Семинар 2. Алгоритмы ветвления, условный оператор, тернарный оператор, оператор switch (переключатель), оператор безусловного перехода

1. Алгоритмы ветвления

Это алгоритм, в котором существует несколько различных путей исполнения кода

2. Условный оператор

Существует две формы условного оператора:

1. Сокращённая форма (if без else)
2. Полная форма (if … else …)

2. Условный оператор

Сокращённая форма условного оператора:
if (условие) оператор;

Если условие является истинным, то оператор будет выполнен.

2. Условный оператор

Существует возможность исполнить несколько операторов, для этого нужно вставить эти операторы внутрь т.н. операторных скобок:
if (условие) {оператор1; оператор2; …}

2. Условный оператор

Операторные скобки, { }, для внешнего кода формируют единый блок-оператор, называемый в программировании составным оператором. Внутри операторных скобок может быть сколь угодно много других операторов.

2. Условный оператор

Пример. Получение частного (результат деления).

```
double a, b;
cout << "Enter 2 numbers\n";
cin >> a >> b;
if (b == 0)
{
        cout << "The denominator equal to zero\n";
        return 0;
}
cout << "Quotient = " << a / b << "\n";
return 0;
```

В программе в 2 различных местах может произойти возврат.

2. Условный оператор

Пример. Получение частного (результат деления).

```
double a, b;
cout << "Enter 2 numbers\n";
cin >> a >> b;
if (b == 0)
{ // эти скобки можно опустить по причине
  // вхождения в составной оператор только одного оператора
        cout << "The denominator equal to zero\n";
}
else
{ // и эти скобки также можно опустить
        cout << "Quotient = " << a / b << "\n";
}
return 0;
```

2. Условный оператор

Полная форма условного оператора:

```
if (условие)
    оператор1;
else
    оператор2;
```

Если условие истинно, выполнится оператор1, если ложно – оператор2.

2. Условный оператор

Пример. Определение чётности числа.

```
int n;
cout << "Enter integer\n";
cin >> n;
if (n % 2 == 0)
        cout << "Even\n";
else
        cout << "Odd\n";
return 0;
```

2. Вложенные условные операторы

В целях решения задач, зависящих от нескольких условий, условные операторы могут быть вложенными друг в друга

2. Вложенные условные операторы

Пример. Получение максимального из трёх чисел.

```
int a, b, c;
cout << "Enter 3 integers\n";
cin >> a >> b >> c;
cout << "Max = ";
if (a > b)
        if (c > a)
                cout << c;
        else
                cout << a;
else
        if (c > b)
                cout << c;
        else
                cout << b;
return 0;
```

2. Условные операторы и логические операции


```
int a, b, c;
cout << "Enter 3 integers\n";
cin >> a >> b >> c;
cout << "Max = ";
if (a >= b && a >= c) {cout << a; return 0;}
if (b >= a && b >= c) {cout << b; return 0;}
if (c >= a && c >= b) {cout << c; return 0;}
return 0;
```

2. Условные операторы и логические операции

a	b	a && b
0	0	0
0	1	0
1	0	0
1	1	1

Конъюнкция. Логическое и. Логическое умножение.

Результат конъюнкции истинен тогда, когда все операнды истинны.

2. Условные операторы и логические операции

a	b	a || b
0	0	0
0	1	1
1	0	1
1	1	1

Дизъюнкция. Логическое или. Логическое сложение.

Результат дизъюнкции истинен тогда, когда хотя бы один из операндов истинен.

2. Условные операторы и логические операции

a	!a
0	1
1	0

Отрицание. Инверсия. Логическое не.

3. Тернарный оператор

Вид тернарного оператора:

усл_выражение ? выр_если_ист : выр_если_ложь

Тернарные операторы, как и операторы if, могут быть вложенными.

3. Тернарный оператор

Пример. Определение чётности числа.

```
int n;
cout << "Enter integer\n";
cin >> n;
(n % 2 == 0) ? (cout << "Even\n") : (cout << "Odd\n");
return 0;
```

4. Оператор switch (оператор выбора)


```
switch(переключающее_выражение)
{
    case выражение1: операторы1;
    case выражение2: операторы2;
    case выражение_n: операторы_n;
    default: операторы
}
```

4. Оператор switch (оператор выбора)

Switch – наиболее удобное средство для организации множественного ветвления.

4. Оператор switch (оператор выбора)

Управление передаётся к тому из операторов, для которого значение константного выражения после case совпадает со значением переключающего выражения. Выполняются все операторы, следующие после оператора, которому передано управление.

4. Оператор switch

Пример. Организация меню.

```
int a, b, item;
cout << "Enter 2 integers\n";
cin >> a >> b;
cout << "1 - addition, 2 - subtraction, 3 - multiplication, 4 - division\n";
cin >> item;
cout << "Result = ";
switch (item)
{
case 1: cout << a+b; break;
case 2: cout << a-b; break;
case 3: cout << a*b; break;
case 4: cout << a/b; break;
default: cout << " invalid operation ";
}
cout << "\n";
return 0;
```

4. Оператор switch

Пример. Организация меню.

```
int a, b, item;
cout << "Enter 2 integers\n";
cin >> a >> b;
cout << "1 - addition, 2 - subtraction, 3 - multiplication, 4 - division\n";
cin >> item;
cout << "Result = ";
if (item==1) cout << a+b;
else if (item==2) cout << a-b;
else if (item==3) cout << a*b;
else if (item==4) cout << a/b;
else cout << " invalid operation ";
cout << "\n";
return 0;
```

5. Оператор безусловного перехода

В Си++ существует оператор безусловного перехода – оператор goto.

Вид оператора:
goto идентификатор;

5. Оператор безусловного перехода

«За многие годы я утвердился во мнении о том, что квалификация программистов - функция, обратно зависящая от частоты появления операторов go to в их программах.»

Эдсгер Вайб Дейкстра, Технологический университет, Нидерланды

http://www.vspu.ac.ru/~chul/dijkstra/goto/goto.htm

5. Оператор безусловного перехода

Пример. Организация бесконечного цикла.

```
int n;
M1: cout << "Enter integer\n";
cin >> n;
(n % 2 == 0) ? (cout << "Even\n") : (cout << "Odd\n");
goto M1;
return 0;
```

5. Оператор безусловного перехода

Оператор break – частный случай оператора goto, служит для принудительного выхода из цикла или из переключателя switch.

5. Оператор continue

Оператор continue употребляется только в операторах цикла. С его помощью завершается текущая итерация и начинается проверка условия дальнейшего продолжения цикла.

Задачи для решения в классе

1. Составить программу для решения квадратного уравнения вида $ax^2+bx+c=0$
2. Составить программу, принимающую в качестве входных значений длины трёх сторон треугольника. Программа должна выводить следующие данные о треугольнике (в порядке приоритета):
   - Прямоугольный
   - Равносторонний (правильный)
   - Равнобедренный (любые две стороны равны)
   - Вырожденный (сумма каких-либо двух сторон равна третьей)
   - Не является треугольником (введены нули или отрицательные числа)

   Если программа вывела какое-либо из слов, стоящих выше в списке, то другие слова не выводятся.

Задачи

1. Пользователь вводит порядковый номер пальца руки (1 – большой, 2 – указательный, …). Вывести на экран название этого пальца.
2. Даны два числа. Вывести большее из них.
3. Даны четыре числа. Вывести большее из них.
4. Дано целое число. Если оно чётное, прибавить к нему 2, а если нечётное – вычесть 1. Вывести результат.
5. Пользователь вводит число от 0 до 9999. Вывести количество цифр в числе.
6. По введённым номеру дня в месяце, номеру месяца и високосности года вывести количество дней, прошедшее с начала года.