

## Basic Programming II

MVRT – 115  
Oct. 12, 2006

## if Statements

- In the line-following example, you were introduced to if statements
- The elementary form is

```
if (condition)
{
    ... statements
}
```
- The statements are performed if "condition" is true, otherwise the statements between { and } are skipped.

## Conditions

- In the if (condition) clause, "condition" can be any legal expression
  - if (1) // always true
  - if (x > 0) // only true when x is positive
  - if (sin(w)) // true except at 0° & 180°
  - if (x > 0 && y > 0) // true when in 1<sup>st</sup> quadrant
- You can (but shouldn't) make extremely complex conditions in your if statements

## What is Truth, Anyway?

- Defining philosophical truth is difficult; C programming, however, truth is easy to define.
  - A condition is true if it evaluates to non-zero.
  - A condition is false if it evaluates to zero
- This can take a little getting used to, but basically all C expressions must evaluate to a number

## Zero or Non-zero

- Let's look at some conditions
  - if (1)
  - if (-1)
  - if (0)
  - if (x > 3)
  - if ((x < 3 \* v && r < 1) || 1)
- Because expressions evaluate to a number, you can (but shouldn't) do some weird things:
  - j = h > q; // if h > q then assign 1 to j, else assign 0
  - x = !x; // if x ≠ 0 assign 0 to x, else assign 1
  - x = !(!x);

## if and if - else

- Looking at the line-follower code, each of the 3 "if" statements can be true or false and may be executed in any combination
- But if you want to test for exclusive conditions, use "else" clauses

```
if (condition 1) // Order is important for logic and efficiency
{
  ...
}
else if (condition 2)
{
  ...
}
else if (condition 47)
{
  ...
}
else
{
  ... // come here when all the other conditions are false
}
```

## Switch Statement

- If you have many conditions, consider using a switch (case) construct

```
#define FORWARD 483
switch (condition)
{
  case 1: // must be number or constant expression
  ...
  break; // if you leave out break, the next case executed: BAD!
  case FORWARD:
  ...
  break;
  default:
  ...
}
```