Function Pointers

Function Pointers Topics

- Functions
- Pointers to Functions
- Passing Function Pointers
- Returning Function Pointers

Functions

- Valid operations on the NAME of a function
  1. call the function by appending to the name a list of args
  2. assign the function to a variable of the appropriate type
  3. use the function name as a parameter to another function, in which case a pointer to the function is passed
- These are explicitly forbidden
  1. sizeof operator ==> sizeof(func(3));
  2. a function cannot return a function

Functions

- Left-To-Right Rule
  - *type name*(parms);
  - Start w/ *name* ➔ "*name* is a …"
  - Move right finding (*parms*) ➔ " … function taking *parms* …"
  - At the ';', return to start of line ➔ "… and returning *type*."

Pointers to Functions

- Function Pointers
  - Variables providing indirection and abstraction to functions
  - The assumption is often made (incorrectly) that function pointers point to the function's code in memory
  - Usually, a function pointer points to a block of information needed to invoke the function
  - A function pointer is not a variable in the same sense as other pointers and cannot take part in pointer arithmetic

Pointers to Functions

- Left-To-Right Rule
  - *type* (**name*) (*params*);
  - Start w/ name ➔ "*name* is a …"
  - Move right finding ')'.
  - Move left to matching '('.
  - Move right finding '*' ➔ " … a pointer to a …"
  - Move right finding (*parms*) ➔ " … function taking *params*…"
  - At the ';' return to start of line.
  - Continue right ➔ "… and returns *type*"

Pointers to Functions

- Example
  - http://faculty.washington.edu/sproedp/advc/csamples/less9-1.c.html

Passing Function Pointers

- Left-to-Right Rule
  - *type name(type_p* (**name_p*) (*params*));
  - Start w/ name ➔ "*name* is a …"
  - Move right finding '(…)' ➔ "*function that takes a …*"
  - *Type_p* (**name_p*) (*params*) ➔ "*function pointer takes a …*"
  - At the ';' return to start of line.
  - Continue right ➔ "… and returns *type*"

Passing Function Pointers

- Example
  - http://faculty.washington.edu/sproedp/advc/csamples/less9-2.c.html

Returning Function Pointers

- Left-to-Right Rule
  - *type*(**name* (*parmsf*))(*parmsp*) ;
  - Start w/ *name*, move right ➔ "*name* is a function taking *parmsf and returning …"*
  - Move past *(parmsf)*
  - Continue right finding ')'.
  - Move left to matching '('.
  - Move right finding '*' ➔ " … a pointer to a …"
  - Move right finding (*parms*) ➔ " … function taking *params*…"
  - At the ';' return to start of line.
  - Continue right ➔ "… and returns *type*"

Returning Function Pointers

- Example
  - http://faculty.washington.edu/sproedp/advc/csamples/less9-3.c.html