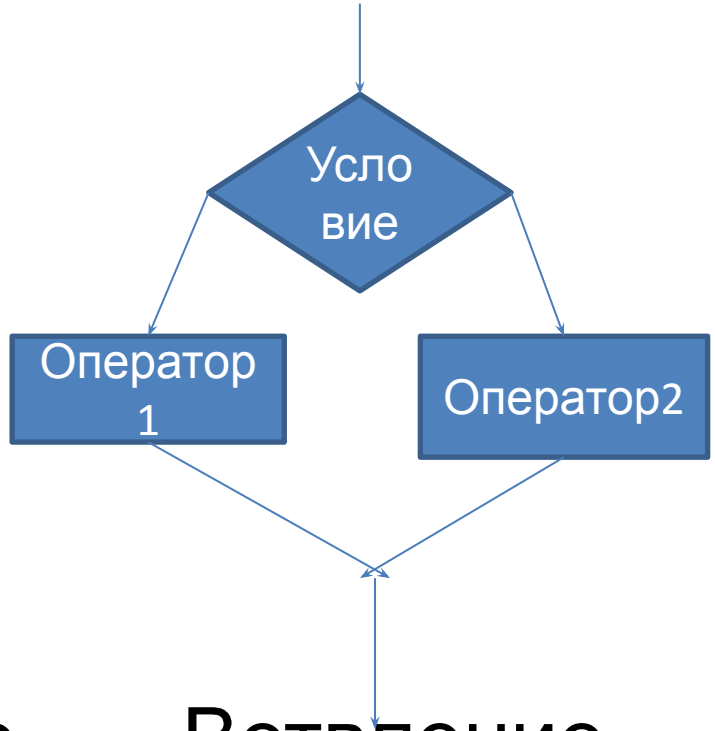
Структурное программирование

Структурное программирование

В теоретическом программировании доказано, что программу для решения задачи любой сложности можно составить только из трех структур, называемых следованием, ветвлением и циклом.

Структурное программирование

Следование



Ветвление

Цикл

Структурное программирование

Следование, ветвление и цикл называют базовыми конструкциями структурного программирования.

Следованием называется конструкция, представляет собой последовательное выполнение двух или более операторов (простых или составных).

Структурное программирование

Ветвление задает выполнение одного или другого оператора в зависимости от выполнения какого-либо условия.

Цикл задает многократное выполнение оператора (простого или составного).

Программа, состоящая из операторов структурного программирования легко читаема, ее проще отлаживать.

Структурное программирование

Оператор выражение

Любое выражение, заканчивающееся точкой с запятой, рассматривается как оператор, вычисляющий выражение. Частным случаем считается пустой оператор. Он используется по синтаксису, но не по смыслу.

Структурное программирование

Оператор ветвление

Условный оператор if используется в тех случаях, когда необходимо разветвить вычислительный процесс на два направления. Формат оператора:

If(expression) operator_1; [else operator_2];

В первую очередь вычисляется выражение expression, которое может иметь арифметический тип или тип указателя.

Структурное программирование

Если результат выражения отличен от нуля (true), выполняется первый оператор, иначе – второй. После чего управление передается оператору непосредственно следующему за условным. Ветвь, содержащая else может отсутствовать.

Если в какой-либо ветви необходимо выполнить более одного оператора, операторы заключаются в блок – { }.

Структурное программирование

Следует отметить, что переменные описанные внутри блока, вне блока не существуют.

Несколько примеров:

If(x<100) y = 20;

If(x>10&&y<20) z = x*y;

If(‘a’) ch = ‘c’;

If(-0.3287) max = b; else b += a;

If(true) rez = false;

If(0) break;

If(64) continue;

Структурное программирование

Оператор переключатель

Оператор переключатель – switch предназначен для разветвления вычислительного процесса на несколько направлений.

Структурное программирование

Общий формат этого оператора:

```
switch(expression)
{
case const_expr_1: [opertor_list_1];
case const_expr_2: [opertor_list_2];
case const_expr_3: [opertor_list_3];
……..
case const_expr_N: [opertor_list_N];
[default : operators ]
}
```

Структурное программирование

Выполнение оператора начинается с вычисления выражения (оно должно относиться к целочисленным типам), а затем управление передается первому оператору из списка, помеченного константным выражением, значение которого совпало с вычисленным. После этого, если выход из переключателя явно не указан, последовательно выполняются все остальные ветви.

Структурное программирование

- Выход из оператора switch обычно выполняется с помощью операторов break или return.
- Все константы должны иметь различные значения, но одного целочисленного типа.
- Если ни одного совпадения не произошло, выполняются операторы после default.

Структурное программирование

Пример простого калькулятора:

```
int op_1, op_2, rez;
char oper;
cout << “ Введите первый (левый) операнд “;
cin >> op_1;
cout << “ Введите знак операции ”;
cin >> oper;
cout << “ Введите второй (правый) операнд “;
```

Структурное программирование

```
bool f = true;
Switch(oper)
{
    case ‘+’: rez = op_1 + op_2; break;
    case ‘-’: rez = op_1 - op_2; break;
    case ‘*’: rez = op_1 * op_2; break;
    case ‘/’: rez = op_1 / op_2; break;
    default: cout << “ Неизвестная операция “;
    f = false;
}
```

Структурное программирование

If(f) cout << “ Результат = “ << rez << endl;

Проверьте работу этой программы и оцените результаты.

Структурное программирование

Операторы цикла

Операторы цикла используются для организации многократно повторяющихся вычислений. Цикл состоит из тела, то есть из последовательности операторов, которые необходимо выполнить, начальных установок, модификации параметров цикла и проверки условий продолжения выполнения оператора цикла.

Структурное программирование

Один проход цикла называется итерацией. Проверка условия выхода выполняется на каждой итерации. Если проверка происходит до тела цикла, то говорят о операторе цикла с предусловием. Если же проверка происходит после выполнения тела цикла – то оператор цикла с постусловием. Отличие состоит в том, что оператор цикла с постусловием выполняется хотя бы один раз.

Структурное программирование

Переменные, используемые в теле цикла и при проверке условия окончания, называются параметрами цикла. Параметры цикла должны быть установлены до начала работы оператора цикла.

Цикл завершается, если условие его продолжения не выполняются.

Структурное программирование

Возможно принудительное завершение итерации с помощью операторов break, continue, return, goto.

Цикл с предусловием

Общий формат оператора с предусловием следующий:

```
while(expression) operator;
```

Структурное программирование

Выражение (арифметический тип, тип указателя) определяет условие повторения выполнения тела цикла. Если результат выражения отличен от ноля (false), цикл повторяется. Вычисление выражения осуществляется перед каждым циклом. Может случиться так, что оператор цикла не выполнится ни разу.

Структурное программирование

Распространенный прием программирования – организация бесконечного цикла:

while(true), while(1), while(-5.5), etc.

Принудительный выход из тела осуществляется при выполнении какого-либо условия.

Структурное программирование

Цикл с постусловием

Цикл с постусловием выполняется по следующему формату:

do operator while(expression);

В начале выполняется простой или составной оператор, после чего вычисляется выражение. Если значения выражения отлично от ноля, выполнение цикла продолжается. Выход из цикла допустим при выполнении некоторого условия через оператор break.

Структурное программирование

Следует отметить, что среди операторов тела цикла с предусловием и с постусловием должен присутствовать по крайней мере один оператор, влияющий на окончание цикла, то есть меняющий параметры цикла.

Структурное программирование

Оператор цикла с параметром

Цикл с параметром имеет следующий формат:

for(инициализатор; выражение;модификатор)

operator;

Инициализатор – область объявления параметров цикла. Если их несколько, то они перечисляются через запятую. Область действия переменных – тело цикла. Инициализация осуществляется один раз.

Структурное программирование

Выражение (имеет арифметический тип или тип указателя) определяет условие выполнения цикла. Если результат выражения отличен от ноля, цикл выполняется.

Модификатор изменяет значения параметров цикла и выполняется сразу после очередной итерации. В этой части допускается несколько операторов через запятую.

Структурное программирование

В теле цикла запрещается изменять параметры цикла принудительным образом!

Любая из частей оператора цикла с параметром может отсутствовать, но точки с запятой обязаны быть на своих местах:

```
for( ;  ;) operator;
```

Еще один пример бесконечного оператора цикла.

Структурное программирование

Примеры:

for(int I = 0; i<=10; i++) ……..

for(int I = 0, j = 0; i<10 && j <10; i++,j++) …….

float X, Xn = 0.1F, Xk = 1.2F, Dx = 0.01;

for(X = Xn; X < Xk; X += Dx) ……….

Операторы цикла взаимозаменяемы, какой из них использовать – дело вкуса программиста и условиями, диктуемыми исходной задачей.

Структурное программирование

***Операторы передачи управления**

В языке C++ есть четыре оператора, изменяющих естественный порядок выполнения программы:

- оператор безусловного перехода goto;
- оператор выхода из цикла break;
- оператор перехода к следующей итерации continue;
- оператор возврата значения из функции return.

Структурное программирование

Оператор goto

Оператор безусловного перехода имеет формат: goto метка;.

В теле того же блока (например, функции) должна присутствовать единственная конструкция вида:

метка: оператор;

Метка – обыкновенный идентификатор, действующий в пределах блока его определения.

Использование оператора безусловного перехода – признак нехорошего стиля программирования.

Структурное программирование

Не следует передавать управление в тело операторов if, switch, операторов цикла. Это может привести к недетерминированному поведению процесса.

Оператор break

Этот оператор используется внутри операторов цикла, операторов if, switch для обеспечения перехода в точку программы, находящуюся непосредственно за оператором, внутри которого находится break.

Структурное программирование

Например,

```
for(int n = 0; fabs(ch)>eps; n++)
{
   y += ch;
   if(n>MaxIter) break;
   // ……….
}
```

Структурное программирование

Оператор continue

Оператор перехода к следующей итерации цикла пропускает все оставшиеся до конца операторы и передает управление на первый оператор в теле цикла.

Оператор return

Оператор возврата из функции завершает выполнение функции и передает управление в точку вызова функции.

Структурное программирование

Формат оператора return следующий:

return[expression];

Выражение expression должно иметь скалярный тип. Если тип возвращаемого функцией результата описан как void, выражение должно отсутствовать.

Указатели и массивы

Указатели

Когда компилятор обрабатывает оператор объявления переменной, например, int var_int = 10;, он выделяет память в соответствии с типом(int) и инициализирует его значением 10. Все обращения в программе к этой переменной по ее имени заменяется компилятором на адрес памяти, в которой хранится значение переменной.

Указатели и массивы

Программист в своей программе может сам определить свои собственные переменные для хранения адресов памяти. Такие переменные называются *указателями*.

В C++ различают три вида указателей:

- указатель на объект;
- указатель на функцию;
- указатель на тип void.

Указатели и массивы

Эти указатели отличаются свойствами и набором допустимых действий (операций). Указатель не является самостоятельным типом, он всегда связан с другим конкретным типом.

Указатель на объект имеет следующий формат:

тип *имя_указателя [= инициализатор];

Указатели и массивы

Тип может быть любым (стандартный, пользовательский), кроме ссылки, причем, к этому моменту тип может быть только объявлен, но еще не определен.

Символ ‘*’ (звездочка) относится непосредственно к имени, поэтому при групповом объявлении указателей, символ нужно ставить перед каждым объектом, например,

```
double *ptr_d_1, var_double, *ptr_d_2;
```

Указатели и массивы

Память выделяемая под указатель зависит от разрядности машины.

Указатель на функцию по синтаксису отличается от остальных указателей, общий формат которого следующий:

тип (*имя_указ)(список_параметров);

Например,

```
double (*ptr_func)(double, const int&);
```

Указатели и массивы

Указатель на тип void – отдельный указатель, используемый в тех случаях, когда конкретный тип объекта адрес которого необходимо сохранить, не известен или не определен. Возможны варианты хранения адресов переменных различного типа, но перед обращением к объекту через указатель на тип void, его необходимо преобразовать к указателю на конкретный тип явным образом.

Указатели и массивы

Указатель может быть константой или переменной, а также указывать на константу или переменную.

Указатели и массивы

Указатели и массивы

Указатели и массивы

Указатели и массивы