C Function Pointers: The Basics

COSC 341/342:

Programming Languages, WI 2010

Bottom line

A function pointer is a pointer to compiled function code.

```
#include <stdio.h>
int addi (int x, int y) { return x+y; }
int subi (int x, int y) { return x-y; }
int useFunctionPtr ( int a, int b,
                     int (* fun ) (int, int)) {
  return fun(a, b);
  }

int main () {
  int x = 3, y = 5, result;
  result = useFunctionPtr (x, y, &addi);
  printf("%d + %d = %d\n", x, y, result);

  result = useFunctionPtr (x, y, &subi);
  printf("%d - %d = %d\n", x, y, result);
  }
```

Declaring and using a function pointer

```
#include <stdio.h>

int addi (int x, int y) { return x + y; }
int subi (int x, int y) { return x - y; }

int main () {
  int x = 3, y = 5;

  int (*func) (int, int);    // func is a pointer to a
                             // function returning int
  /* calculating x + y */
  func = &addi;
  printf("%d + %d = %d\n", x, y, (*func)(x, y));

  /* calculating x - y */
  func = &subi;
  printf("%d - %d = %d\n", x, y, (*func)(x, y));
  }
```

Another example, declaring & using

```
#include <stdio.h>
#include <stdlib.h>

int addi (int x, int y) { return x+y; }
int subi (int x, int y) { return x-y; }

int main () {
  int x, y, result;
  char op;
  int (*func) (int, int);

  printf("Enter int op int, no blanks (e.g., 3+144)");
  scanf("%d%c%d", &x, &op, &y);
  printf("You entered %d %c %d\n", x, op, y);
  if (op == '+') func = &addi;
  else if (op == '-') func = &subi;
  else exit(1);
  result = (*func)(x, y);
  printf("%d %c %d= %d\n", x, op, y, result);
  }
```

Using an array of function pointers

```
#include <stdio.h>

int addi(int x, int y) { return x+y; }
int subi(int x, int y) { return x-y; }

int main () {
  int i, x, y, result;
  char op;
  int (*funcArr[2]) (int, int); // an array of function ptrs
  funcArr[0] = &addi;
  funcArr[1] = &subi;
  int dataArr[4][2] = { {1, 10}, {2, 20}, {3, 30}, {4, 40}};
  for (i=0; i<4; i++) {
    x = dataArr[i][0];
    y = dataArr[i][1];
    op = (i%2)? '-' : '+';
    result = (*funcArr[i%2])(x, y);
    printf("%d %c %d = %d\n", x, op, y, result);
    }
  }
```

Nice tutorial

`http://www.newty.de/fpt/`