Flow Control

Control Flow Basics

- If statement allows for alternative paths of execution

```
int main ()
{
 char option;

 cout << "Enter h to say hello, or m to multiply a number by 2 :>";
 cin >> option;

 if ( option == 'h' )
   cout << "Hello world" << endl;
 else if ( option == 'm' )  {
   int n;
   cout << "What number shall I multiply by 2? ";
   cin >> n;
   cout << n << " * 2 = " << n * 2 << endl;
 }  else
   cout << "Please enter either h or m here!" << endl;
}
```

Conditional Expressions

- Used in if statements

```
if ( x == 6 )
  ...
```

Use == not = for 'is equal to'.

```
if ( y != 4 )
  ...
```

!= means 'is not equal to'.

```
if ( x >= 4 && y < 2 )
  ...
```

&& means 'and'.

```
if ( x != 9 || y <= 7 )
  ...
```

|| means 'or'.

Assignment vs Comparison Gotcha

- Beware using assignment instead of comparison
  - = instead of ==

```
if ( option = 'h' )
   cout << "Hello C++\n";
```

The expression in the () actually puts the character 'h' into the variable option, overwriting its original value. Whether the statement after the if gets executed depends on the value within the (parentheses), which in this case will always by 'h'. C++ treats any non-zero value as being 'true'. As 'h' is not zero, the statement after the 'if' is always executed.

- Modern compilers warn about this

Conditional Expression

- Shorthand alternative to if statement when value is required

```
int main ()
{
 int n, m;

 cout << "Enter two numbers: ";
 cin >> n >> m;

 cout << "Larger number: "
      << ( ( n>m ) ? n : m )
      << endl;
}
```

```
$ c++ -o cexpr cexpr.cpp
$ ./cexpr
Enter two numbers: 4 7
Larger number: 7
$
```

The switch Statement

- Used when different actions required based on value of an expression

```
…
char cmd;
cin>>cmd;

switch ( cmd ) {
  case 'a':
  case 'A':
    do_option_A();
    break;
  case 'b':
  case 'B':
    do_option_B();
    break;
  default:
    cout << "Illegal option: " << cmd << endl;
}
…
```

The While Loop

- Repeat execution until condition evaluates to false

```
int main ()
{
 char cmd = 'y';

 while ( cmd == 'y' ) {
   do_stuff();
   cout << "Do more stuff? ";
   cin >> cmd;
 }

 cout << "Bye..." << endl;
}
```

```
$ ./while
Doing some stuff...
Do more stuff? y
Doing some stuff...
Do more stuff? n
Bye...
$
```

Exit Condition Loop – do-while

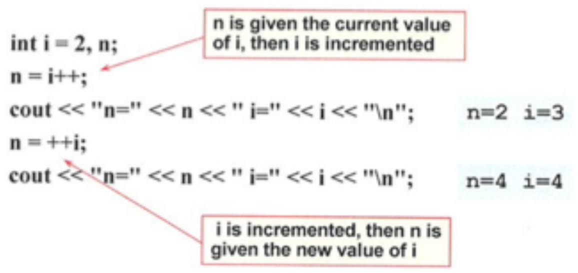
- Body of the loop guaranteed to be executed at least once

```
int main ()
{
 char cmd;

 do {
   do_stuff();
   cout << "Do more stuff? ";
   cin >> cmd;
 } while ( cmd == 'y' );

 cout <<"Bye..." << endl;
}
```

```
$ ./dowhile
Doing some stuff...
Do more stuff? y
Doing some stuff...
Do more stuff? n
Bye...
$
```

Increment and Decrement Operators

- May be prefix or postfix
  - i++ or ++i
  - i-- or --i
- Important to appreciate the difference
  - especially for C++



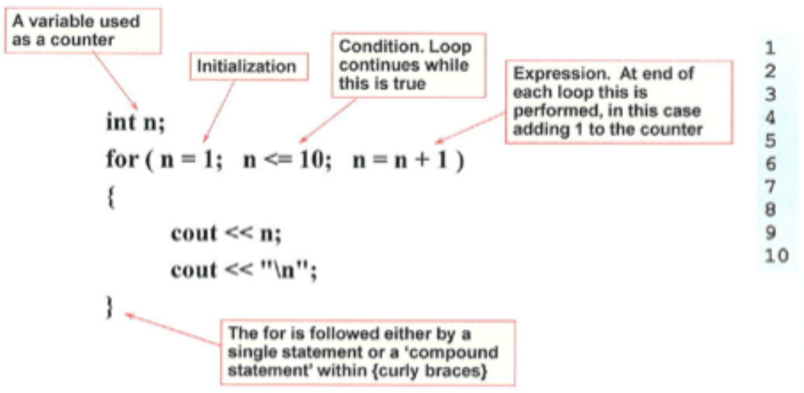
```
int i = 2, n;
n = i++;
cout << "n=" << n << " i=" << i << "\n";        n=2  i=3
n = ++i;
cout << "n=" << n << " i=" << i << "\n";        n=4  i=4
```

n is given the current value of i, then i is incremented

i is incremented, then n is given the new value of i

The for loop

- Used when number of iterations is predictable
  - Fixed number of iterations



```
int n;
for ( n = 1;  n <= 10;  n = n + 1 )
{
        cout << n;
        cout << "\n";
}
```

```
1
2
3
4
5
6
7
8
9
10
```

A variable used as a counter

Initialization

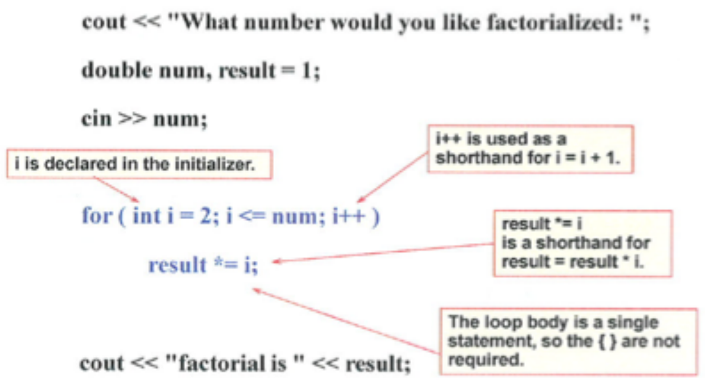
Condition. Loop continues while this is true

Expression. At end of each loop this is performed, in this case adding 1 to the counter

The for is followed either by a single statement or a 'compound statement' within {curly braces}

The for loop

- Used when number of iterations is predictable
  - Fixed number of iterations



```
cout << "What number would you like factorialized: ";
double num, result = 1;
cin >> num;
for ( int i = 2; i <= num; i++ )
    result *= i;

cout << "factorial is " << result;
```

i is declared in the initializer.
i++ is used as a shorthand for i = i + 1.
result *= i is a shorthand for result = result * i.
The loop body is a single statement, so the { } are not required.