Лекция 4. Логические операции. Массивы. Строки.

Евгений Линский

Логические операции. И (and).

Логические выражения можно объединять.
Операция “и” (в C — &&): true только если оба операнда true.
(a && b: && — операция, a и b операнды).

a	b	a && b
true	true	true
true	false	false
false	true	false
false	false	false

Подойдет только нечетное число больше 10:

```
int b = 5;
if ((b > 10) && (b % 2 == 1)) {
  printf("%d\n", b);
}
```

Подходят: 13, 43, 101.
Не подходят: 1, 2, 4, 3, 7, 8, 1000.

Логические операции. Или (or).

Операция “или” (в C — ||): true если хотя бы один операнд true.

a	b	a || b
true	true	true
true	false	true
false	true	true
false	false	false

Подойдет или нечетное число или число больше 10:

```
int b = 5;
if ((b > 10) || (b % 2 == 1)) {
  printf("%d\n", b);
}
```

Подходят: 1, 13, 7, 3, 43, 101, 1000.
Не подходят: 2, 4, 8.

Примеры.

Лучше ставить скобки, чтобы облегчить чтение программы.
Подойдет число $b \in [-10; 0) \cup b \in (100, 200)$

```
int b = 5;
if (((b >= -10) && (b < 0)) || ((b > 100) && (b < 200))) {
  printf("%d\n", b);
}
```

Выполнять, пока не будет введено нечетное число больше 10.

```
int b = 0;
while( (b <= 10) || (b % 2 == 0) ) {
  scanf("%d", &b);
}
```

Логические операции. Операция (not).

Операция “не” (в C — !): меняет true на false, а false на true.

a	!a
true	false
false	true

Подойдет только четное число.

```
int b = 5;
if (!(b % 2 == 1)) {
  printf("%d\n", b);
}
```

Примеры.

Выполнять, пока не будет введено нечетное число больше 10.
Без not:

```
int b = 0;
while( (b <= 10) || (b % 2 == 0) ) {
  scanf("%d", &b);
}
```

С not:

```
int b = 0;
while( !((b > 10) && (b % 2 == 1)) ) {
  scanf("%d", &b);
}
```

Логические и побитовые операции.

Не путать:
a && b, a || b (логические операции)
и
a & b, a | b (побитовые операции; разберем в другой лекции)

Массивы

Массив — непрерывный участок памяти; к ячейкам обращаются с помощью индекса (номера).
Одномерные:

```
int array[10]; // размер 10*sizeof(int)
//Инициализация:
int array[10] = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9};
int array[10] = {0}; // обнулить
int array[] = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9};
```

Двумерные:

```
int m[10][10];
int m[2][2] = { {1,2} , {3,4} };
```

Массивы

```
int array[10] = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9};
int index = 3;
//присвоить 3-ий элемент массива переменной a
int a = array[index];
//записать в 3-ий элеимент массива число 5
array[index] = 5;
//ошибки времени выполенния
array[10] = 1; array[-1] = 1;
```

“Труднонаходимые” ошибки

1. Выход за пределы массива не контролируется компилятором
2. Исходы:
   - программа корректно отработала (не задели данные)
   - программа некорректно (зависла, неверный ответ) отработала (задели данные)
   - ОС аварийно завершила программу (задели чужие данные)
3. Исход зависит от того, что (данные, другие программы) в памяти в данный момент

Примеры.

Вывести массив.

```
int arr[] = {1, 13, 41, 55, 63, 71, 88};
int i = 0;
for(i = 0; i < 7; i ++) {
  printf("%d ", a[i]);
}
```

Ввести массив.

```
int arr[7] = {0};
int i = 0
for(i = 0; i < 7; i ++) {
  scanf("%d ", &a[i]);
}
```

Примеры.

Найти сумму элементов массива.

```
int arr[] = {112, 13, 41, 5, 1100, 600, 750};
int sum = 0;
for(int i = 0; i < 7; i ++) {
  sum = sum + arr[i];
}
```

Найти минимальное число в массиве.

```
int arr[] = {112, 13, 41, 5, 1100, 600, 750};
int min = a[0];
for(i = 0; i < 7; i++) {
  if (a[i] <= min)
    min = a[i];
}
```

Примеры.

Заполнить двумерный массив таблицей умножения.

```
int arr[10][10];
for(int i = 0; i < 10; i++) {
  for(int j = 0; j < 10; j++) {
    arr[i][j] = i * j;
  }

}
```

Заполнить диагональ числом 42.

```
int arr[5][5];
int min = a[0];
for(i = 0; i < 5; i++) {
  a[i][i] = 42;
}
```

Символы

```
// присвоить в c1 код (106) символа 'F'
char c1 = 'A';
// можно и так
char c2 = 106;
```

Перевод из символа в цифру (коды символов с цифрами расположен последлвательно)

```
// в c1 записано число 71
char c1 = '9';
// в i1 записано число 9
int i1 = c1 - '0';
```

Строки

```
//для типа char:
char array[] = {0, 1, 2, 3, 4, 5, 6, 7, 8, 9};
char array[] = {'H', 'e', 'l', 'l', 'o'};
// размер равен 6 * sizeof(char), компилятор в конец добавит 0
char array[] = "Hello";
```

- В C строки заканчиваются числом 0.
- На это расчитаны все функции из стандартной библиотеки.

Примеры

char s[] = "My name is Evgeny.";
Вычислить длину строки:

```
int len = 0;
while (s[len] != 0)
  len++;
```

Подсчитать число пробелов в строке:

```
int i = 0;
int count = 0;
while (s[i] != 0) {
  if (s[i] == ' ')
    count++;
  i++;
}
```

Примеры

char s[] = "My name is Evgeny. ";
Подсчитать число слов строке:

```
int in_word = 0, i = 0, word_count = 0;
while (s[i] != 0) {
  if (s[i] != ' ')
    in_word = 1;
  else {
    if (in_word == 1)
      word_count++;
    in_word = 0;
  }
  i++;
}
if (in_word == 1)
  word_count++;
```

Слово в этой программе — последовательность любых символов без пробелов.