Лекция 3. Управляющие конструкции: ветвление и цикл.

Евгений Линский

Выражения

```
int a = 3;
int b = 5;
int c = 8;
int d = 0;
```

Арифметическое выражение: результатом вычсиления является число.

```
(b * b - 4 * a * c) / (2 * a)
```

Логическое выражение: результатом вычисление является истина (true) или ложь (false).

```
a > b   // false
a >= b  // false
a < b   // true
a <= b  // true
a == b  // false (проверка на равенство)
a != b  // true (проверка на неравенство)
```

Ветвление if - I

Простейший случай.

```
if (логическое выражение) {
  код
}
```

Код выполнится, только если логическое выражение вычислится (говорят “вернет”) как true.

```
int a = 4;
if (a % 2 == 1) {
  printf("%d is odd number\n", a);
}
```

Задача: вывести фразу, если число нечетное
NB: перед вложенными инструкциями принято ставить отступ для читаемости программы!

Ветвление if - II

Задана альтернативная ветка.

```
if (логическое выражение) {
  код
}
else {
  код
}
```

Код блока else выполнится, если логическое выражение вернет false.

```
int a = 0;
scanf("%d", &a);
if (a % 2 == 1) {
  printf("%d is odd number\n", a);
}
else {
  printf("%d is even number\n", a);
}
```

Задача: вывести одну фразу, если число нечетное, и другую фразу – если число четное.

Ветвление if - III

Заданы несколько альтернативных веток с проверкой дополнительных условий.

```
if (логическое выражение) {
  код
}
else if (логическое выражение) {
  код
}
// блоков else if может быть несколько
else if (логическое выражение) {
  код
}
else {
  код
}
```

Выполнится только одна из веток (если ни одна ветка if/else if не вернет true, то else).

Ветвление if - III

```
int a = 3;
int b = 5;
if (a > b) {
  printf("max is %d\n", a);
}
else if (a < b) {
  printf("max is %d\n", b);
}
else {
  printf("a and b are equal\n").
}
```

Задача: вывести максимальное число из чисел a и b, вывести фразу если они равны.

Вложенность

- Одни ветвления можно вклыдвать в другие.
- Если в блоке кода одна инструкция, то скобки { } можно опустить.

```
int a = 3, b = 5, c = 7, m = 0;
if (a >= b) {
  if (a >= c)
    m = a;
  else
    m = c;
}
else {
  if (b >= c)
    m = b;
  else
    m = c;
}
printf("%d\n", m);
```

Что делает программа?

Вложенность

- Одни ветвления можно вклыдвать в другие.
- Если в блоке кода одна инструкция, то скобки { } можно опустить.

```
int a = 3, b = 5, c = 7, m = 0;
if (a >= b) {
  if (a >= c)
    m = a;
  else
    m = c;
}
else {
  if (b >= c)
    m = b;
  else
    m = c;
}
printf("%d\n", m);
```

Что делает программа?

1. Выводит на экран максимум из 3 чисел.

Цикл с предусловием while

Цикл будет выполняться снова и снова, пока логическое выражение равно истине.

```
//цикл с предусловием
while (логическое выражение) {
  код
}
```

```
int i = 0;
while (i <= 10) {
  printf("%d", i);
  i = i + 1;
}
```

Задача: вывести числа от 0 до 10.

Цикл с предусловием while

```
int i = 0;
int sum = 0;
while (i <= 10) {
  sum = sum + i;
  i = i + 1;
}
printf("%d", sum);
```

Задача: вывести сумму нечетных чисел 0 до 10.

Цикл с предусловием while

```
int res = 1, i = 1;
int x = 3;
int p = 8;
while (i <= p) {
  res = res * x;
  i = i + 1;
}
printf("%d", res);
```

Что делает программа?

Цикл с предусловием while

```
int res = 1, i = 1;
int x = 3;
int p = 8;
while (i <= p) {
  res = res * x;
  i = i + 1;
}
printf("%d", res);
```

Что делает программа?

1. Возводит число x в степень p

Цикл с предусловием while

Задача: вывести разряды числа на отдельных строках.
Подзадачи:

- Как выделить один младший разряд? $digit = n\%10$
- Как “отрезать” один младший разряд? $n = n/10$
- Как долго повторять шаги? пока число не станет равно 0.

Цикл с предусловием while

Задача: вывести разряды числа на отдельных строках.

```
int n = 153;
int digit = 0;
while (n != 0) {
  digit = n % 10;
  printf("%d\n", digit);
  n = n \ 10;
}
```

Есть ли n, на котором отработает некорректно?

Цикл с предусловием while

Задача: вывести разряды числа на отдельных строках.

```
int n = 153;
int digit = 0;
while (n != 0) {
  digit = n % 10;
  printf("%d\n", digit);
  n = n \ 10;
}
```

Есть ли n, на котором отработает некорректно?

1. Если n == 0, то ничего выведено не будет

Цикл с постусловием do while

Цикл выполнится хотя бы 1 раз, т.к. условие (логическое выражение) проверяется в конце.

```
do {
  код
} while(условие);
```

```
int n = 153;
int digit = 0;
do {
  digit = n % 10;
  printf("%d\n", digit);
  n = n \ 10;
} while (n != 0)
```

while + if

Управляюшие конструкции можно комбинировать.
Задача: определить, является ли число n простым.
Алгоритм:

- делить число n на все числа от 1 до n.
- подсчитать, сколько раз разделиться нацело (остаток равен 0).
- если разделилось нацело, только два раза — число простое.

while + if

```
int n = 153;
int count = 0;
int i = 1;
while (i <= n) {
  if (n % i == 0)
    count = count +1;
  i = i + 1;
}

if (count == 2) {
  printf("%d is prime \n", n);
}
```

Есть ли n, на котором отработает некорректно?

while + if

```
int n = 153;
int count = 0;
int i = 1;
while (i <= n) {
  if (n % i == 0)
    count = count +1;
  i = i + 1;
}

if (count == 2) {
  printf("%d is prime \n", n);
}
```

Есть ли n, на котором отработает некорректно?

1. Если n == 1, то ничего выведено не будет

while + if

```
int n = 153;
int count = 0;
int i = 1;
if (n == 1) {
  count = 2;
}
else {
  while (i <= n) {
    if (n % i == 0)
      count = count +1;
    i = i + 1;
  }
}
if (count == 2) {
  printf("%d is prime \n", n);
}
```

Q: как эту программу можно ускорить?

Цикл со счетчиком

```
int i;
for (i = 0; i < 10; i = i + 1) {
  // код
}
```

Эти два цикла аналогичны.

```
int i;
i = 0;
while (i < 10) {
  // код
  i = i +1;
}
```

Цикл + Цикл

Задача: вывести таблицу умножения.

```
int i, j;
for (i = 1; i < 10; i = i + 1) {
  for (j = 1; j < 10; j = j + 1) {
    printf("%d ", i * j);
  }
  printf("\n");
}
```

Для каждого шага внешнего цикла по i выполняется 9 шагов внутреннего цикла по j.