

02 - C++ Design and Thinking

Dr. Robert Lowe

Division of Mathematics and Computer Science
Maryville College

Outline

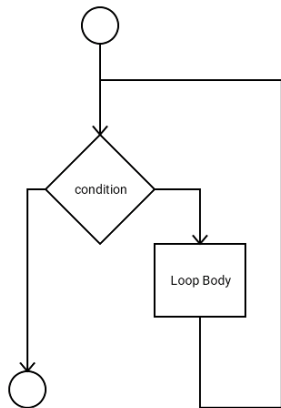
- 1 Loops
- 2 Functions
- 3 Makefile
- 4 Lab Assignment

While Loop

While Loop Syntax

```
while ( condition )  
    statement/block
```

- If the *condition* is true, the loop body is executed.
- After the loop body executes, the process begins again.
- How many times will the loop body execute?
 - Zero or more times!

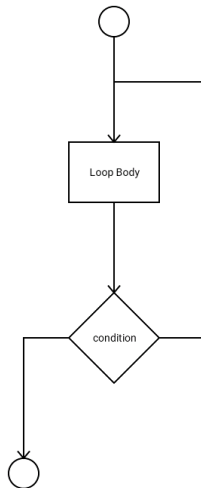


Do..While Loop

While Loop Syntax

```
do  
    statement/block  
while ( condition );
```

- The `do..while` loop is called the postcondition loop.
- The condition is checked after the loop body.
- Executes 1 or more times.
- Commonly used with input validation and menus.



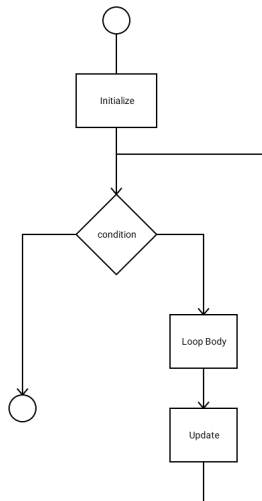
For Loop

The For Loop

```
for ( initialize; condition; update ) {  
    loop body  
}
```

Example: Count to 10

```
for(num=0; num <= 10; num++)  
{  
    cout << num << endl;  
}
```



The Problem

- Thus far, all known software is written by humans.
- The human race is a member of the hominidae family.
- We are apes.
- We are the most successful ape.
- We are still apes, nonetheless.
- We can hold about seven ideas in our heads at once.
- This is insufficient for almost all useful programming tasks.



Source: wikipedia.org 

Function Definition

Function Syntax

```
return_type name( parameters )  
{  
    //function body  
}
```

- A function is a block of code that can be called multiple times.
- A function's signature consists of the following:
 - return type** This is the type of value the function evaluates to when it is used in an expression.
 - name** The identifier which names the function.
 - parameters** The local variables which receive the arguments of the function.

Function Prototypes

- Function prototypes allow you to declare a function before it is defined.
- This is a sort of “contract” between you and the compiler.
- This allows you to have functions in any order in the file.
- Change the first few lines of `roman.cpp` so it reads as follows:

```
#include <iostream>
```

```
using namespace std;
```

```
//function prototypes
```

```
void print_roman_numeral(int value);
```


Gluing it Together With Header Files

- A function must be declared before it can be used.
- Because the definitions are in a separate file, they are not declared in the file that contains our main function.
- We can solve this with prototypes.
- Repeating prototypes in every file is painful.
- Enter the header file!
- A header file usually has a `.h` extension and contains:
 - Function Prototypes
 - Constants
 - Type Definitions

Makefile – Explicit Recipes

```
sodasim: sodasim.o soda-machine.o
    g++ -o sodasim sodasim.o soda-machine.o

sodasim.o: sodasim.cpp soda-machine.h
soda-machine.o: soda-machine.cpp soda-machine.h
```

Some Predefined Variables

- The make syntax is itself a scripting language.
- Variables begin with dollar signs \$.
- There are several pre-defined variables, the two most commonly used ones are:
 - \$@ – The name of the target
 - \$^ – The list of all ingredients
- We could simplify the sodasim Makefile like so:

```
sodasim: sodasim.o soda-machine.o  
    g++ -o $@ $^
```

```
sodasim.o: sodasim.cpp soda-machine.h  
soda-machine.o: soda-machine.cpp soda-machine.h
```

Example Makefile – Address Book

```
TARGETS=stock

#application builds
all: $(TARGETS)
stock: iofun.o main.o stock.o transaction.o portfolio.o
      g++ -o $@ $^

#object files
iofun.o: iofun.h iofun.cpp
main.o: main.cpp iofun.h stock.h transaction.h portfolio.h
stock.o: stock.h stock.cpp
transction.o: transaction.cpp transaction.h
portfolio.o: portfolio.cpp portfolio.h

#delete all binaries
clean:
      rm -f *.o $(TARGETS)
```

Programming Project 5.9

Programming Project 5.9 from Big C++

Write a program that, given a month and year, prints a calendar, such as

```
        June 2016
Su Mo Tu We Th Fr Sa
      1  2  3  4
  5  6  7  8  9 10 11
12 13 14 15 16 17 18
19 20 21 22 23 24 25
26 27 28 29 30
```

Make a helper function to print the header and a helper function to print each row.