

Operators Precedence

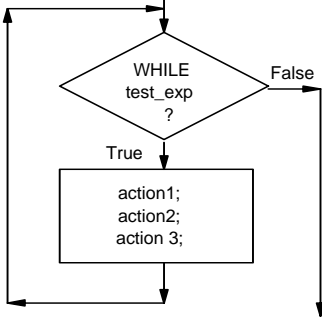
```

( )
float() int() sizeof() !
* / %
+ -
<< >>
< <= > >=
== !=
&&
||
=
    
```

Copyright © 2003 R.M. Laurie | 1

Repetition (Loop) Structure

- Control structure used to repeat a sequence of instruction is in a loop.
- The simplest loop structure is the while()

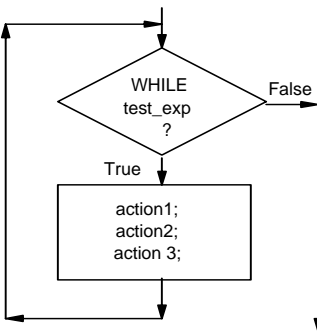


```

while(test_exp)
{
    action1;
    action2;
    action3;
}
    
```

Copyright © 2003 R.M. Laurie | 2

Repetition Structure Example



```

nCount = 0;
while(nCount < 10)
{
    cout << nCount "\n";
    nCount=nCount+1;
}
cout << "Done"
    
```

Copyright © 2003 R.M. Laurie | 3

Temperature Program -Ver.3

```

#include <iostream.h>
#include <iomanip.h>
int main(void)
{
    // DECLARATION SECTION
    char cQuestion;
    float fTemperature;
    cout << fixed; // Allows float point format

    // PROCESSING SECTION
    cout << "This program converts temperatures between\n"
         << "degrees Celsius and degrees Fahrenheit.\n"
         << "You may enter either a Celsius or "
         << "Fahrenheit\n temperature for conversion.\n\n";
}
    
```

Copyright © 2003 R.M. Laurie | 4

Temperature Program -Ver.3

```

while(1)
{
    cout << "> <-- Enter C (Celsius), F (Fahrenheit), or Q (Quit)\r>";
    cin.get(cQuestion);
    cin.ignore(100,'\n');

    if(cQuestion == 'C' || cQuestion == 'c')
    {
        cout << " <-- Enter temperature in degrees Celsius\r>";
        cin >> fTemperature;
        cin.ignore(100,'\n');
        cout << "Results: " << setprecision(2)
            << fTemperature << " C = "
            << (((fTemperature * 180) / 100) + 32) << " F\n";
    }
}
    
```

Copyright © 2003 R.M. Laurie 5

Temperature Program -Ver.3

```

else if(cQuestion == 'F' || cQuestion == 'f')
{
    cout << " <-- Enter temperature in degrees Fahrenheit \r>";
    cin >> fTemperature;
    cin.ignore(100,'\n');
    cout << "Results: " << setprecision(2)
        << fTemperature << " F = "
        << (((fTemperature - 32) * 100) / 180) << " C\n";
}
else if (cQuestion == 'Q' || cQuestion == 'q')
{
    cout << "\nGood bye";
    return 0;
}
else
    cout << "\a";
}
    
```

Copyright © 2003 R.M. Laurie 6

Temperature Program -Ver.3

This program converts temperatures between degrees Celsius and degrees Fahrenheit. You may enter either a Celsius or Fahrenheit temperature for conversion.

```

>g<-- Enter C (Celsius), F (Fahrenheit), or Q (Quit)
>g<-- Enter C (Celsius), F (Fahrenheit), or Q (Quit)
>c<-- Enter C (Celsius), F (Fahrenheit), or Q (Quit)
>25 <-- Enter temperature in degrees Celsius
Results: 25.00 C = 77.00 F
>C<-- Enter C (Celsius), F (Fahrenheit), or Q (Quit)
>0 <-- Enter temperature in degrees Celsius
Results: 0.00 C = 32.00 F
>F<-- Enter C (Celsius), F (Fahrenheit), or Q (Quit)
>95 <-- Enter temperature in degrees Fahrenheit
Results: 95.00 F = 35.00 C
>q<-- Enter C (Celsius), F (Fahrenheit), or Q (Quit)

Good bye
    
```

Copyright © 2003 R.M. Laurie 7

break; continue; commands

```

while(test_exp)
{
    if(expression1)
        continue;
    if(expression2)
        break;
    action1;
    action2;
}
    
```

Copyright © 2003 R.M. Laurie 8

Another Example

```
# include <iostream.h>
int main(void)
{
    long InSSN;
    long InMax = 999999999, InMin = 100000000;
    while (1)
    {
        cout << "Enter Soc. Sec. Number\n>";
        cin >> InSSN;

        if (InSSN >= InMin && InSSN <= InMax)
            break;
        cout << "ERROR: Soc. Sec Number must be
            9 digits\n\n";
    }
    cout << "Social Security Number is Valid";
    return(0);
}
```

Copyright © 2003 R.M. Laurie 9

Functional Decomposition -1

- Also called Top-Down Design, Modular Design, and Structured Design
- Develop a program by dividing problem into sub-modules that are more easily coded and debugged.
- Then after sub-modules are coded, debugged, and tested, Assemble modules to create program.
- Text p181 to 191 Do walk through.

Copyright © 2003 R.M. Laurie 10

Functional Decomposition -2

- Best to do Top Flow Chart first.
- Create flow charts for modules.
- Code modules.
- Debug Test Debug Test.
- Assemble modules into program.
- Debug Test Debug Test.

- Pseudocode may help,
 - See Example p.185
 - See Chapter 6 in Robertson for Loops

Copyright © 2003 R.M. Laurie 11

Functional Decomposition - 3

- Don't forget to test all possible selection structure paths of final program.
- Use breakpoints and watch variables in the testing phase as well as debugging phase.
- See Test and Debug Hints p.249-263

Copyright © 2003 R.M. Laurie 12