Variable Names

* letters and digits
* starts with letter or - (library routines use to start var names)
* case-sensitive.
* lower case
* UPPER CASE - Symbolic Constants.
* Keywords - if, else, while, etc. can't be used.
* name your variables wisely!

Data Types and Sizes

{ char, int, float, double }

only 4 basic data types in C!

short, long

{ qualifiers.

eq. short int sh; long int counter;

( or )

short sh;
long counter;
Size of data types on "most" 32-bit machines:

- char: 1 byte
- short: 2 bytes
- int: 2 or 4 bytes
- long: 4 bytes

The compiler is free to choose appropriate sizes for its own HW.

Only Restriction:

\[
\text{short} \leq \text{int} \leq \text{long} \quad \text{at least 16 bits} \leq \text{at least 16 bits} \leq \text{at least 32 bits}
\]

Signed or Unsigned (Qualifiers)

Applied to: char! or ints.

Unsigned numbers - obey modulo \(2^n\) arithmetic.

#ade1

SHOW CODE IN CLASS

#Homework

default qualifier for char

machine-dependent

HW 1 - Find the qualifier for plain char in your machine.
long - can also be applied to double.
=> long double.

headers -> <limits.h> and <float.h>

eg. INT_MAX = +32767
     INT_MIN = -32767

     UINT_MAX = 65535

HW2 - Find the values for the following constants:

CHAR_MAX, CHAR_MIN, LONG_MAX, LONG_MIN,
SHORT_MAX, SHORT_MIN.

3. Constants

   1234 = int

   123456789 \(L\) = long
   (or)

   123456789 \(L\) = long

Floating point constants

123.45 or 1e-2

default = double

f or F = float

L or l = long double

u or U = unsigned constant

eg. 707 u

ul or UL = unsigned long constant

eg. 230223032362UL

23022362UL
Octal and hexadecimal

\[ 31_{10} = 037 = OX1F \]

\( \text{zero (octal)} \rightarrow \text{zero } x \text{ (hexadecimal)} \)

\( OXFUL = 15_{10} \) (unsigned long)

Character constant

\( 'x' \rightarrow \text{value is ASCII value of } x \)

\[ \text{eq.} \quad '0' = 48 \quad | \quad 'a' = 97 \quad | \quad 'A' = 65 \]

* Character constants participate in numeric operations just as any other integers.

# Code 2 - Show code for counting the characters in a string of char (line of text).

Escape Sequences

\( \backslash n \) - newline
\( \backslash t \) - horizontal tab
\'0' \( \rightarrow \) character with value zero (null character)

\( '0' = 0 \) (\'0\' is preferred)
Constant expression

```c
#define MAXLINE 1000
char line[MAXLINE + 1];
```

String constant or string literal

"Let it be!" or "" (empty string)

```c
Let it be!
```

`strlen - <string.h>`

```c
\text{\texttt{X}} = 88 \text{ (ASCII value of \texttt{X})}
```

```c
\text{\texttt{X}} = \texttt{X}'\texttt{\backslash\'O}\
```

Integer used to represent \texttt{X} in machine's char set.

Enumeration constant

```c
enum boolean {{NO, YES}};
```

Array of chars.

(Alternatives to `#define`)

```c
enum months {JAN = 1, FEB, MAR, APR, NOV, DEC};
```
# Declarations

```c
int lower;
int integers[100];
int upper = 1000;
char c;
char line[1000];
```

## Initialization

- **External or static variables** — initialized to zero (by default)
- **Automatic (local) variables** — NOT initialized

Contains GARBAGE VALUE!

## Code 4 - Show this in class.

```c
"const" qualifier
```

```c
const double e = 2.71828182845905;
const char msg[] = "Warning: ";
```