Types

CS 217

Types

- The type of an object determines…
  - the values it can have
  - the operations that can be performed on it
- Base types

`char`	a character, typically a byte
`int`	an integer; typically a word
`float`	single-precision floating point
`double`	double-precision floating point

Type Qualifiers

- Length qualifiers
  - `short int` (smaller; 16-bits on 32-bit machine)
  - `long int` (larger; 32-bits on 32-bit machine)
- Unsigned integers
  - `unsigned int`
  - `unsigned short int`
  - `unsigned char`
- Constant (read-only) variables
  - `const double pi = 3.14159`

Constant Expressions

- Evaluated at compile time
  ```
  int p = 1 – 1;
  ```
- Use constant expressions…
  - to reduce the number of **#define** constants
  - to increase readability
  - to improve changeability; e.g.,
    ```
    #define MAXLINE
    …
    char buf[2*MAXLINE + 1];
    ```


Types of Constants

char	**'a'**	character constant (single quote)
	'\035'	character code **35** octal
	'\x29'	character code **29** hexadecimal
	'\t'	tab **('\011',** do **man ascii)**
	'\n'	newline **('\012')**
	'\0'	null character
int	**156**	decimal constant
	0234	octal
	0x9c	hexadecimal
long	**156L**	
	156l	don't do it
float	**15.6F**	
	15.6f	
double	**15.6**	defaults to **double**
	15.6L	
	15.6l	

Arrays

- Array declarations specify the number of elements, not an upper bound on the index
  ```
  int digits[10];
  ```
  says that **digits** is an array of **10 int**s
  ```
  digits[0], digits[1], … digits[9]
  ```
- Array may be indexed by integer expression
  ```
  digits[f(x)/2 + BASE]
  ```
- No bounds checking!

Arrays (cont)

- Multi-dimensional arrays
  `float matrix[3][4][5]`
  is a 3-dimensional array w/ 3x4x5=60 elements
- Arrays are stored in row-major
  `matrix[0][0][0], matrix[0][0][1], …`
  last subscript varies the fastest

Strings & Initialization

- Arrays of characters
  `"hello\n"` | h | e | l | l | o | \n | \0 |
  the compiler always provides a terminating `\0`
- Length can be derived from initialization
  ```
  char s[] = "hello\n";
  ```
  is equivalent to
  ```
  char s[7] = "hello\n";
  char s[7] = {'h','e','l','o','\n','\0'};
  ```


Initialization (cont)

- Ditto for arrays…

```
int x[] = { 1, 2, 3 };
int y[][3] = {
    { 1, 3, 5 },
    { 2, 4, 6 },
    { 3, 5, 7 },
    { 4, 6, 8 }
};
```

Enumerations

- Associate constant values with identifiers

```
enum boolean { FALSE, TRUE };
enum color { RED, BLUE, GREEN };
```

- **`enum`** identifiers are **`int`** constants
- Values can be specified

```
enum escapes {TAB='\t',BACKSPACE='\b'};
enum months {Jan=1, Feb, Mar, Apr, May,
  June, Jul, Aug, Sep, Oct, Nov, Dec};
```