Control Flow: Decision & Repetition Statements
September 27th 2005
Fun Stuff

- Good/Bad News
- Quiz Rumors
- Killing Trees
- DoS
Outline

• Review
  - keywords
  - basic data types, arrays, string handling
  - command line arguments

• Control Flow (break, return, continue)

• Decision Statements (if, else, switch, case)
  - boolean expressions

• Repetition Statements (for, while)
  - array processing
Keywords in C (reserved words)

if, else, switch, while, for, do
break, continue, goto, case, default
int, float, double, char
long, short, signed, unsigned, register, const, volatile, extern, static, auto, entry
typedef, struct, union, enum, sizeof
return, void
Control Flow

- The actual sequence of instructions executed
- Not necessarily the order of the source listing
- Groups of related code go in statement blocks
- {}
Changing Control Flow

- Predicated on the evaluation of a boolean expression or explicit keyword
- Two ways to change control flow
  - decide on a choice between alternatives
  - repeat the current block of statements
Boolean Expressions (review)

• Boolean expressions are any valid C expression that evaluates to an integer value
• The value zero is taken to mean 'false'
  – any other value is 'true', although 1 (one) is used most often by convention
• Programs can make a decision between two different flows of control based on the result of a boolean expression
  – also based on the value of a computation
boolexp.c demo
• The 'if' keyword is an operator that evaluates a boolean expression and conditionally executes the code of statement block immediately following the 'if' if the expression evaluates to 'true'

```java
if(expression)
{
    //code to execute
}
```
else

• If 'if' statement evaluates to 'false', then the code statements in the body of the 'if' are not executed.
  – Instead, control flow 'falls through' the if
• Sometimes, we want to execute code if the condition is false. This is accomplished with 'else'

    if(condition)
    {
      //code1
    }else{
      //code2
    }
The 'switch' statement allows you to pick from different cases:

```java
int data_value = 0;
switch(data_value)
{
    case 0 :
        //do something
        break;
    case 1 :
        //do something else
        break;
    default :
        //do safe thing
}
```
Looping & Repetition

• Often, you want to execute the same set of statements multiple times
  – reading input
  – drawing graphics
  – calculating something

• need a way to 'loop' or repeat
  – loop control variable
  – initialization
  – increment/decrement/loop control maintenance
  – condition
The 'for' statement is like 'while' but gathers the bookkeeping work into a single statement

```c
for(i=0;i<some_limit;i++)
{
    //do loop work
}
```
The 'while' statement allows for looping while a condition is true.

while(1) {
    //loop forever
}

int counter=0;
int limit = 10;
while(counter<limit) {
    printf("counter == %d\n"),counter;
    counter++;  
}
mathline.c demo