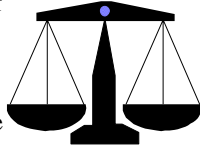


## Twenty Questions (more or less)

I have three coins, but one of them is fake!

The fake is heavier than the others.

You have a scale. What's the smallest number of weighings you can use to find the fake?



F-1  
4/6/01

CSE142  
Computer Programming I

## Conditionals

...or if it's true, go ahead you!  
if it's false, fall to else.

F-2  
4/6/01

## Finding the Fake Coin

Weigh two coins against each other:

- if the left one is heavier, it's the fake
- otherwise, if the right is heavier, it's the fake
- otherwise, the remaining one is the fake

*How can we do this in C?*

F-3  
4/6/01

## Overview

Conditional execution

if statement

A strange bit of syntax: {Compound statements}

Conditional expressions

Relational and logical operators

F-4  
4/6/01

## Related Reading

Read Sections 4.1-4.5, 4.7, 4.9

- 4.1: Control structure preview
- 4.2: Relational and logical operators
- 4.3: if statements
- 4.4: Compound statements
- 4.5: Example (uses some future concepts)
- 4.7: Nested if statements

F-5  
4/6/01

## Control Flow of our Scales

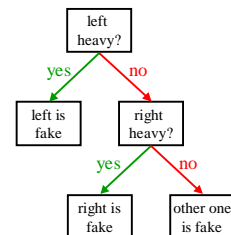
Here's our algorithm:

*control* of the process...

*flows* from box to box.

The algorithm is clearly stated and deterministic.

The computer should be able to do it!



F-6  
4/6/01

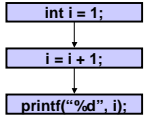
## Control Flow

“Control flow” is the order in which statements are executed

Until now, control flow has been sequential:

the next statement executed is the next one that appears, in order, in the C program

```
{
  int i = 1;
  i = i + 1;
  printf("%d", i);
}
```

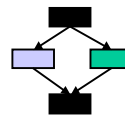


*But... what are those {}? Let's come back to that!*

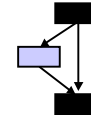
F-7  
4/6/01

## Conditional Control Flow

choosing which of two (or more) statements to execute before continuing



choosing whether or not to skip a statement before continuing



F-8  
4/6/01

## Conditional Execution

Conditional statements allow the computer to choose an execution path depending on the value of a variable or expression

- if the withdrawal is more than the bank balance, then print an error
- if today is my birthday, then add one to my age
- if it's a 9:30 class, prop your eyelids open; otherwise (it's 11:30), gnaw on your arm while you wait for lunch.

F-9  
4/6/01

## “Compound statements”

Before we get into writing conditionals in C...

Let's look at an *apparently* unrelated bit of syntax, the “compound statement”.

```
{
  statement1 ;
  statement2 ;
  ...
}
```

**Groups statements so that they are treated as a single statement:**

**Indicates sequential control flow!**

*Also called a “block.”*

F-10  
4/6/01

## You've seen this before...

```
int main(void) {
  printf(“Hello, world!\n”);

  return 0;
}
```

Now, detour over. But keep this in mind.

F-11  
4/6/01

## Combining and Substituting Statements

You may use a compound statement anywhere that a single statement may be used.

Anywhere that a statement is allowed in C, any kind of statement can be used.

A compound statement can contain any number of statements (including 0).

Among other things, these principles imply that compound statements can be *nested* to any depth.

*“nested” means “put inside one another”.*

F-12  
4/6/01

## Conditional ("if") Statement

```
if (condition)
    statement;
```

The **statement** is executed if the **condition** is true.

Otherwise, the **statement** is skipped!

```
if (withdrawalAmount > balance)
    printf("Not enough money\n");
```

```
if (x < 100)
    x = x + 1;
```

```
if (temperature > 98.6)
    printf("You have a fever.\n");
    printf("Go see the doc.\n");
```

*WAIT! There's something wrong with the last one. What?*

F-13  
4/6/01

## Blocks are Back!

To perform multiple statements conditionally, we use a compound statement!

```
if (condition) {
    statement1;
    statement2;
    ...
}
if (temperature > 98.6) {
    printf("You have a fever.\n");
    printf("Go see the doc.\n");
}
```

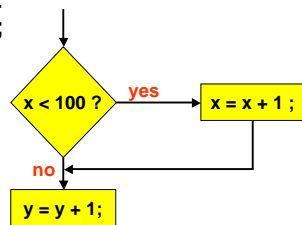
If **condition** is true, all **statements** between the braces are executed.

*As a point of style, we will ALWAYS use the braces for conditionals!!!*

F-14  
4/6/01

## Conditional Flow Chart

```
if (x < 100) {
    x = x + 1;
}
y = y + 1;
```



F-15  
4/6/01

## Conditions

In parentheses is a condition, also called a "logical" or "Boolean" expression

Made up of variables, constants, arithmetic expressions, and the relational operators

Math symbols: <, ≤, >, ≥, =, ≠  
in C: <, <=, >, >=, ==, !=

*What should we call ==?*

F-16  
4/6/01

## Conditional Expressions

```
air_temperature > 80.0
98.6 <= body_temperature
marital_status == 'M'
divisor != 0
```

Such expressions are used in "if" statements and numerous other places in C.

F-17  
4/6/01

## Value of Conditional Expressions

What is the value of a conditional expression??

Answer: we think of it as TRUE or FALSE

Under the hood in C, it's really an integer

FALSE is 0 (and 0 is FALSE)

TRUE is 1 (and 1 is TRUE)

TRUE is also any other non-zero value...

But relational ops will always give 1 for TRUE

(e.g., 4 < 7 evaluates to 1)

F-18  
4/6/01

## Complex Conditionals

if I have at least \$15 *or* you have at least \$15,  
then we can go to the movies  
if the temperature is below 32 degrees *and* it's  
raining, then it's snowing  
if it's not the case that it's Saturday *or*  
Sunday, then it's a work day

F-19  
4/6/01

## Complex Conditionals in C

C represents these with "Boolean" operators.

**Boolean operators:**    **&&**    **||**    **!**  
                                  **and**    **or**    **not**

```
#define TRUE 1
#define FALSE 0

if (myMoney >= 15.0 || yourMoney >= 15.0) {
    canGoToMovies = TRUE;
}
```

*More on these later!*

F-20  
4/6/01

## History Break

I guess I was  
a *bit* ahead  
of my time.



When we write conditions, we use "Boolean algebra", the symbolic representation of logic.

This algebra is at the heart of *everything* computers do! *Every* operation is eventually calculated in terms of Boolean algebra.

Guess when George Boole invented it...

*Almost 150 years ago,  
in 1854.*

Picture/autobiography thanks to math dept.  
U. of St. Andrews, Scotland.

F-21  
4/6/01

## Finding Absolute Value

Problem: Compute the absolute value  $|x|$  of  $x$

Put the answer in variable `abs`.

```
if (x >= 0) {          abs = x;
    abs = x;          } if (x < 0) {
}                    }   abs = -x;
if (x < 0) {
    abs = -x;
}
```

```
if (x >= 0) {
    abs = x;
} else {
    abs = -x;
}
```

*Which of these is right?*

F-22  
4/6/01

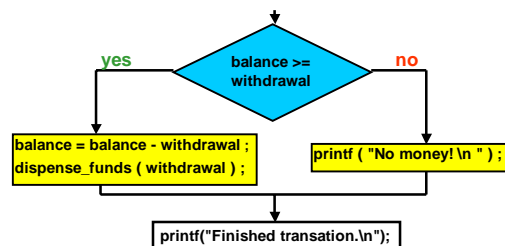
## if - else

Example: print error message if condition is false:

```
if ( balance >= withdrawal ) {
    balance = balance - withdrawal ;
    dispense_funds ( withdrawal ) ;
} ← no ; here!!
else {
    printf ( "Insufficient Funds! \n " ) ;
}
printf("Finished transaction.\n");
```

F-23  
4/6/01

## if-else Control Flow



F-24  
4/6/01

## Nested if statements

```

if ( x == 5 ) {
  if ( y == 5 ) {
    printf ( "Both are 5. \n " );
  }
  else {
    printf ( "x is 5, but y is not. \n " );
  }
}
else {
  if ( y == 5 ) {
    printf ( "y is 5, but x is not. \n " );
  }
  else {
    printf ( "Neither is 5. \n " );
  }
}

```

*Any statement can go inside an if statement.*

*Therefore, an if statement can go inside an if statement.*

F-25  
4/6/01

## Tax Table Example

Problem: Print the % tax based on income:

income	tax
< 15,000	0%
15,000, < 30,000	18%
30,000, < 50,000	22%
50,000, < 100,000	28%
100,000	31%

F-26  
4/6/01

## Direct Solution

```

if ( income < 15000 ) {
  printf ( "No tax." );
}
if ( income >= 15000 && income < 30000 ) {
  printf ( "18%% tax." );
}
if ( income >= 30000 && income < 50000 ) {
  printf ( "22%% tax." );
}
if ( income >= 50000 && income < 100000 ) {
  printf ( "28%% tax." );
}
if ( income >= 100000 ) {
  printf ( "31%% tax." );
}

```

**Mutually exclusive conditions - only one will be true.**

F-27  
4/6/01

## Cascaded ifs

```

if ( income < 15000 ) {
  printf ( "No tax." );
} else {
  if ( income < 30000 ) {
    printf ( "18%% tax." );
  } else {
    if ( income < 50000 ) {
      printf ( "22%% tax." );
    } else {
      if ( income < 100000 ) {
        printf ( "28%% tax." );
      } else {
        printf ( "31%% tax." );
      }
    }
  }
}

```

**Order is important. Conditions are evaluated in order given.**

F-28  
4/6/01

## Warning: Danger Ahead

The idea of conditional execution is natural, intuitive, and highly useful

However...

Programs can get convoluted and hard to understand

There are syntactic pitfalls to avoid

F-29  
4/6/01

## Pitfalls of if: The World's Last C Bug

```

status = check_radar ( ) ;
if ( status == 1 ) {
  launch_missiles ( ) ;
}

```

**Bug! = is used instead of ==**

This is *not* a syntax error, so the compiler will not report any errors and the program can execute

F-30  
4/6/01

## Pitfalls of if, Part II

---

No:

```
if ( 0 <= x <= 10 ) {  
    printf ( "x is between 0 and 10. \n " );  
}
```

Yes:

```
if ( 0 <= x && x <= 10 ) {  
    printf ( "x is between 0 and 10. \n " );  
}
```

*and*

F-31  
4/6/01

## Pitfalls of if, Part III

---

& is different from &&

| is different from ||

- & and | are not used in this class, but are legal C
- If used by mistake, no syntax error, but program may produce bizarre results

F-32  
4/6/01

## Pitfalls of if, Part IV

---

Beware == and != with doubles:

```
double x ;  
x = 30.0 * (1.0 / 3.0) ;  
if ( x == 10.0 ) ...
```

*Remember! doubles are not exact!*

F-33  
4/6/01

## Next Time

---

We'll be discussing functions, a major topic of the course

Many students find it intellectually challenging compared to the previous material.

F-34  
4/6/01

## QOTD Tomb Raider: Find and Destroy the Dead Code!

---

"Dead" code is code that will never be executed no matter how you run the program.

Find the "dead" code in the following...

```
if (speed >= 0) {  
    printf("You don't go backward.\n");  
    if (speed == -1) {  
        printf("Wait! I was wrong!");  
    }  
} else if (speed > 0) {  
    printf("You go forward.\n");  
} else if (speed < 0) {  
    printf("You go backward.\n");  
} else {  
    printf("What did you do?\n");  
}
```

*Note: we put the "else" on the same line as the "}" here. That's inconsistent with what we did elsewhere (bad style!). We did it b/c this had to fit on a slide.*

F-35  
4/6/01