

# Основы алгоритмизации и программирования

ФИСТ 1 курс

Власенко

Олег

Федосович

Лекция 4

Управляющие конструкции Си:

if, do while, while

Трассировка

Знакомство с консольным вводом/выводом:

scanf, printf

# Основные типы данных

Тип	Длина байт	Диапазон значений	Минимальное значение	Максимальное значение
char				
double				
short				
long				

# Составить таблицу символов

```
#include <stdio.h>
```

```
void main() {  
    char ch = ' ';  
  
    int i = 0;  
    do {  
        printf("%4d--> '%c'\t", ch, ch);  
        ch = ch + 1;  
        i = i + 1;  
    } while (i <= 256);  
}
```

# Основные типы данных (ASCII)

Тип	Длина байт	Диапазон значений	Минимальное значение	Максимальное значение
char	<b>1</b>	<b>256</b>	<b>-128</b>	<b>+127</b>
double				
short				
long				

# Подсчитать MAX short

```
void main() {  
    short i = 1;  
    long n = 0;  
    do {  
        i = i + 1;  
        n = n + 1;  
    } while (i > 0);  
    printf("%li\n", n);  
}
```

# Основные типы данных

Тип	Длина байт	Диапазон значений	Минимальное значение	Максимальное значение
char	1	256	-128	+127
double				
short				<b>32767</b>
long				

# Сколько байт в short и long?

```
void main() {  
  
    short i;  
    long l;  
  
    printf("sizeof short = %d\n", sizeof(i));  
    printf("sizeof long = %d\n", sizeof(l));  
}
```

# Основные типы данных

Тип	Длина байт	Диапазон значений	Минимальное значение	Максимальное значение
char	1	256	-128	+127
double				
short	2			32767
long	4			



# Основные типы данных

Тип	Длина байт	Диапазон значений	Минимальное значение	Максимальное значение
char	1	$2^8 = 256$	-128	+127
double	8	<a href="#">IEEE 754 standard</a>	$2.22507e-308$	$1.79769e+308$
short	2	$2^{16} = 65\,536$	-32 768	32767
long	4	$2^{32} = 4,294,967,296$	-2,147,483,648	+2,147,483,647

# Строка форматирования

Тип	scanf/printf
char	%c
short	%hi
int	%d или %i
long	%li
float	%f
double	%lf
long double	%Lf

Консоль – что из себя представляет.  
Знакоместо – что это такое.

# Поиск корней квадратного уравнения

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
#include <math.h>

void main() {
    double a, b, c;
    double D;
    double x1, x2;

    scanf("%lf", &a);
    scanf("%lf", &b);
    scanf("%lf", &c);

    D = b * b - 4 * a * c;

    x1 = (-b + sqrt(D)) / (2 * a);
    x2 = (-b - sqrt(D)) / (2 * a);

    printf("x1 = %lf", x1);
    printf("x2 = %lf", x2);
}
```



```
#include <stdio.h>
```

```
void main() {  
    int i = 1;  
    int a = 1, b = 2, c = 3, d = 4, e = 5, f = 6;  
    do {  
        printf("%d ", i);  
        if (a < b) {  
            for (b = d; b < f; b++) {  
                a = c;  
                while (a < f) {  
                    d += a;  
                    a++;  
                }  
                c = a;  
            }  
            e += d;  
        }  
    }  
}
```

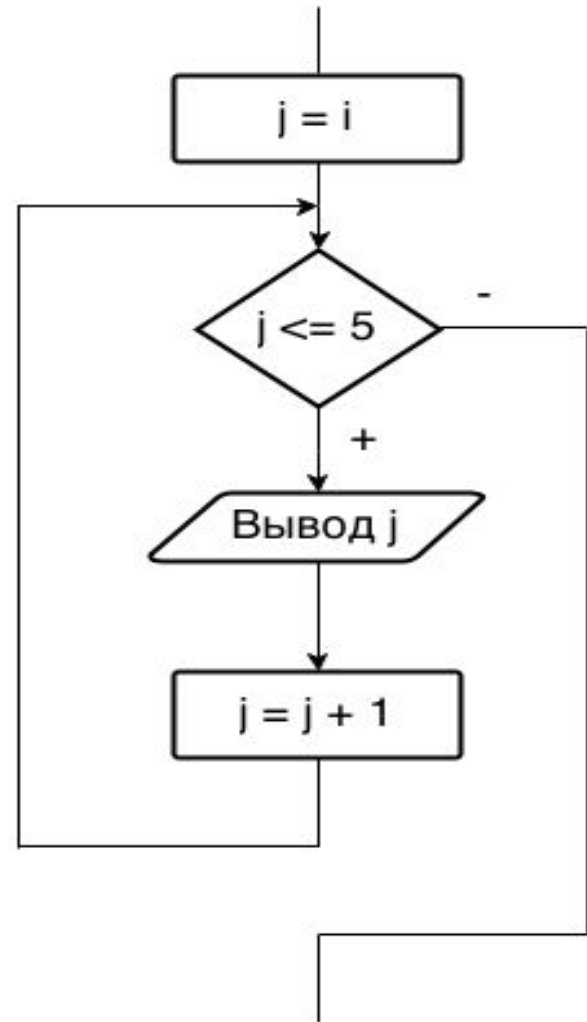


```
else {  
    for (f = e; f > a; f--) {  
        if (c < a) {  
            c = a;  
            d++;  
            break;  
        }  
        f += a;  
    }  
    }  
    i++;  
} while (i <= 5);  
  
printf("%d %d %d", d, e, f);  
}
```



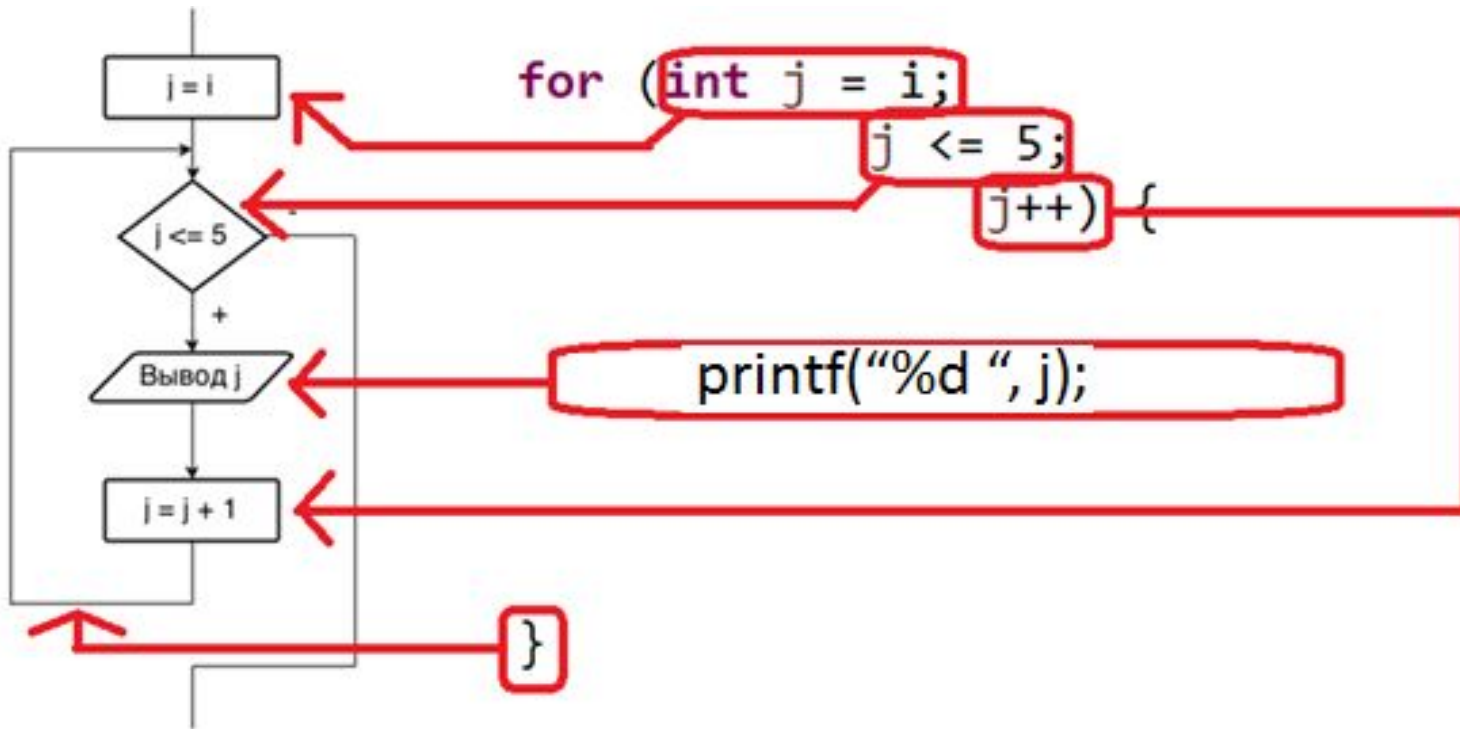
# Цикл for

```
int j = i; // инициализация счетчика цикла
while (j <= 5) { // условие продолжения цикла
    printf("%d ", j);
    j++; // изменение счетчика цикла
}
```



# Цикл for (2)

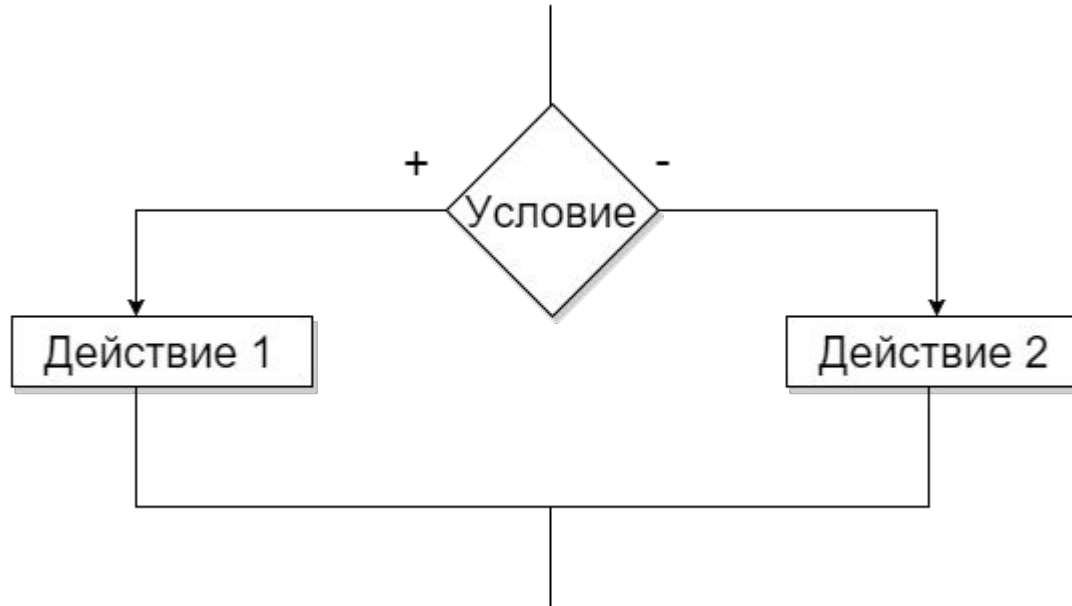
```
for (int j = i; j <= 5; j++) {  
    printf("%d ", j);  
}
```





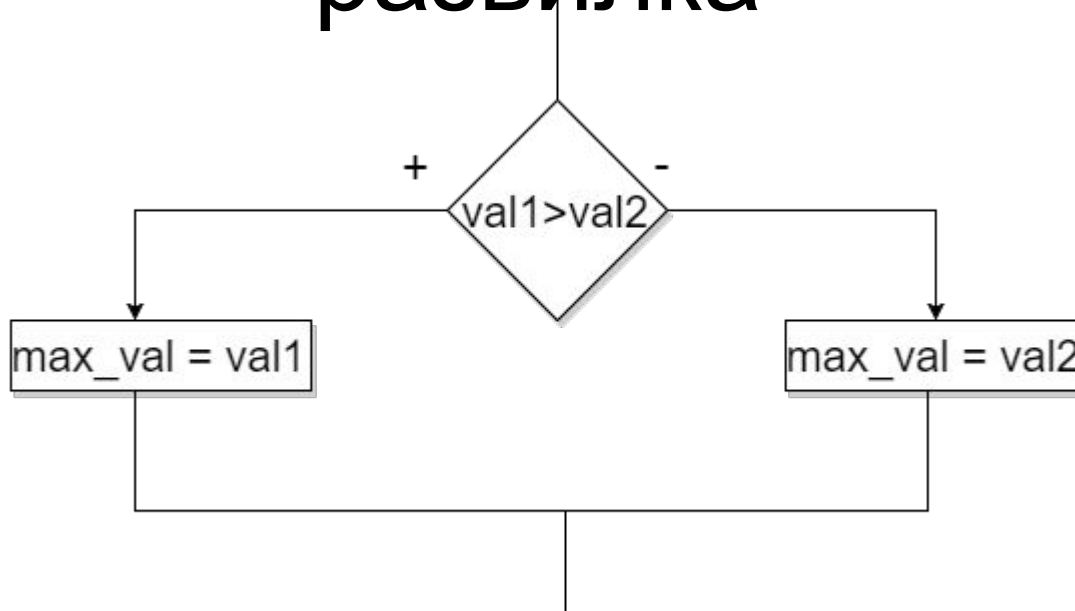


# Развилка



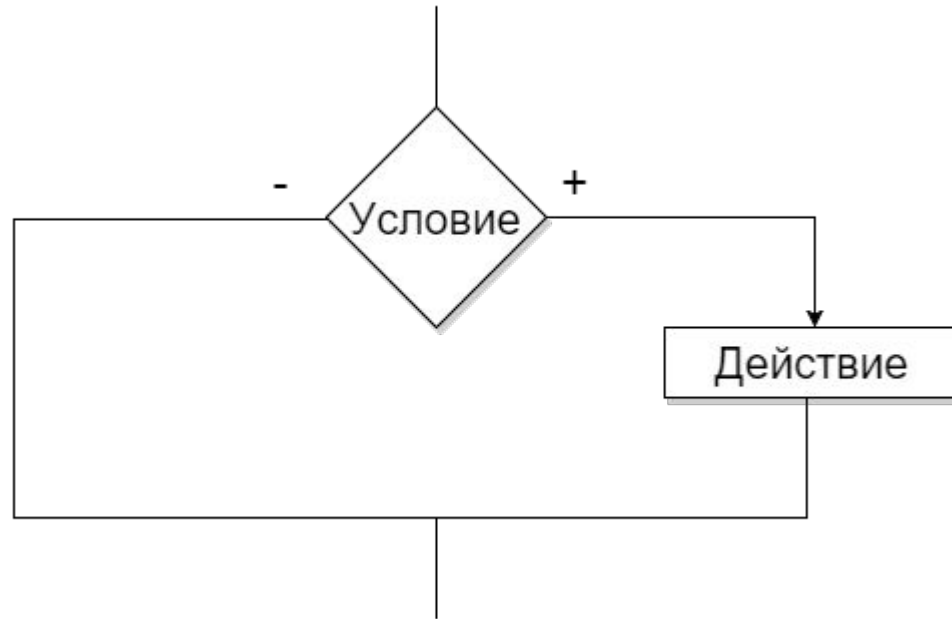
```
if (Условие)  
    Действие1;  
else  
    Действие2;
```

# Найти максимум - полная развилка



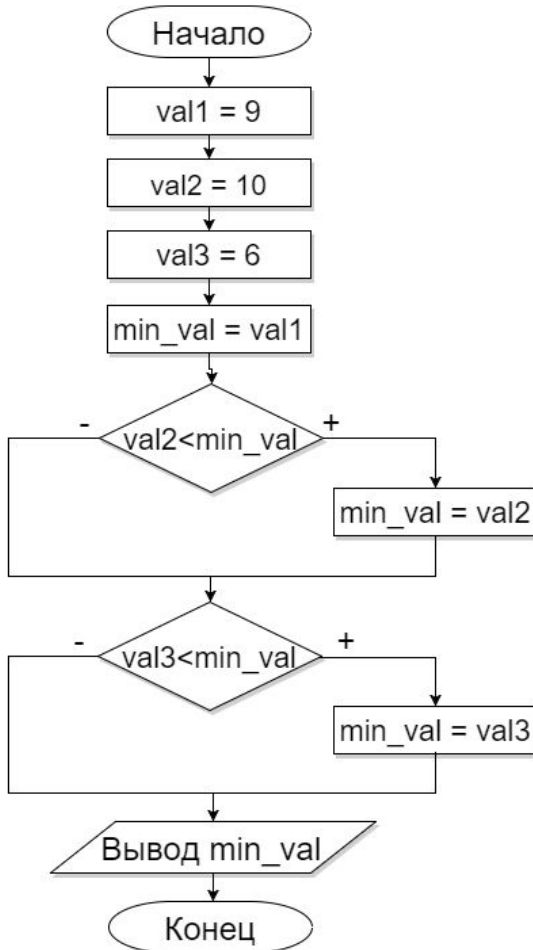
```
if (val1 > val2) {  
    max_val = val1;  
} else {  
    max_val = val2;  
}
```

# Усеченная развилка



```
if (Условие) {  
    Действие;  
}
```

# Минимум из 3 чисел



```
void main() {
```

```
int val1 = 9;
```

```
int val2 = 10;
```

```
int val3 = 6;
```

```
int min_val = val1; // берем за минимальный val1
```

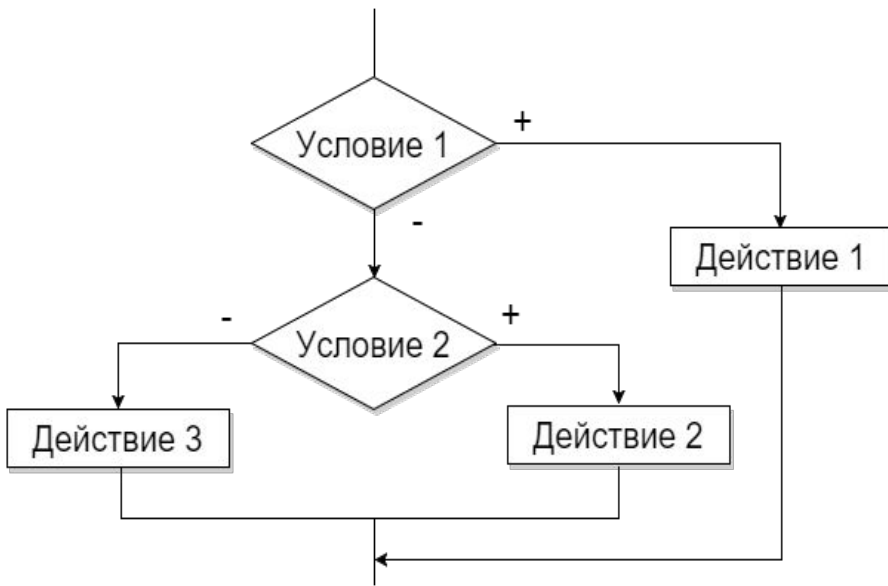
```
if (val2 < min_val) { // если второе меньше  
    min_val = val2; // то теперь минимальное val2  
}
```

```
if (val3 < min_val) { // если третье меньше  
    min_val = val3; // то теперь минимальное val3  
}
```

```
printf("min_val = %i", min_val);
```

```
}
```

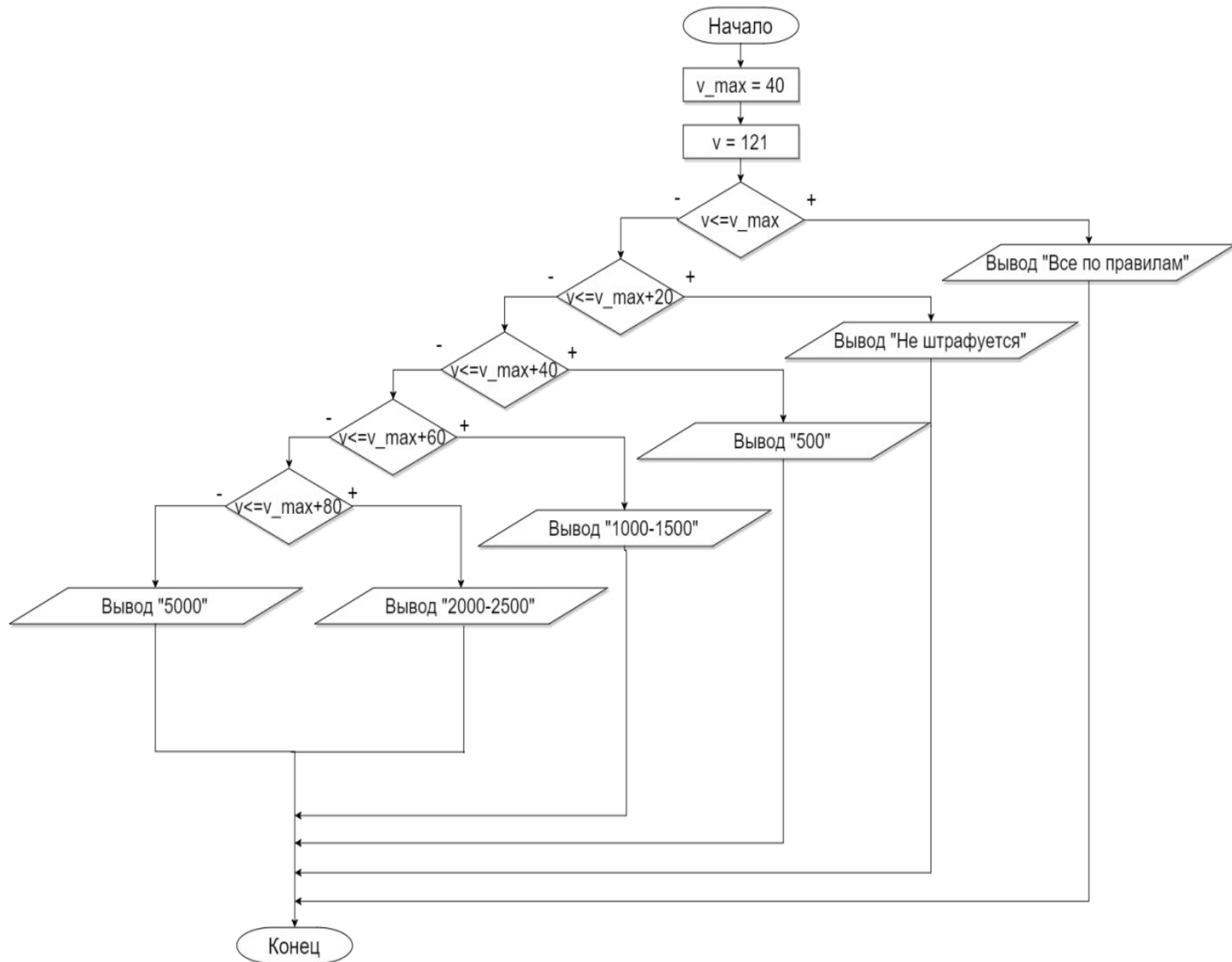
# Вложенные развилки



```
if (Условие 1) {  
    Действие 1  
} else {  
    if (Условие 2) {  
        Действие 2  
    } else {  
        Действие 3  
    }  
}
```

```
if (Условие 1) {  
    Действие 1  
} else if (Условие 2) {  
    Действие 2  
} else {  
    Действие 3  
}
```

# Штраф за превышение скорости



# Штраф за превышение скорости

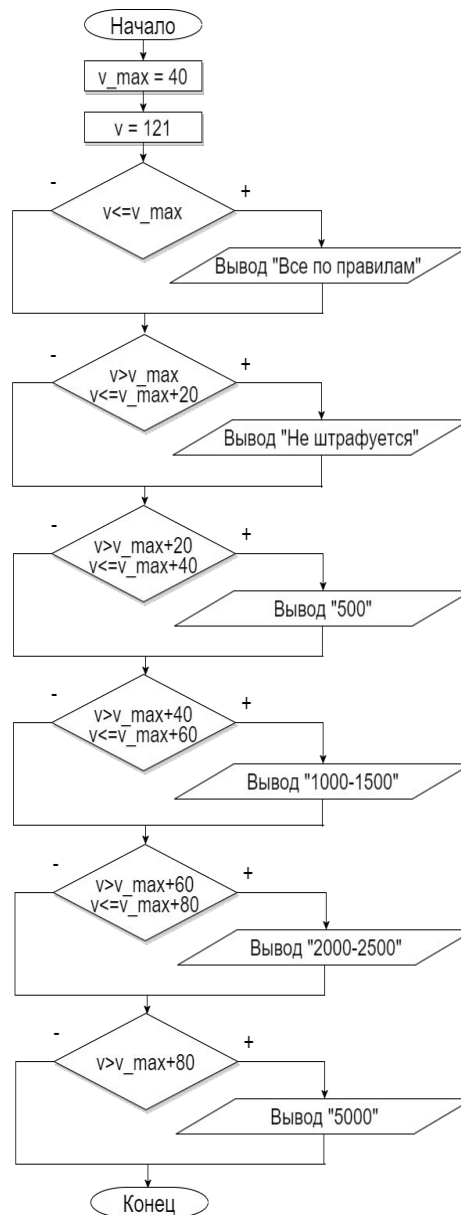
## – полная развилка

```
void main() {
    int v_max = 40;
    int v = 30;

    if (v <= v_max) {
        printf("All right!");
    } else if (v <= v_max + 20) {
        printf("No $$$");
    } else if (v <= v_max + 40) {
        printf("500");
    } else if (v <= v_max + 60) {
        printf("1000-1500");
    } else if (v <= v_max + 80) {
        printf("2000-2500");
    } else {
        printf("5000");
    }
}
```



# Штраф за превышение скорости



# Штраф за превышение скорости – усеченная развилка

```
void main() {
    int v_max = 40;
    int v = 70;

    if (v <= v_max) {
        printf("Все по правилам!");
    }
    if ((v > v_max) && (v <= v_max + 20)) {
        printf("не штрафуются");
    }
    if ((v > v_max + 20) && (v <= v_max + 40)) {
        printf("500");
    }
    if ((v > v_max + 40) && (v <= v_max + 60)) {
        printf("1000-1500");
    }
    if ((v > v_max + 60) && (v <= v_max + 80)) {
        printf("2000-2500");
    }
    if (v > v_max + 80) {
        printf("5000");
    }
}
```

# Логические операции

Оператор	Описание
&&	Логическое И (AND)
	Логическое ИЛИ (OR)
!	Логическое унарное НЕ (NOT)

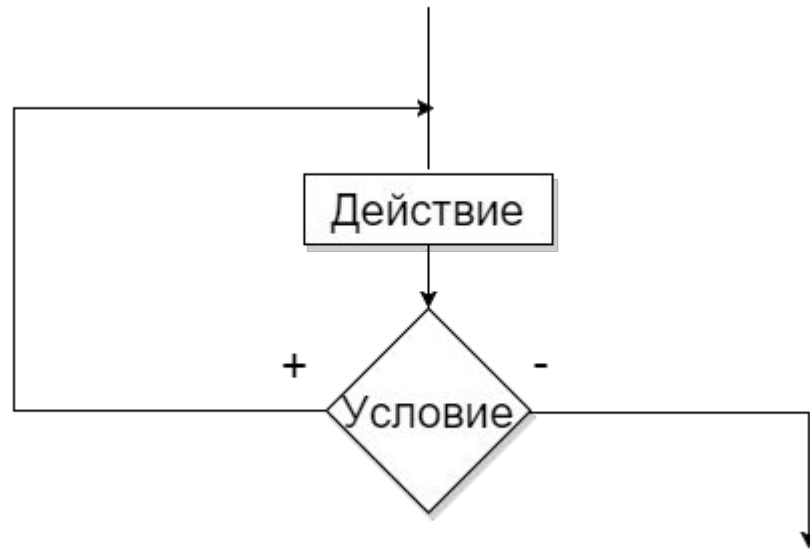
A	!A
0	1
1	0

A	B	A && B	A    B
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	1

```
if (time < 7.00 || day >= 6) rest();
```

```
if (!closed && money > 1000) eat();
```

# Цикл с постусловием do while

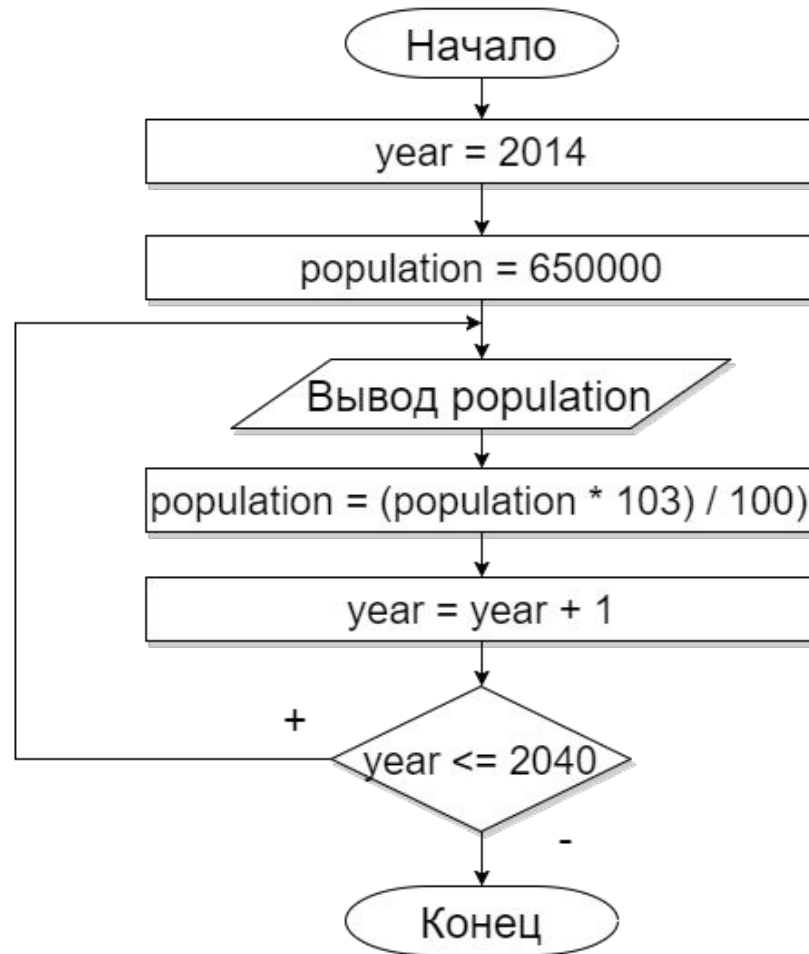


```
do {  
    Действие;  
} while (Условие);
```

# Пример для цикла do while

Население города увеличивается на 3% каждый год. В 2014 году население города составляло 650 000 человек. Напишите программу, которая выведет на экран предсказываемую численность населения города в каждом году, вплоть до 2040.

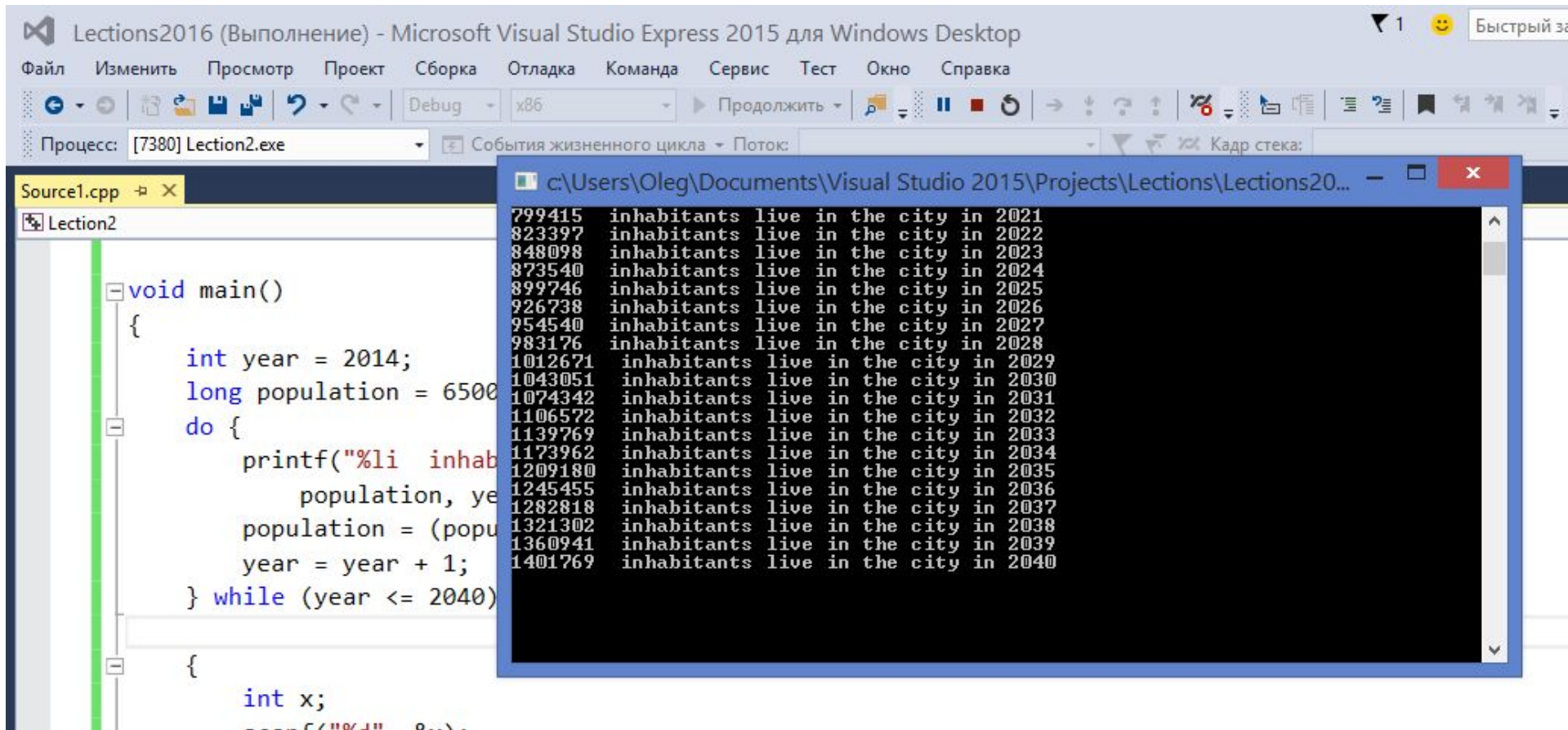
# Блок-схема



# Программа

```
void main() {  
    int year = 2014;  
    long population = 650000;  
    do {  
        printf("%li inhabitants live in the city in %i\n",  
            population, year);  
        population = (population * 103) / 100;  
        year = year + 1;  
    } while (year <= 2040);  
}
```

# Программа в работе



Lections2016 (Выполнение) - Microsoft Visual Studio Express 2015 для Windows Desktop

Файл Изменить Просмотр Проект Сборка Отладка Команда Сервис Тест Окно Справка

Процесс: [7380] Lection2.exe

```
Source1.cpp
Lection2

void main()
{
    int year = 2014;
    long population = 6500
    do {
        printf("%li inhab
            population, ye
            population = (popu
            year = year + 1;
    } while (year <= 2040)

    {
        int x;
        ---- C/"%d" o...
```

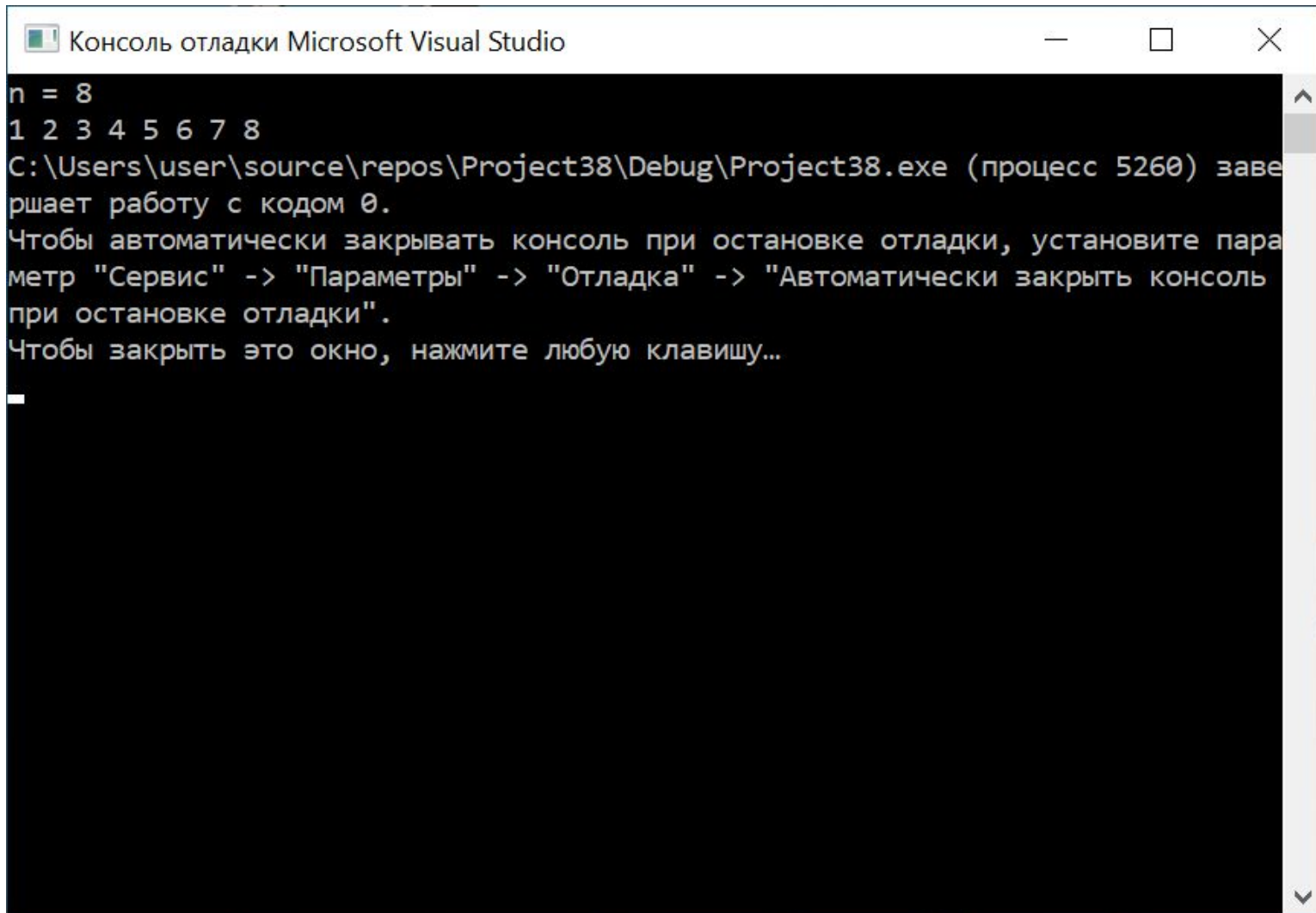
```
799415 inhabitants live in the city in 2021
823397 inhabitants live in the city in 2022
848098 inhabitants live in the city in 2023
873540 inhabitants live in the city in 2024
899746 inhabitants live in the city in 2025
926738 inhabitants live in the city in 2026
954540 inhabitants live in the city in 2027
983176 inhabitants live in the city in 2028
1012671 inhabitants live in the city in 2029
1043051 inhabitants live in the city in 2030
1074342 inhabitants live in the city in 2031
1106572 inhabitants live in the city in 2032
1139769 inhabitants live in the city in 2033
1173962 inhabitants live in the city in 2034
1209180 inhabitants live in the city in 2035
1245455 inhabitants live in the city in 2036
1282818 inhabitants live in the city in 2037
1321302 inhabitants live in the city in 2038
1360941 inhabitants live in the city in 2039
1401769 inhabitants live in the city in 2040
```



# Задача 1. Ряд натуральных чисел

Вводится N.

Нужно вывести натуральные числа от 1 до N (включительно).



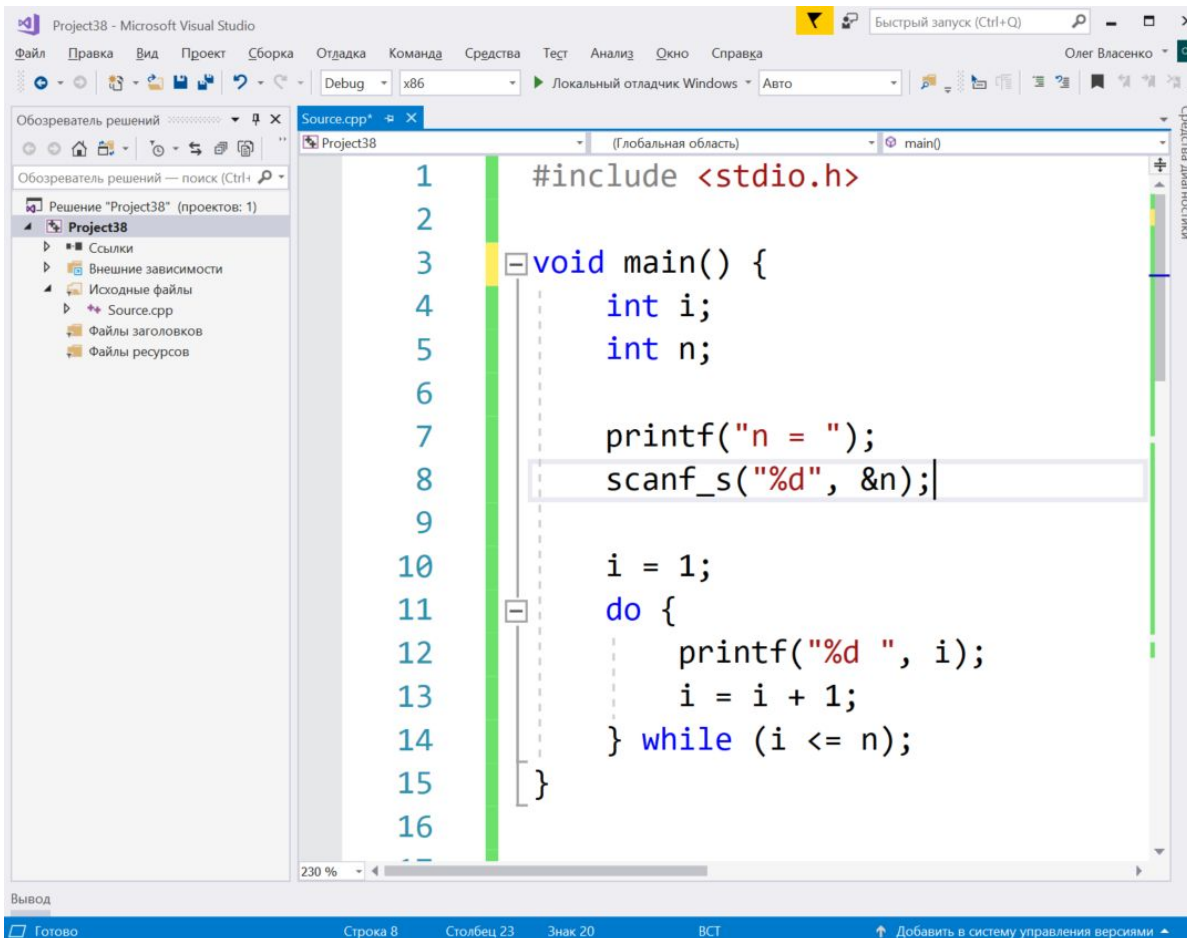
```
Консоль отладки Microsoft Visual Studio
n = 8
1 2 3 4 5 6 7 8
C:\Users\user\source\repos\Project38\Debug\Project38.exe (процесс 5260) завершает работу с кодом 0.
Чтобы автоматически закрывать консоль при остановке отладки, установите параметр "Сервис" -> "Параметры" -> "Отладка" -> "Автоматически закрыть консоль при остановке отладки".
Чтобы закрыть это окно, нажмите любую клавишу...
```

# Задача 1. Ряд натуральных чисел

Вводится N.

Нужно вывести натуральные числа от 1 до N (включительно).

(Ниже – программа целиком)



```
1  #include <stdio.h>
2
3  void main() {
4      int i;
5      int n;
6
7      printf("n = ");
8      scanf_s("%d", &n);
9
10     i = 1;
11     do {
12         printf("%d ", i);
13         i = i + 1;
14     } while (i <= n);
15 }
16
```

The screenshot shows the Microsoft Visual Studio IDE with a C++ project named 'Project38'. The main window displays the source code for 'Source.cpp'. The code includes the standard input/output header, defines the main function, and uses a do-while loop to print natural numbers from 1 to N. The user interface includes a menu bar, a toolbar, a solution explorer on the left, and a status bar at the bottom.



















# Задача 1. Ряд натуральных чисел – трассировка(8)

```
printf("n = ");
scanf_s("%d", &n);

i = 1;
do {
    printf("%d ", i);
    i = i + 1;
} while (i <= n);
```

i	n	i<=n	Вывод	Ввод
1	4		"n = "	4 <Enter>
2		2 <= 4 +	"1 "	
			"2 "	

# Задача 1. Ряд натуральных чисел – трассировка(9)

```
printf("n = ");
scanf_s("%d", &n);

i = 1;
do {
    printf("%d ", i);
    i = i + 1;
} while (i <= n);
```

i	n	i<=n	Вывод	Ввод
1	4		"n = "	4 <Enter>
2		2 <= 4 +	"1 "	
3			"2 "	

# Задача 1. Ряд натуральных чисел – трассировка(10)

```
printf("n = ");
scanf_s("%d", &n);

i = 1;
do {
    printf("%d ", i);
    i = i + 1;
} while (i <= n);
```

i	n	i<=n	Вывод	Ввод
1	4		"n = "	4 <Enter>
2		2 <= 4 +	"1 "	
3		3 <= 4 +	"2 "	

# Задача 1. Ряд натуральных чисел – трассировка(11)

```
printf("n = ");
scanf_s("%d", &n);

i = 1;
do {
    printf("%d ", i);
    i = i + 1;
} while (i <= n);
```

i	n	i<=n	Вывод	Ввод
1	4		"n = "	4 <Enter>
2		2 <= 4 +	"1 "	
3		3 <= 4 +	"2 "	
			"3 "	

# Задача 1. Ряд натуральных чисел – трассировка(12)

```
printf("n = ");  
scanf_s("%d", &n);  
  
i = 1;  
do {  
    printf("%d ", i);  
    i = i + 1;  
} while (i <= n);
```

i	n	i<=n	Вывод	Ввод
1	4		"n = "	4 <Enter>
2		2 <= 4 +	"1 "	
3		3 <= 4 +	"2 "	
4			"3 "	

# Задача 1. Ряд натуральных чисел – трассировка(13)

```
printf("n = ");  
scanf_s("%d", &n);  
  
i = 1;  
do {  
    printf("%d ", i);  
    i = i + 1;  
} while (i <= n);
```

i	n	i<=n	Вывод	Ввод
1	4		"n = "	4 <Enter>
2		2 <= 4 +	"1 "	
3		3 <= 4 +	"2 "	
4		4 <= 4 +	"3 "	



# Задача 1. Ряд натуральных чисел – трассировка(14)

```
printf("n = ");  
scanf_s("%d", &n);  
  
i = 1;  
do {  
    printf("%d ", i);  
    i = i + 1;  
} while (i <= n);
```

i	n	i<=n	Вывод	Ввод
1	4		"n = "	4 <Enter>
2		2 <= 4 +	"1 "	
3		3 <= 4 +	"2 "	
4		4 <= 4 +	"3 "	
			"4 "	

# Задача 1. Ряд натуральных чисел – трассировка(15)

```
printf("n = ");
scanf_s("%d", &n);

i = 1;
do {
    printf("%d ", i);
    i = i + 1;
} while (i <= n);
```

i	n	i<=n	Вывод	Ввод
1	4		"n = "	4 <Enter>
2		2 <= 4 +	"1 "	
3		3 <= 4 +	"2 "	
4		4 <= 4 +	"3 "	
5			"4 "	

# Задача 1. Ряд натуральных чисел – трассировка(16)

```
printf("n = ");  
scanf_s("%d", &n);  
  
i = 1;  
do {  
    printf("%d ", i);  
    i = i + 1;  
} while (i <= n);
```

i	n	i<=n	Вывод	Ввод
1	4		"n = "	4 <Enter>
2		2 <= 4 +	"1 "	
3		3 <= 4 +	"2 "	
4		4 <= 4 +	"3 "	
5		5 <= 4 -	"4 "	

# Задача 1. Ряд натуральных чисел

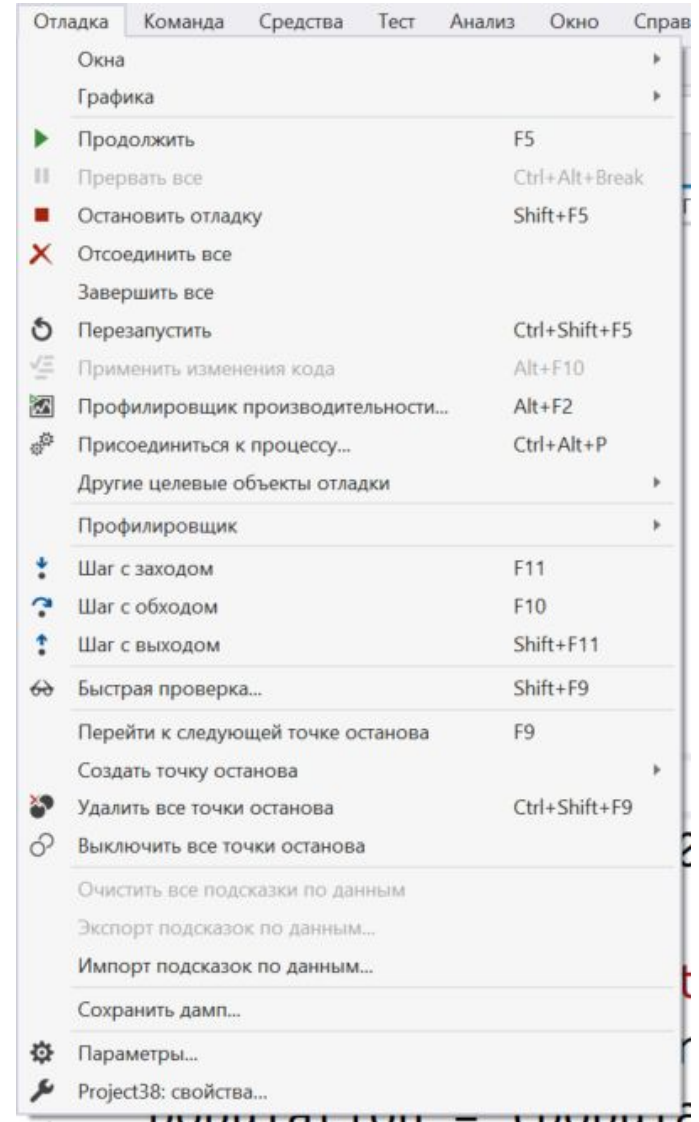
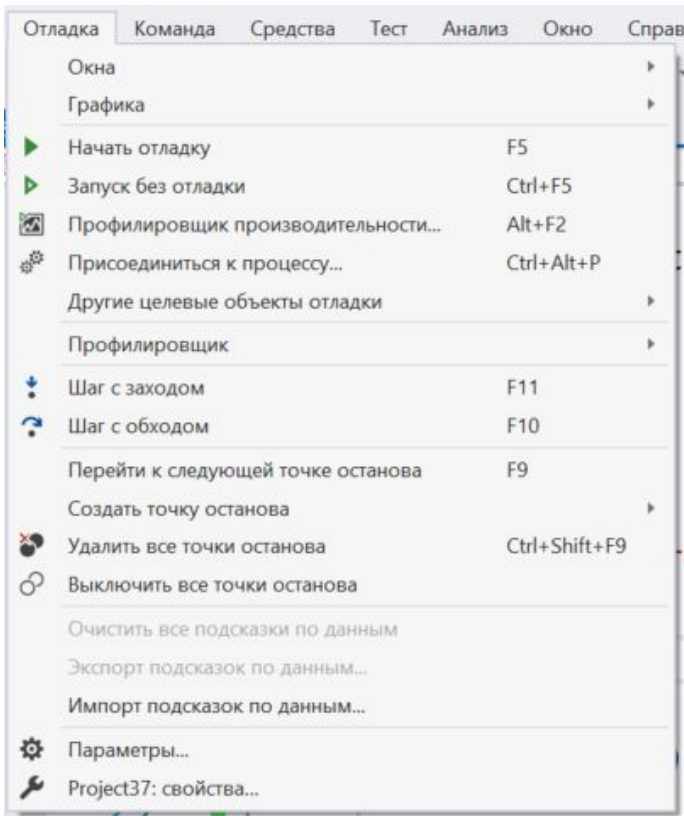
Вводится  $N$ .

Нужно вывести натуральные числа от 1 до  $N$  (включительно).

**Нарисуйте блок схему к Задаче 1.**

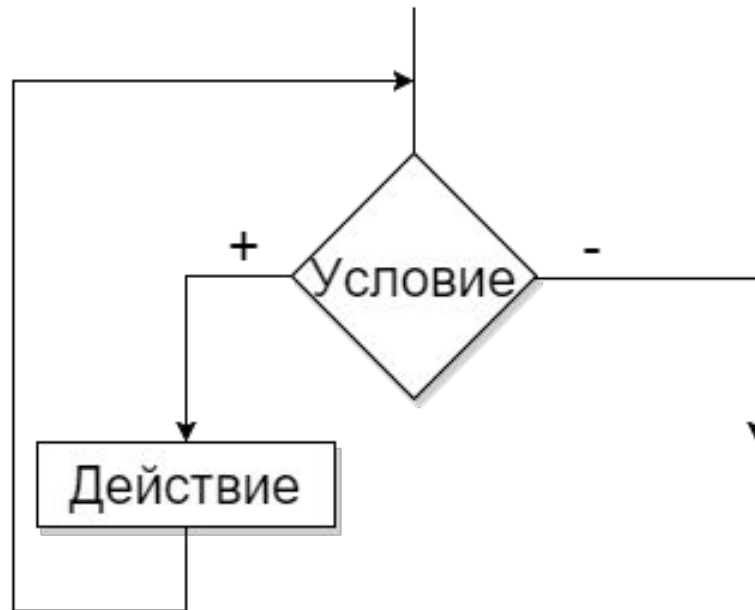
# Отладка программы

Можно использовать горячие клавиши:



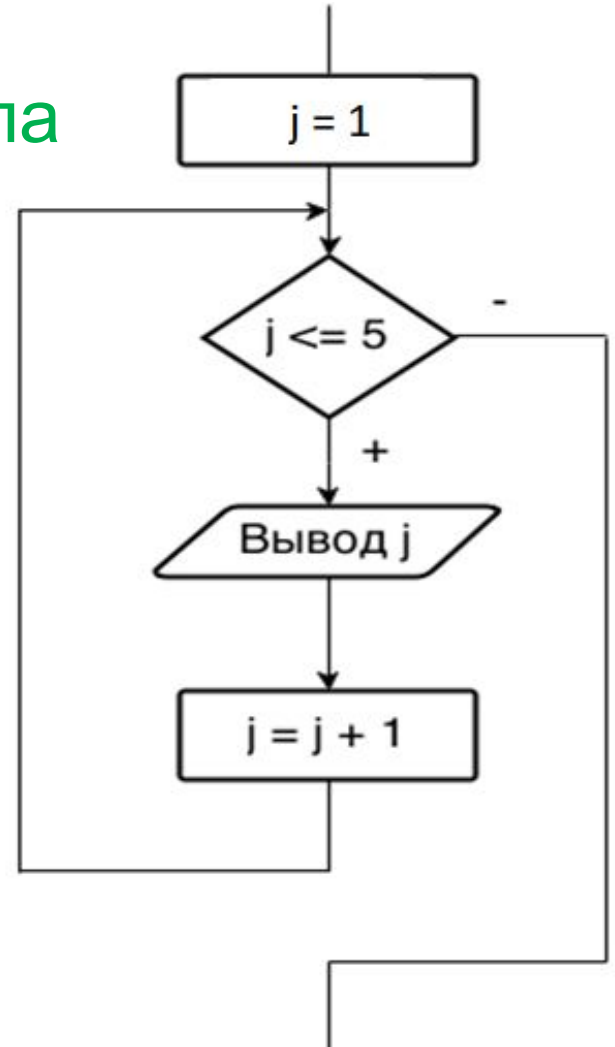
# Цикл с предусловием while

```
while (Условие) {  
    Действие;  
}
```



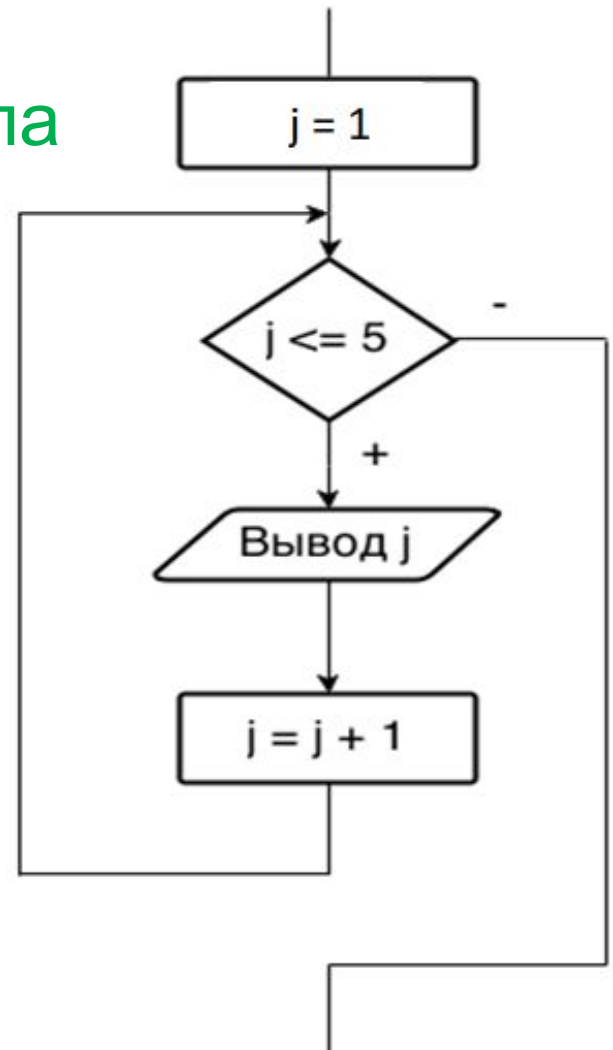
# Пример кода с while

```
int j = 1; // инициализация счетчика цикла
while (j <= 5) { // условие продолжения цикла
    printf("%d ", j);
    j++; // изменение счетчика цикла
}
```



# Пример кода с while - трассировка

```
int j = 1; // инициализация счетчика цикла
while (j <= 5) { // условие продолжения цикла
    printf("%d ", j);
    j++; // изменение счетчика цикла
}
```







# Домашнее задание

(желательное)

1. В режиме пошаговой отладки («дебага») выполнить несколько (3-5) циклических фрагментов кода.
2. В режиме пошаговой отладки выполнить код, содержащий развилки.
3. \* В режиме пошаговой отладки выполнить код, содержащий функции
4. \*\*\* В режиме пошаговой отладки выполнить код, содержащий **рекурсивные** функции

# Источники информации

- Полный справочник по C  
[https://cpp.com.ru/shildt\\_spr\\_po\\_c/index.html](https://cpp.com.ru/shildt_spr_po_c/index.html)
- Ч. Петзолд Программирование для Windows® 95  
<http://softtime.ru/files/books/Petzold1.pdf>
- «Программирование на C и C++» - полезные книги -  
<https://cpp.com.ru/>
- <http://rstdn.ru/>
- <https://msdn.microsoft.com/ru-ru/default.aspx>
- <http://habrahabr.ru/>
- <https://www.google.ru/>