

Инструкции языка C

Общая схема инструкций в С

Инструкция может быть:

Объявление – описание свойств переменной, функции или пользовательского типа;

Составная – группа инструкций, заключенная в {} или блок кода;

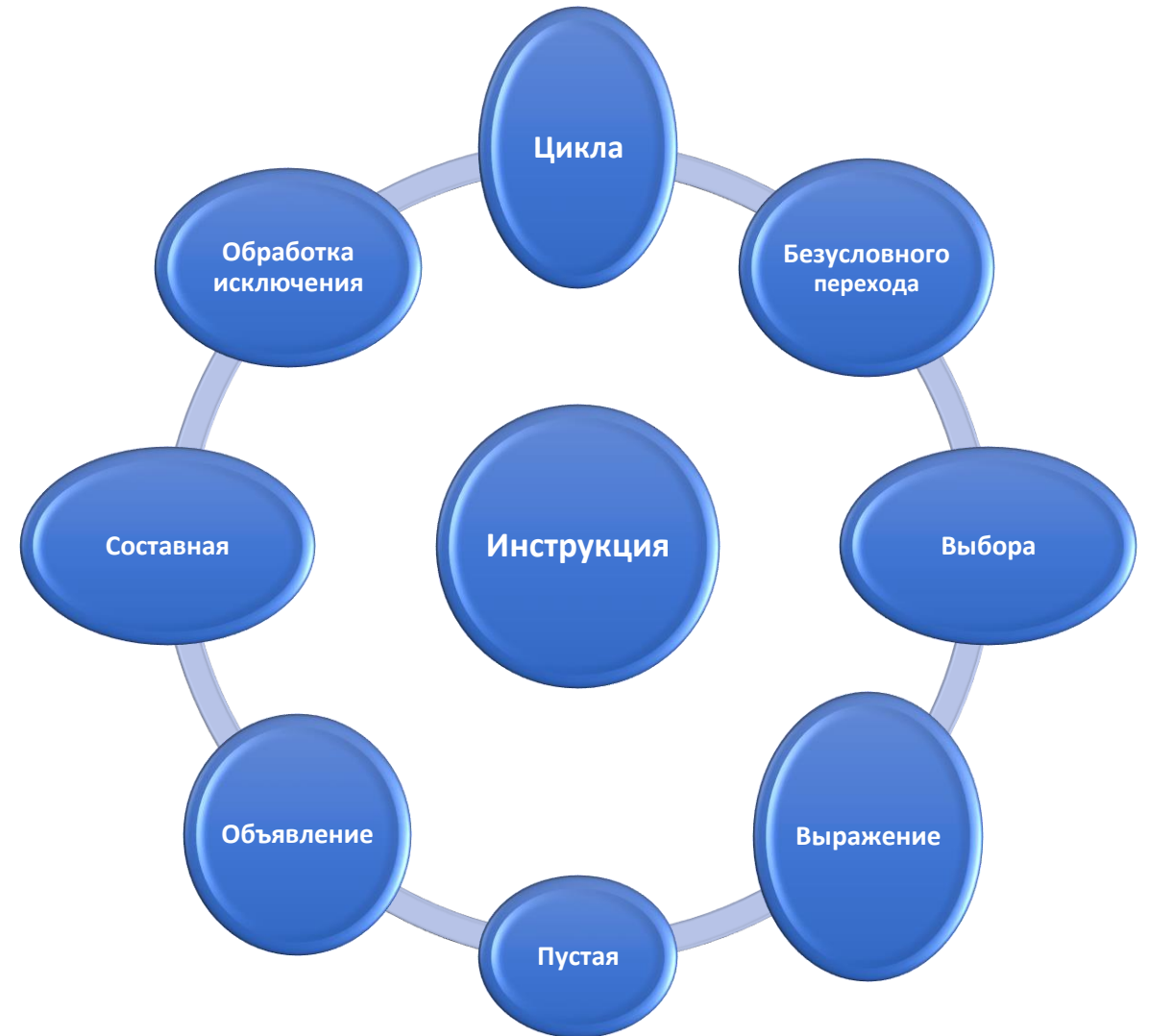
Выражение – содержит последовательность операторов с операндами;

Пустая – состоит только из разделителя «;»;

Выбора - конструкция if.. else или switch;

Цикла – конструкция while, for, do.. while;

Безусловного перехода – break, continue, return;



Инструкции выбора (if ... else)

```
if (Условие)
{
    БлокОпераций1;
}
```



Оператор ветвления **if** в зависимости от условия позволяет выбрать одно из двух возможных продолжений программы.

```
if (Условие)
{
    БлокОпераций1;
}
else
{
    БлокОпераций2;
}
```



Условием может быть как строгое равенство так и логическое выражение или отношение.

Примеры оператора if ...else

Связывание стейтментов if

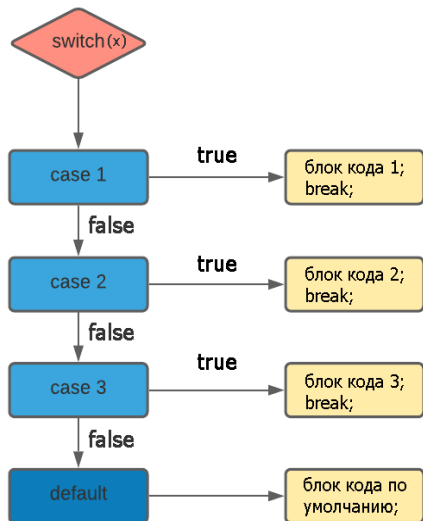
```
#include <stdio.h>
int main()
{
    int a = 4;
    if (a > 15)
        printf(" is greater than 15\n");
    else if (a < 15)
        printf(" is less than 15\n");
    else
        printf(" is exactly 15\n");
    return 0;
}
```

Вложенные ветвления if/else

```
#include <stdio.h>
int main()
{
    int a = 17;
    if (a > 15)
    {
        if (a < 25)
            printf(" is between 15 and 25\n");
        else // относится к внутреннему оператору if
            printf(" is greater than or equal to 25\n");
    }
    return 0;
}
```

Инструкции выбора (switch)

- **switch** отличается от **if** тем, что он может выполнять только операции проверки строгого равенства, в то время как **if** может вычислять логические выражения и отношения.
- Не может быть двух констант в одном операторе **switch**, имеющих одинаковые значения. Конечно, оператор **switch**, включающий в себя другой оператор **switch**, может содержать аналогичные константы.
- Если в операторе **switch** используются символьные константы, они автоматически преобразуются к целочисленным значениям.



Оператор switch предназначен для организации выбора из множества различных вариантов.

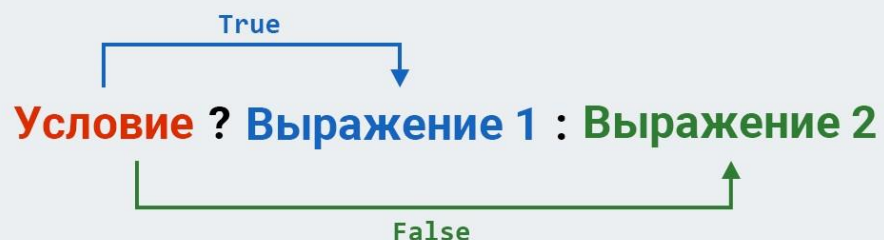
```
switch ( <переменная> ) {  
  case значение1:  
    Выполнить если <переменная> == значение1  
    break;  
  case значение2:  
    Выполнить если <переменная> == значение2  
    break;  
  ...  
  default:  
    выполнить, если ни один вариант не подошел  
    break;  
}
```

Пример использования оператора switch

Использование переменных внутри блока switch ... case

```
#include<stdio.h>
int main() {
    int num = 3;
    switch (num) {
        case 1:
            int z; // ок, объявление разрешено
            z = 5; // ок, операция присваивания разрешена
            break;
        case 2:
            z = 6; // ок, переменная z была объявлена выше, поэтому мы можем использовать её здесь
            break;
        case 3:
            int c = 4; // нельзя, вы не можете инициализировать переменные внутри case
            break;
        default:
            printf("default case");
            break;
    }
    return 0;
}
```

Инструкции выбора (тернарный оператор)



Тернарный оператор – сокращенная форма записи простейших блоков с **if**.

```
#include <stdio.h>
int main()
{
    int n =5;
    int m = 6;
    int res = (n > m) ? n-m : m-n; // res = 1;
    printf("res: %d", res);
    return 0;
}
```

Циклы

Для эффективной организации многократно повторяющихся действий служат циклические конструкции.

Любой цикл содержит:

- тело цикла (инструкции, выполняющаяся многократно);
- начальные установки;
- изменения параметра цикла;
- проверку условия продолжения выполнения цикла.

```
while (условие)
{
    блок инструкций
}
```

```
do
{
    Блок инструкций
}
while (условие);
```

```
for (счетчик = значение; счетчик < значение; шаг) {
    тело цикла;
}
```

- Проверка условия выполняется на каждой итерации (итерация – это один проход цикла). Если условие продолжения цикла не выполняется, то он завершается.
- Для принудительного завершения текущей итерации, или цикла в целом используются инструкции передачи управления (**break**, **continue**, **return**);
- Циклы могут быть вложенными;

Цикл с предусловием



Цикл с постусловием



Примеры использования циклов

```
#include <stdio.h>

int main()
{
    int count = 0;
    while (count < 10)
    {
        printf(" count %d\n", ++count);
    }
    printf("done!\n");
    return 0;
}
```

```
#include <stdio.h>


int main()
{
    int i = 0;
    do {
        printf("%d\n", i);
        i++;
    } while(i < 10);
    printf("done!\n");
    return 0;
}
```

```
#include <stdio.h>

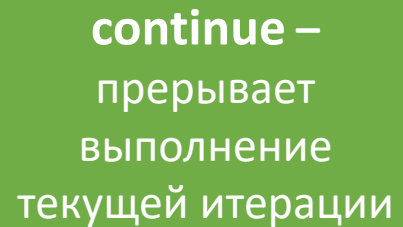
int main()
{
    int aaa, bbb;
    for (aaa = 0, bbb = 9; aaa < 10; ++aaa, --bbb)
        printf(" aaa =%d, bbb = %d\n", aaa, bbb);

    return 0;
}
```

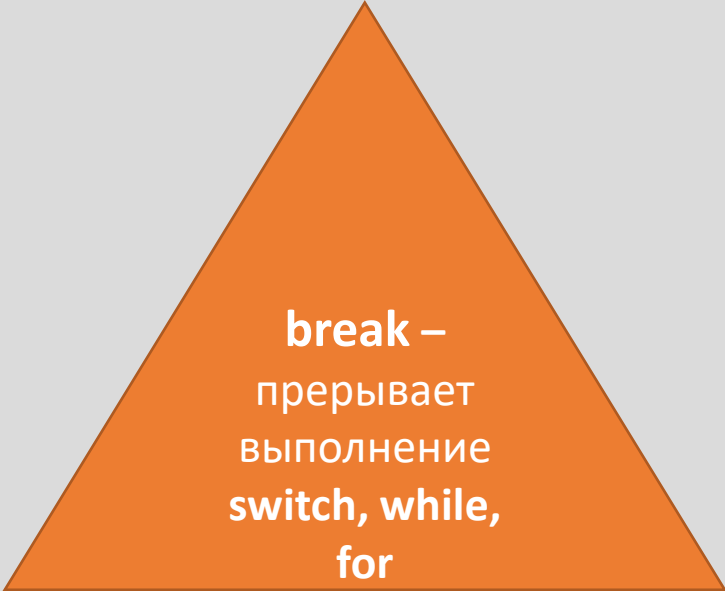
Инструкции безусловного перехода



return –
прерывает
выполнение
текущей ф-ции



continue –
прерывает
выполнение
текущей итерации



break –
прерывает
выполнение
switch, while,
for

Примеры использования операторов безусловного перехода

```
#include <stdio.h>
int main(void)
{
    int t;
    for(t = 0; t < 100; t++)
    {
        printf("%d ", t);
        if (t == 10)
            break;
    }
    return 0;
}
```

```
#include <stdlib.h>
int main(void)
{
    int x;
    do
    {
        scanf("%d", &x);
        if( x < 0 )
            continue;
        printf("%d ", x);
    } while(x != 100);
    return 0;
}
```

```
#include <stdlib.h>
int main(void)
{
    int x;
    for(int t = 0; t < 100; ++t)
    {
        scanf("%d", &x);
        if (x<0)
            continue;
        printf ("%d ", x);
    }
    return 0;
}
```

На сегодня всё!