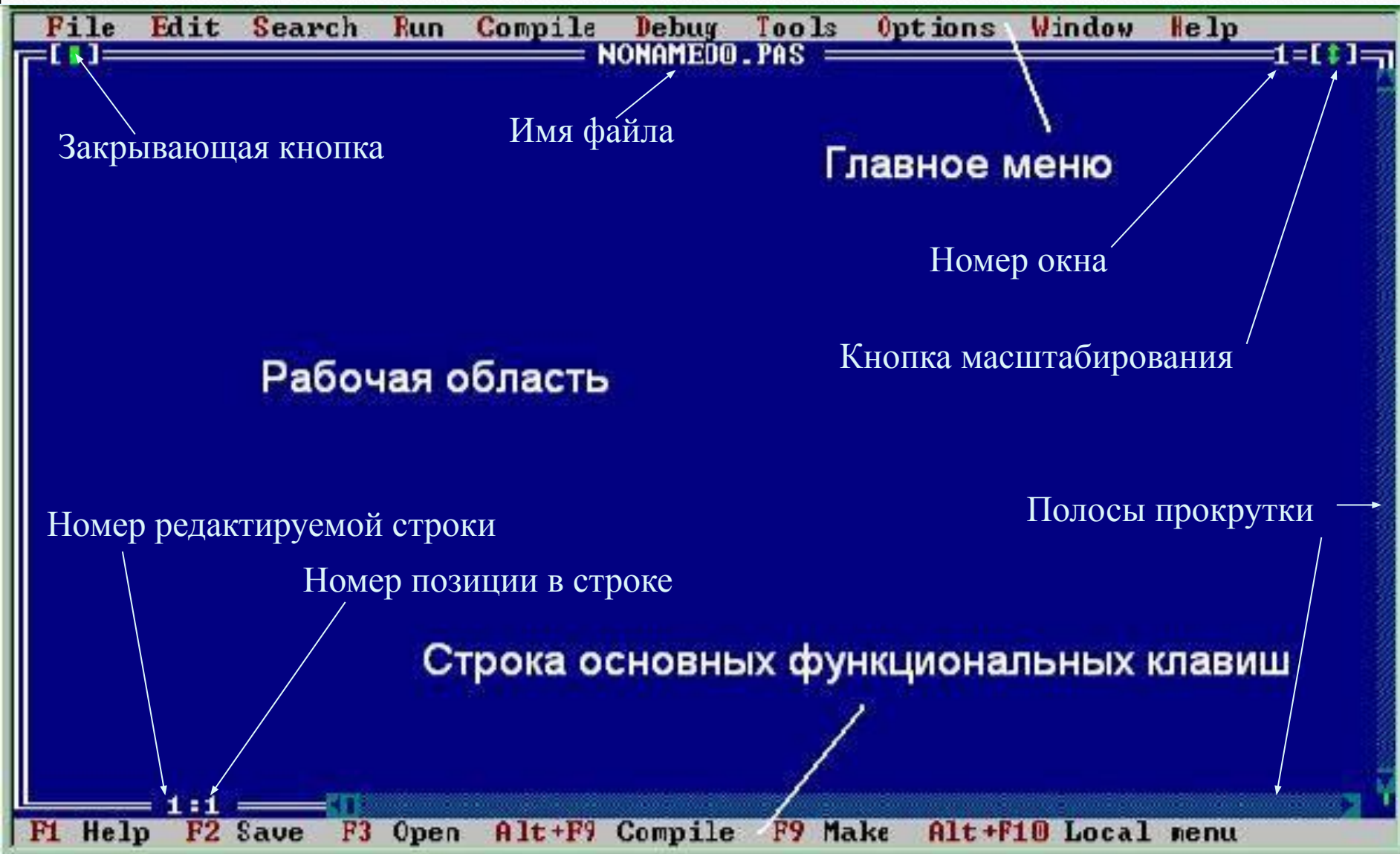
# Немного истории...



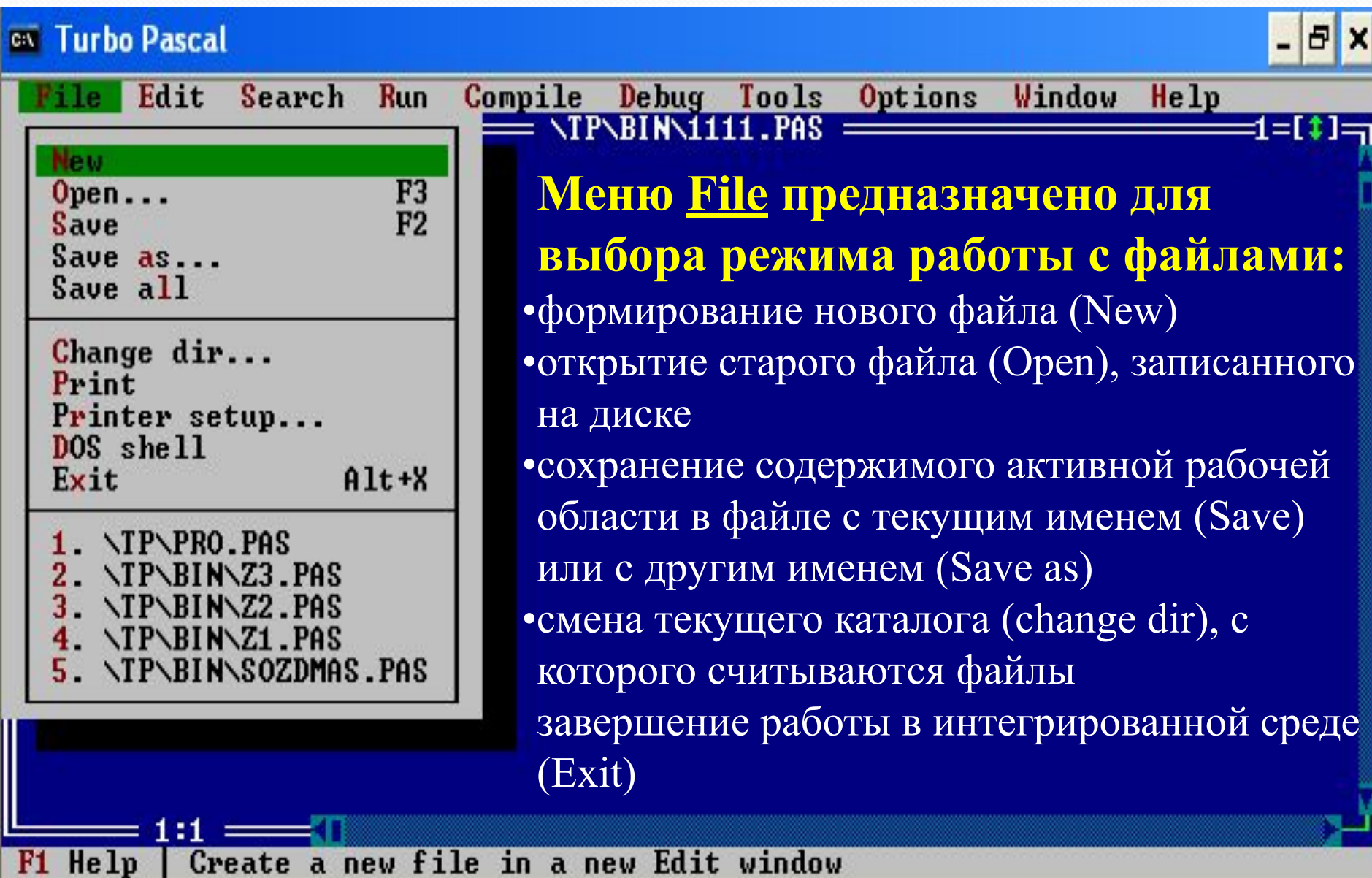
Язык Паскаль был создан Никлаусом Виртом в 1968—1969 годах. Он был опубликован в 1970 году как небольшой и эффективный язык, чтобы способствовать хорошему стилю программирования, использовать структурное программирование и структурированные данные.



# Элементы экрана



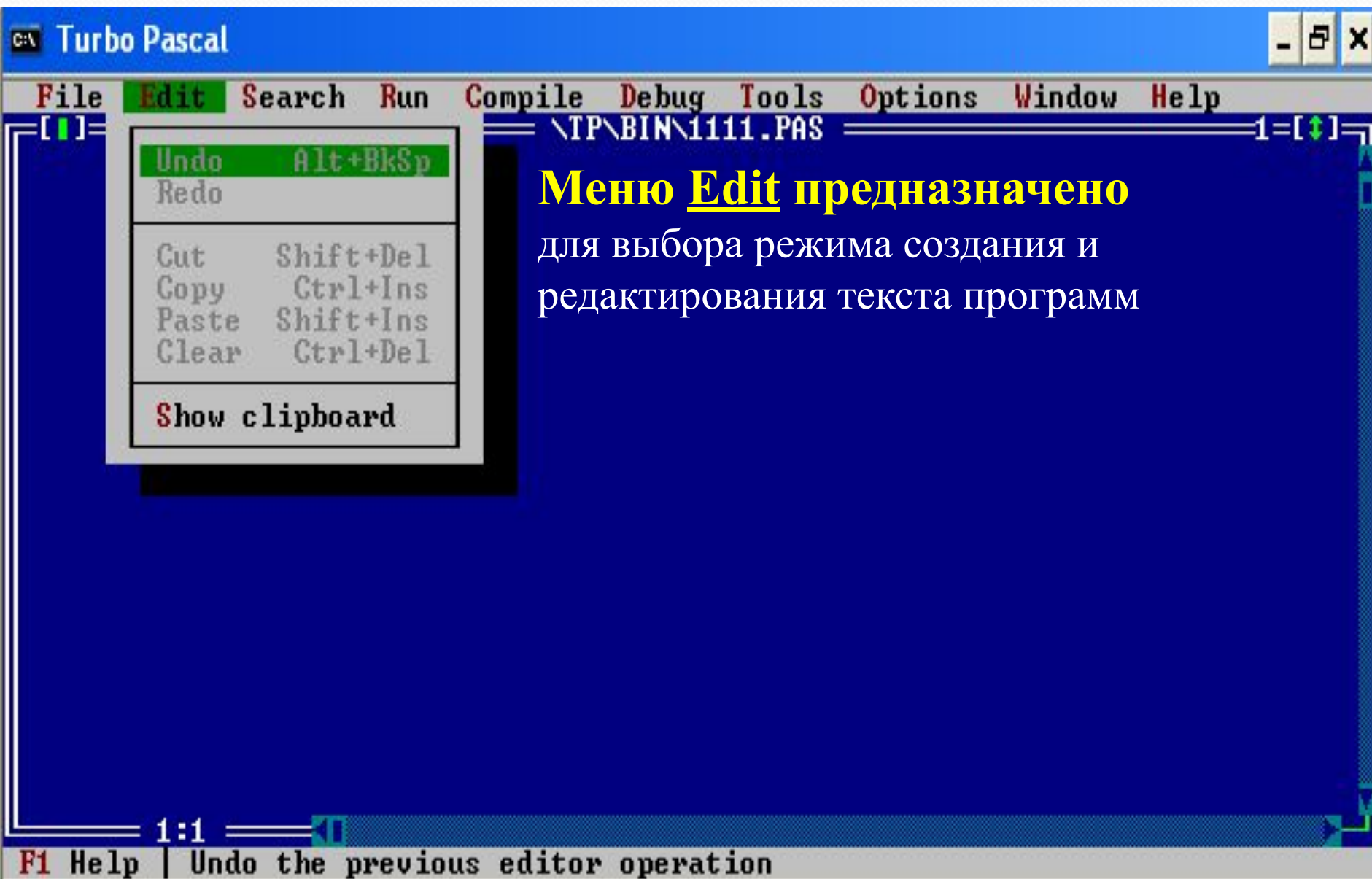
# Назначение пунктов меню



**Меню File предназначено для выбора режима работы с файлами:**

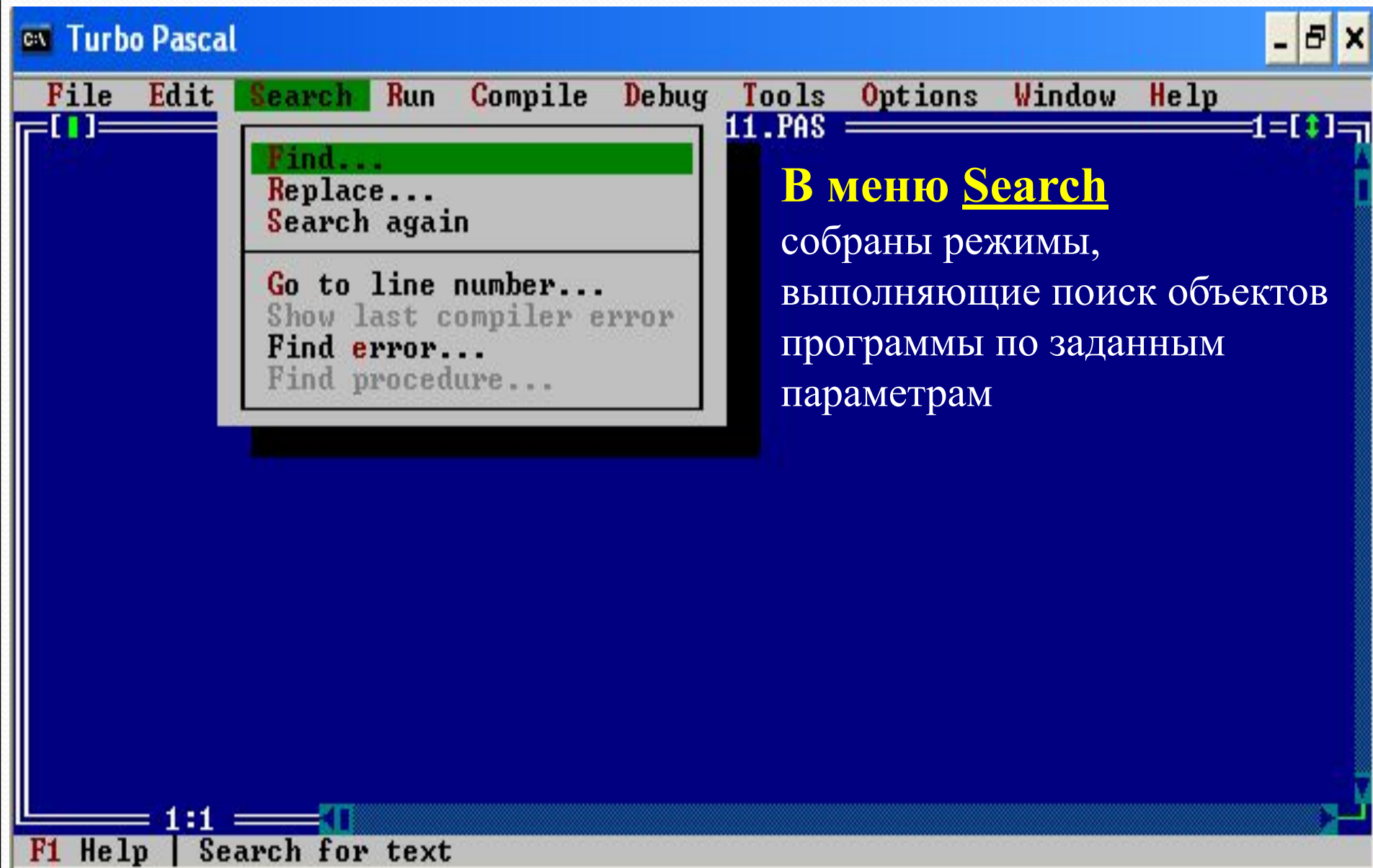
- формирование нового файла (New)
- открытие старого файла (Open), записанного на диске
- сохранение содержимого активной рабочей области в файле с текущим именем (Save) или с другим именем (Save as)
- смена текущего каталога (change dir), с которого считываются файлы
- завершение работы в интегрированной среде (Exit)

# Назначение пунктов меню





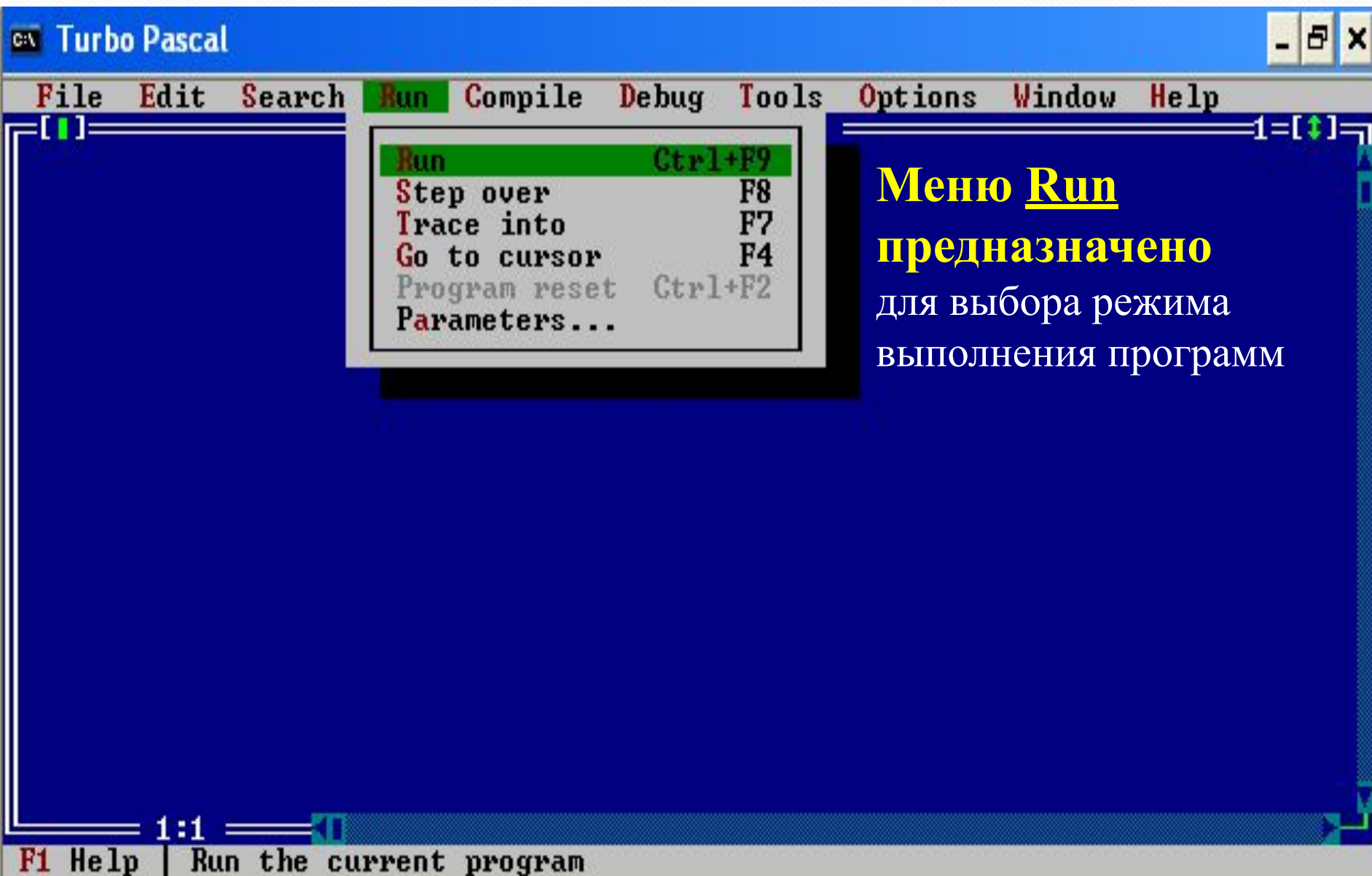
# Назначение пунктов меню



## В меню Search

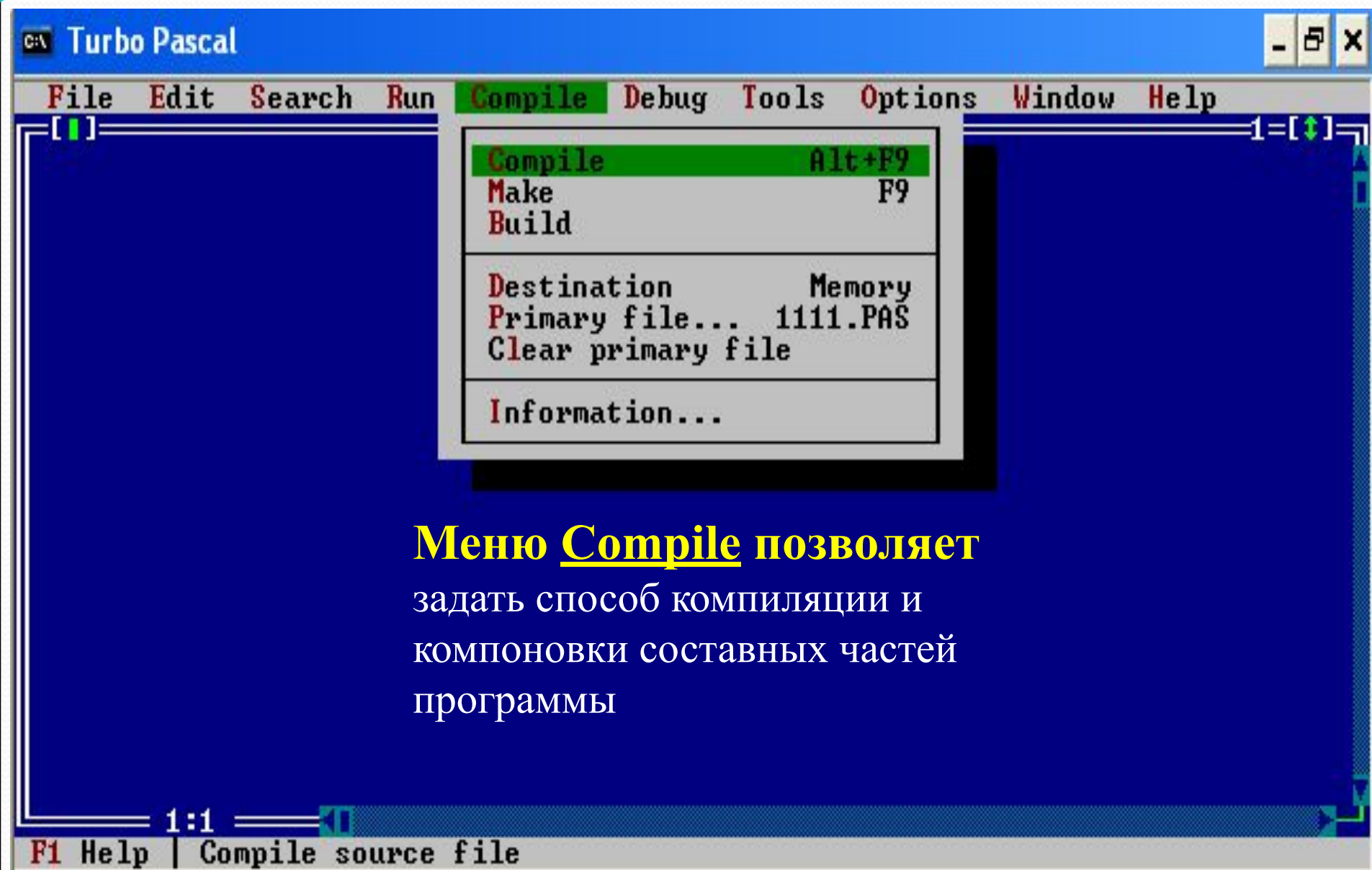
собраны режимы,  
выполняющие поиск объектов  
программы по заданным  
параметрам

# Назначение пунктов меню





# Назначение пунктов меню

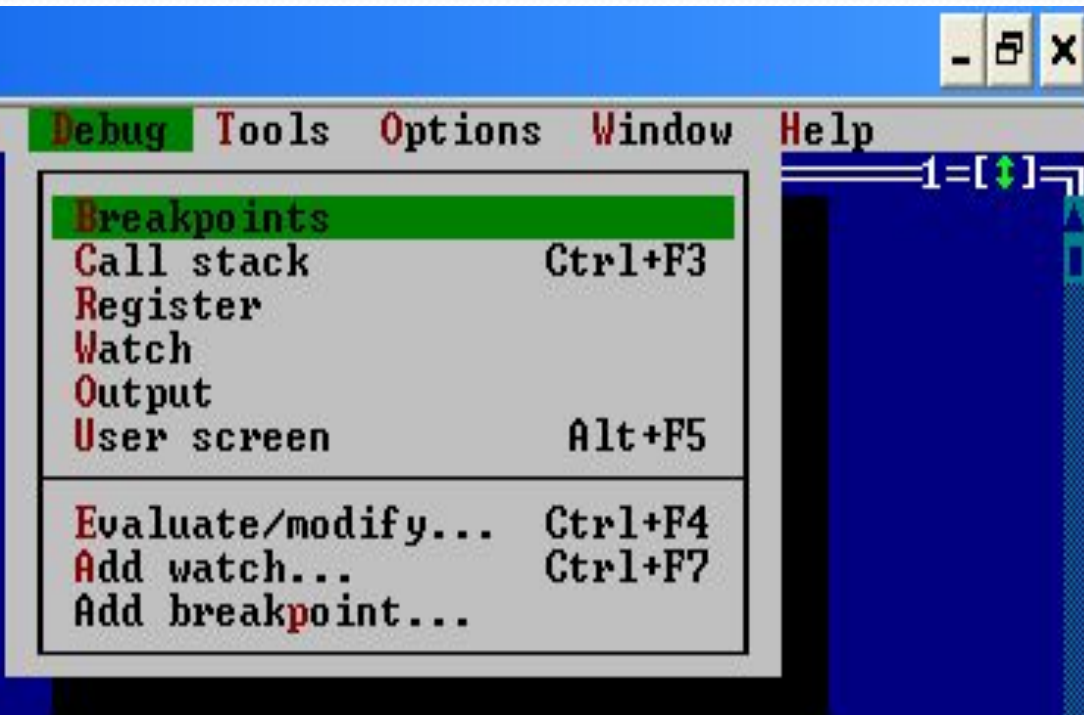


**Меню Compile позволяет**  
задать способ компиляции и  
компоновки составных частей  
программы

# Назначение пунктов меню

## Меню Debug

**выполняется** настройка отладчика, в частности выбор переменных, значения которых выводятся в окно наблюдения

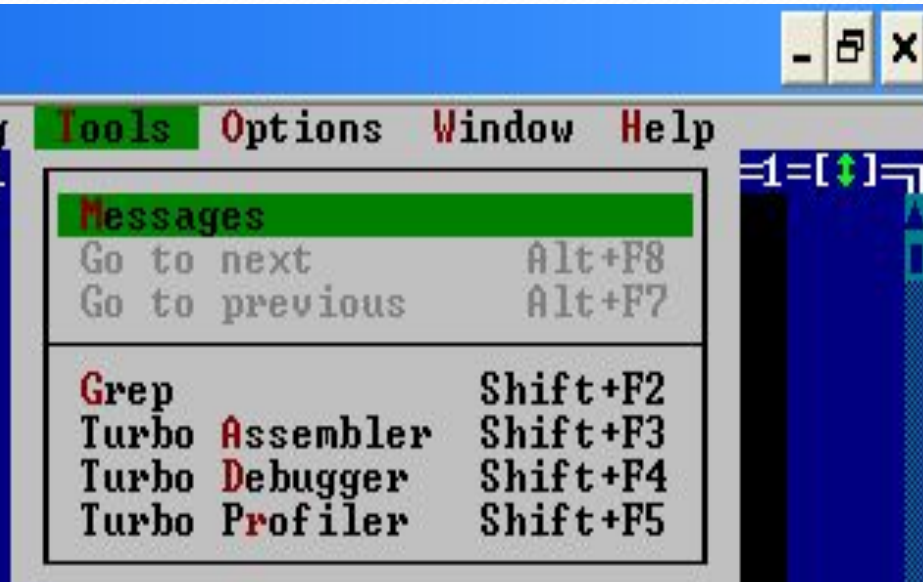


1:1

**F1** Help | Set conditional breakpoints, view and edit breakpoints

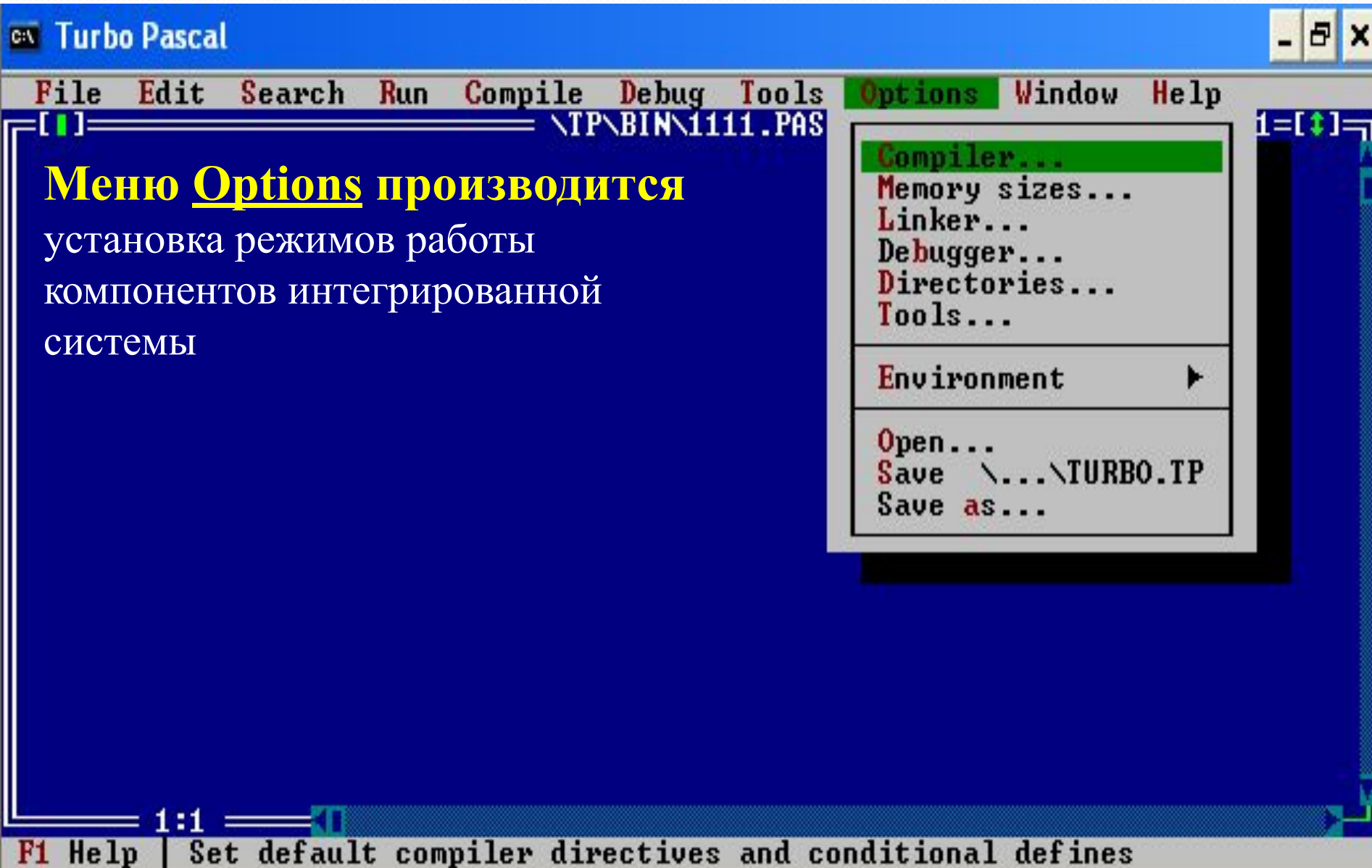
# Назначение пунктов меню

**Меню Tools** позволяет  
обращаться к ассемблеру и  
отладчику





# Назначение пунктов меню



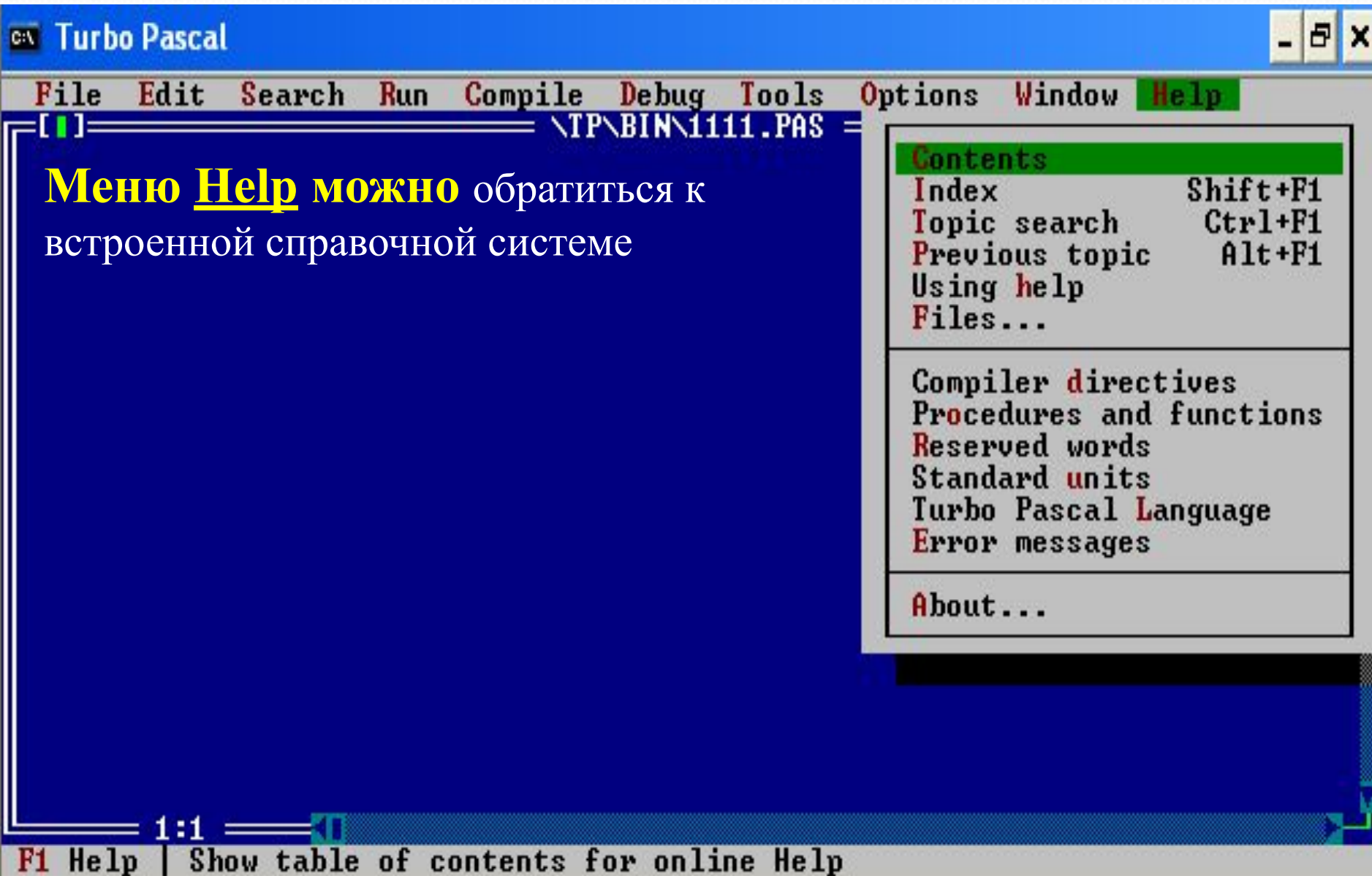
# Назначение пунктов меню

**Menu Windows production**  
installation of the current active work area and the method of displaying the content of the work area (there can be several, that is simultaneously possible to open several programs)

<b>Tile</b>	
Cascade	
Close all	
Refresh display	
<b>Size/Move</b>	Ctrl+F5
<b>Zoom</b>	F5
<b>Next</b>	F6
<b>Previous</b>	Shift+F6
<b>Close</b>	Alt+F3
<b>List...</b>	Alt+0

F1 Help | Arrange windows on desktop by tiling

# Назначение пунктов меню



The screenshot shows the Turbo Pascal IDE window titled "C:\ Turbo Pascal". The menu bar includes "File", "Edit", "Search", "Run", "Compile", "Debug", "Tools", "Options", "Window", and "Help". The "Help" menu is open, displaying a list of options: "Contents" (highlighted), "Index" (Shift+F1), "Topic search" (Ctrl+F1), "Previous topic" (Alt+F1), "Using help", "Files...", "Compiler directives", "Procedures and functions", "Reserved words", "Standard units", "Turbo Pascal Language", "Error messages", and "About...". The main editor area has a blue background and contains the text: "Меню Help можно обратиться к встроенной справочной системе". The status bar at the bottom shows "F1 Help | Show table of contents for online Help".

File Edit Search Run Compile Debug Tools Options Window **Help**

[ ] \TP\BIN\1111.PAS

**Меню Help можно** обратиться к  
встроенной справочной системе

**Contents**  
Index Shift+F1  
Topic search Ctrl+F1  
Previous topic Alt+F1  
Using help  
Files...

Compiler directives  
Procedures and functions  
Reserved words  
Standard units  
Turbo Pascal Language  
Error messages

About...

1:1

F1 Help | Show table of contents for online Help



# Структура программы

**Program** Имя программы;

**Uses** Подключаемые библиотеки (модули);

**Label** Список меток основной программы;

**Const** Введение констант;

**Type** Описание новых типов;

**Var** Описание переменных;

Определение процедур;

Определение функций;

**Begin**

Тело основной программы;

**End.**

# Константы

*Константы* – это данные, значения которых в процессе выполнения программы не могут изменяться.

Константы вводятся в блоке const:

```
const
```

```
  a=5;
```

```
  b=1E-3/a;
```

```
  c='значение неизвестно';
```

В общем виде:

**ИМЯ КОНСТАНТЫ = выражение;**

# Переменные

*Переменные* – это данные, которые могут изменяться в процессе выполнения программы.

Переменные имеют имя, тип и значение.

Описание переменных происходит в блоке var:

```
var
```

```
  a: integer;
```

```
  b: real;
```

```
  c: char;
```

В общем виде:

**имя переменной: тип переменной;**



# Типы переменных

Некоторые простые типы:

1. **Целые типы** (ShortInt, Integer, LongInt, Byte, Word).
2. **Вещественные типы** (Real, Single, Double, Extended, Comp).
3. **Логический** (Boolean).
4. **Символьный** (Char).
5. **Строковые типы** (String, String [n]).

# Целочисленные типы:

Тип	Диапазон	Формат	Размер в байтах
Byte	0..255	Беззнаковый	1
ShortInt	-128..127	Знаковый	1
SmallInt	-32768..32767	Знаковый	2
Word	0..65535	Беззнаковый	2
Integer	-32768..32767	Знаковый	2
Cardinal	=LongWord	Беззнаковый	4
LongWord	0..4294967295	Беззнаковый	4
LongInt	-2147483648..2147483647	Знаковый	4
Int64	-9223372036854775808..9223372036854775807	Знаковый	8
QWord	0..18446744073709551615	Беззнаковый	8



# Вещественные типы:

Тип	Диапазон	Количество значащих цифр	Размер в байтах
Real/Double	зависит от платформы	???	8
Real48	???	11-12	6
Single	1.5E-45..3.4E38	7-8	4
Extended	1.9E-4932..1.1E4932	19-20	10
Comp	-2E64+1..2E63-1	19-20	8
Currency	-9223372036854775808..9223372036854775807	19-20	8



# Оператор присваивания:

Общий вид:

**переменная := выражение;**

Может быть: переменной,  
элементом массива,  
полем записи и др.

Не может быть:  
константа, выражение.

Может быть: константой,  
переменной, элементом  
массива, арифметическим  
или логическим  
выражением.

Работа оператора: если справа стоит выражение, то сначала вычисляется его значение, а затем это значение пересылается в переменную стоящую слева.

# Арифметические выражения

*Арифметические выражения* – это конструкции, содержащие данные, знаки математических операций, математические функции.

Название	Знак	Тип операндов	Тип результата	Пример
Сложение	+	Целое, вещ.	Целое вещ.	$12.5 + 3 \{=15.5\}$
Вычитание	-	Целое, вещ.	Целое вещ.	$15.5 - 3 \{=12.5\}$ $15 - 3 \{=12\}$
Умножение	*	Целое, вещ.	Целое вещ.	$3.5 * 2 \{=7.0\}$ $3 * 2 \{=6\}$
Деление	/	Целое вещ.	вещ.	$3/5 \{=0.6\}$
Целое частное от деления	div	целое	целое	$3 \text{ div } 5 \{=0\}$
Целый остаток от деления	mod	целое	целое	$3 \text{ mod } 5 \{=3\}$

# Операторы ввода и вывода

Оператор ввода:

```
read (список переменных);
```

Оператор вывода:

```
write (‘сообщение’, список переменных);
```

Отличие операторов **read** и **write** от операторов **readln** и **writeln** состоит в том, что после выполнения операторов **readln** и **writeln** курсор переводится на новую строку.



# Оператор условия

Для реализации ветвления можно использовать условный оператор:

```
if условие then  
    begin  
        оператор; ...  
    end  
else  
    begin  
        оператор; ...  
    end;
```

Если условие выполняется, то выполняется ветвь **then**, если условие не выполняется – то ветвь **else**.

Где *условие* – это выражение логического типа.

# Оператор цикла со счётчиком

```
for параметр := нач_значение to кон_значение do  
    begin  
        оператор; ...  
    end;
```

## Работа оператора:

1. Вычисляются начальное и конечное значения параметра и фиксируются;
2. Если *нач\_значение*  $\leq$  *кон\_значения*, то выполняется оператор;
3. Значение параметра цикла возрастает (для целого – на единицу);
4. Если значение параметра  $\neq$  *кон\_значения*, то выполняется оператор, и переходит к п.3
5. Цикл выполняется последний раз, когда параметр = *кон\_значению*, затем управление передается оператору после цикла.

# Оператор цикла с предусловием

```
while условие do  
    begin  
        оператор; ...  
    end;
```

## Работа оператора:

Сначала проверяется условие, если оно верно, то выполняется оператор, затем опять проверяется условие и т.д., пока условие не перестанет выполняться.

Если условие не верно, то оператор игнорируется и управление передается следующему за циклом оператору.



# Оператор цикла с постусловием

**repeat**

*оператор; ...*

**until** *условие;*

Работа оператора:

Выполнение операторов  
повторяется, пока условие  
не станет верным.

# Тест

Вопрос №1

Вопрос №2

Вопрос №3

Вопрос №4

Вопрос №5

Вопрос №6

Вопрос №7

**Завершить тест**

# Вопрос № 1

Определить значение переменной  $c$  после выполнения фрагмента программы.

```
a := -5;  
a := a + 6;  
b := -a;  
c := a - 2 * b;
```

1)  $c = -11$

3)  $c = 27$

2)  $c = 15$

4)  $c = 33$



# Вопрос № 2

Определить значение целочисленных переменных  $a$  и  $b$  после выполнения фрагмента программы.

```
a:=3+8*4;  
b:=(a div 10)+14;  
a:=(b mod 10)+2;
```

1)  $a = 0, b = 18$

3)  $a = 10, b = 18$

2)  $a = 11, b = 19$

4)  $a = 9, b = 17$

# Вопрос № 3

Определить значение переменной  $c$  после выполнения следующего фрагмента программы.

```
a:=100;  
b:=30;  
a:=a-b*3;  
if a>b then  
  c:=a-b  
else c:=b-a;
```

1)  $c = 20$

3)  $c = -20$

2)  $c = 70$

4)  $c = 180$



# Вопрос № 4

Определить значение целочисленных переменных  $x$ ,  $y$  и  $z$  после выполнения фрагмента программы.

```
x:=52;  
y:=x mod 10;  
z:=x div 10;  
x:=y*10+z;
```

1)  $x = 55, y = 2, z = 5$

3)  $x = 25, y = 5, z = 2$

2)  $x = 22, y = 2, z = 5$

4)  $x = 25, y = 2, z = 5$



# Вопрос № 5

Определить значение переменной  $b$  после выполнения фрагмента программы.

```
a:=2;  
b:=12;  
for i:=1 to 5 do  
    a:=a+3;  
b:=b+a;
```

1)  $b = 65$

2)  $b = 67$

3)  $b = 29$

4)  $b = 17$

# Вопрос № 6

Определить значение переменной  $b$  после выполнения фрагмента программы.

```
a:=1;  
b:=20;  
while a<8 do  
begin  
  b:=b-3;  
  a:=a*2;  
end;
```

1)  $b = 16$

2)  $b = 8$

3)  $b = 28$

4)  $b = 17$



# Вопрос № 7

Определить значение переменной  $b$  после выполнения фрагмента программы.

```
x:=0;  
y:=3;  
repeat  
  x:=2*x-y;  
  y:=y+2;  
until y>10
```

1)  $b = -74$

2)  $b = -67$

3)  $b = 47$

4)  $b = -47$





# Верно

## Список вопросов



# Не верно

Список вопросов

# Домашнее задание

Составить программу нахождения произведения двух чисел, значение которых вводится с клавиатуры, а результат произведения выводится на экран.