Управляющие операторы

Общие сведения

Как уже говорилось ранее, алгоритмы и их участки можно условно делить на линейные, нелинейные и циклические.

До сих пор мы писали линейные программы – алгоритм представляет собой последовательность действий, которая при любых условиях будет выполнена.

Операторы, которые в зависимости от заданных условий, выбирают, какую ветвь алгоритма нужно выполнить, называются управляющими

Оператор условия if-else

Самый простой и одновременно эффективный метод ветвление – условный переход. Если условие, стоящее после **if** в скобках истинно, то выполняется идущая сразу после него ветвь, иначе выполняется ветвь, идущая сразу за **else**. Ветвь **else** может отсутствовать.

if	(условие)	{ветвь истины}	else	{ветвь лжи}

Для всех операторов, кроме **switch**, наличие фигурных скобок не является обязательным в том случае, если должна выполняться только одна инструкция.

```
1  if (a < b) {
2      cout << "A < B" << endl;
3  }
4  else {
5      cout << "A >= B" << endl;
6  }
```

Оператор переключения switch

Существуют ситуации, когда есть необходимость в выполнении определенных действий в зависимости от **конкретных значений** переменных. Тогда принято использовать оператор **switch** и вспомогательные **case**, **break**, **default**.

Считается, что внутри **switch** весь код - линейный участок кода с возможностью начала его выполнения с определенной части. Для этого используется оператор **case**. Для выхода из **switch** используется оператор прерывания **break**.

Для обработки незаданных значений используется вхождение **default**.

```
1  switch(a) {
2      case 0:
3      case 1:
4          cout << "a < 2" << endl;
5          break;
6
7      case 2:
8          cout << "a = 2" << endl;
9      case 3:
10         //Сюда кроме 3 попадает и 2
11         cout << "a div 2 = 1" << endl;
12         break;
13
14     case 4:
15         cout << "a = 4" << endl;
16         break;
17
18     default:
19         cout << "a not in range [0; 4]" << endl;
20 }
```

Схема использования switch

switch	(переменная)	{	Блок кода с вхождениями	}

Как уже было продемонстрировано ранее, блок кода состоит из вхождений **case/default** и прерываний **break**.

case	значение	:
default	:	
break	;	

Цикл условия while

Цикл while выполняет заданную ветвь кода до тех пор, пока выполняется условие. Есть так же цикл с постусловием do-while. Последний гарантирует хотя бы один проход цикла.

Проходы цикла называются итерациями.

while	(условие)	{блок кода}

do	{блок кода}	while	(условие)	;

```
1  while (x > 0)
2      x /= 10;
3
4  do {
5      x /= 10;
6  } while (x > 0);
```

Цикл счетчика for

Цикл работает по следующем принципу

1. Инициализируется переменная-счетчик
2. Проверяется условие (если ложь - завершается)
3. Выполняется блок кода
4. Выполняется шаг цикла
5. Переход к пункту 2

for	(	Инициализация	;	Условие	;	Шаг цикла	)	{блок кода}

```
1 for(int i = 0; i < 10; ++i) {
2     cout << i << endl;
3 }
```