

Computer Science & Engineering 150A

Problem Solving Using Computers

Mini Lecture – Pointer Review

Christopher M. Bourke

Spring 2009

- Recall that *pointers* are references to memory addresses
- Regular variable declaration:
`int k;`
- Pointer variable declaration:
`int *ptr;`
- `&k` - the memory address of `k`: “referencing”
- `*k` - treat the value stored in `k` as a memory address (**illegal**)
- `*ptr` - access the value stored at the memory address `ptr`
“dereferencing”

Quiz 3 Program 1

CSCE150A

Introduction

Program
Example

```
1  int function01(int a, int *b);
2  int function02(int *a, int b);
3
4  int main(void) {
5      int x = 10;
6      int y = 20;
7      int z = 0;
8
9      printf("x = %d, y = %d, z = %d\n",x,y,z);
10     z = function01(x,&y);
11     printf("x = %d, y = %d, z = %d\n",x,y,z);
12     z = function02(&x,y);
13     printf("x = %d, y = %d, z = %d\n",x,y,z);
14
15     return 0;
16 }
17
```

```
18 | int function01(int a, int *b) {
19 |     a = 5;
20 |     *b = 10;
21 |     return a + (*b);
22 | }
23 |
24 | int function02(int *a, int b) {
25 |     *a = 100;
26 |     b = 50;
27 |     return (*a) - b;
28 | }
```

```
int function01(int a, int *b);
```

- Function returns an integer *value*
- `a` is passed *by value*
- `b` is passed *by reference*

```
int function02(int *a, int b);
```

- Function returns an integer *value*
- `a` is passed *by reference*
- `b` is passed *by value*

Passing by value:

- A *copy* of the value of the variable is placed at the top of the memory stack
- Function can change the value *only of this copy*
- The original value of the variable is unchanged in the calling function

Passing by reference:

- The memory address of a variable is passed
- Function can change the value *globally*
- Changes in the function are reflected in the calling function

```
1   int x = 10;  
2   int y = 20;  
3   int z = 0;  
4  
5   printf("x = %d, y = %d, z = %d\n",x,y,z);
```

Output: `x = 10, y = 20, z = 30`

```
1  z = function01(x,&y);  
2  printf("x = %d, y = %d, z = %d\n",x,y,z);
```

- `x` is passed by value
- `y` is passed by reference

```
1  int function01(int a, int *b) {  
2      a = 5;  
3      *b = 10;  
4      return a + (*b);  
5  }
```

- Value of `x` is only changed locally
- Value of `y` is changed globally
- Return value: $5 + 10$
- Output: `x = 10, y = 10, z = 15`

```
1  z = function02(&x,y);  
2  printf("x = %d, y = %d, z = %d\n",x,y,z);
```

- `x` is passed by reference
- `y` is passed by value

```
1  int function02(int *a, int b) {  
2      *a = 100;  
3      b = 50;  
4      return (*a) - b;  
5  }
```

- Value of `x` is changed globally
- Value of `y` is only changed locally
- Return value: $100 - 50$
- Output: `x = 100, y = 10, z = 50`