CMSC202 Computer Science II for Majors

Lecture 05 – References

Dr. Katherine Gibson

Based on slides by Chris Marron at UMBC

Last Class We Covered

- Variables
 - Values
 - Addresses
- Pointers
 - Creating
 - Initializing
 - Dereferencing
- Pointers and Functions
 - "Returning" more than one value

AN HONORS UNIVERSITY IN MARYLAND

Any Questions from Last Time?

Today's Objectives

• To review and better understand pointers

- To discuss how pointers are used to pass entire arrays to functions
- To learn about references



AN HONORS UNIVERSITY IN MARYLAND

Review of Pointers

AN HONORS UNIVERSITY IN MARYLAND

variable name		
memory address		
value		

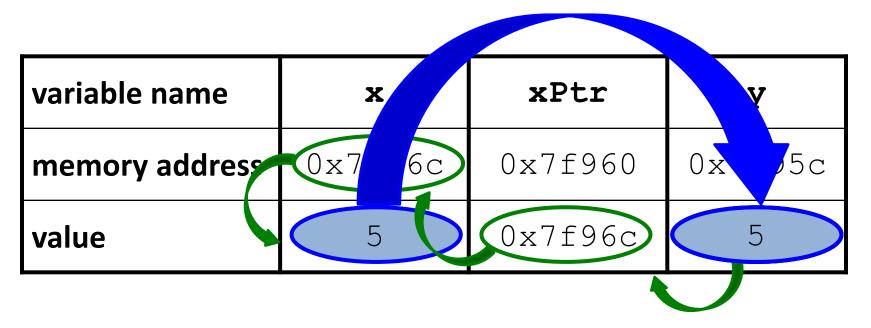
AN HONORS UNIVERSITY IN MARYLAND

variable name	x	xPtr	У
memory address	0x7f96c	0x7f960	0x7f95c
value	5	0x7f96c	?

AN HONORS UNIVERSITY IN MARYLAND

variable name	x	xPtr	У
memory addres	0x7f96c	0x7f960	0x7f95c
value	5	0x7f96c	?

AN HONORS UNIVERSITY IN MARYLAND



AN HONORS UNIVERSITY IN MARYLAND

variable name	x	xPtr	У
memory address	0x7f96c	0x7f960	0x7f95c
value	3	0x7f96c	2

AN HONORS UNIVERSITY IN MARYLAND

variable name	x	xPtr	У
memory address	0x7f96c	0x7f960	0x7f95c
value	3	0x7f96c	2



Pointers and Arrays and Functions

 Because arrays are pointers, they are <u>always</u> passed by address to a function

- What does this mean?
 - Program does <u>not</u> make a copy of an array
 - Any changes made to an array inside a function will remain after the function exits

 Passing one element of an array is still treated as pass by value

- For example
 - -classNums[0] is a single variable of type int, and is passed to the function by value
 - classNums is an array, and is passed to the function by its address

• Reminder!

AR

- C-style strings are arrays of characters
- So functions always pass C-Strings by...
 Address!
- Pass to a function by name only
 Just like any other array

- In a function prototype, that looks like this:
 - /* function takes a char pointer */
 void toUpper (char *word);
 char str[] = "hello";
 toUpper (str);

This is also a valid function prototype:
 void toUpper (char word[]);



AN HONORS UNIVERSITY IN MARYLAND

Passing Variables: 3 Options

AN HONORS UNIVERSITY IN MARYLAND Review: Passing by Value

• The "default" way to pass variables to functions

// function prototype
void printVal (int x);

int x = 5; int *xPtr = &x; printVal(x); // function call printVal(*xPtr); // also valid call

AN HONORS UNIVERSITY IN MARYLAND Review: Passing by Address

• Uses pointers, and uses * and & operators

// function prototype
void changeVal (int *x);

AN HONORS UNIVERSITY IN MARYLAND Third Option: References

- References are
 - -Safer than pointers
 - -Less powerful
 - More restricted in usage
- Use the ampersand (&) for declaration
 int &xRef = x;

References

- Once created, references don't need to use the ampersand or asterisk
 - -They look like "normal" variables
 - -But behave (somewhat) like pointers
- References **must** be initialized at declaration
- References **cannot** be changed
- References can be treated as another "name" for a variable (no dereferencing)

 Functions that take in references (instead of addresses) look almost identical to functions that take in "normal" values

```
void changeByRef (int &x) {
    x = x + 1;
}
```

• Prototype changes, but function body looks like that of a function that takes in a value

AN HONORS UNIVERSITY IN MARYLAND Calling Reference Functions

• Calling also looks similar to functions "by value"

void changeByRef(int &x); //prototype

int x = 5; int &xRef = x; //create reference changeByRef(x); //function call changeByRef(xRef); //also valid call • References are static

 Once initialized, they are forever tied to the thing that they reference

- Using them looks identical to using a value
 That's a good thing though? It's easier!
 - -But it can also be confusing
 - May think you're passing by value, and that the contents of the variable won't be changed



AN HONORS UNIVERSITY IN MARYLAND

LIVECODING!!!

- Project 1 has been released
- Found on Professor's Marron website
- Due by 9:00 PM on February 23rd
- Get started on it now!
- Next time: Classes and Objects

 Write a function called makeChange() that takes in a value in cents, represented as an int and then calculates the number of quarters, dimes, nickels, & pennies needed for change

- The function can take in multiple arguments
- The function does not return anything
- The cents value is guaranteed to be correct

- A valid integer, positive, etc.