ANSI C (Class 2)

Data Analysis in Geophysics

Bob Smalley

Demián D. Gómez

November 2013

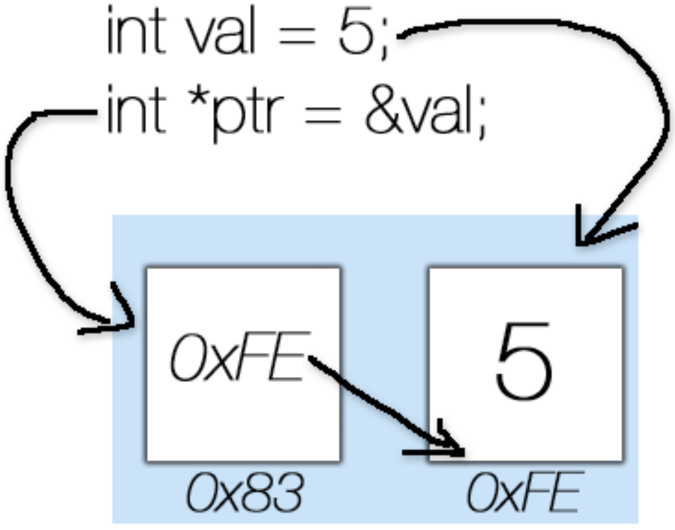
Advanced Stuff (I)

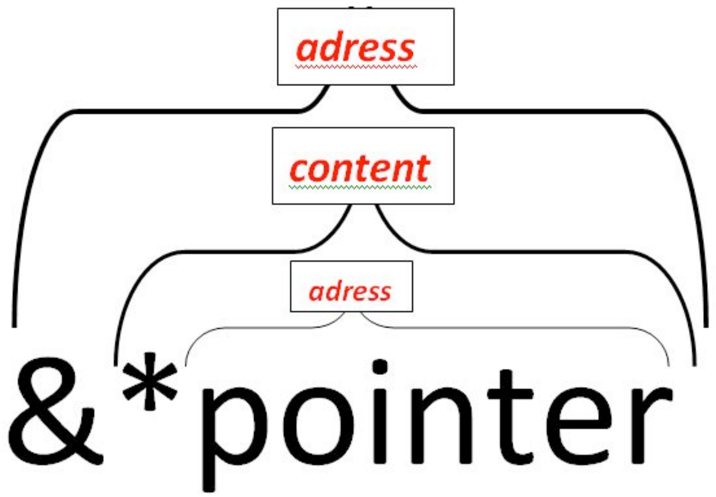
- Pointers: indirect addressing technique to read and write data from the computer memory. They are declared with an *

```
int *A=0;
double *vector=0;
```

Advanced Stuff (II)

- Address of operator:





Pointer to ***char*** Example

```
#include <iostream>
#include <stdio.h>
using namespace std;

int main()
{
	int *a=0;
	const char *str="Hi there! This is a pointer test.";

	for (int i=0; i<strlen(str); i++)
	{
		cout << str[i];
	}
	cout << endl;

	cout << "What is this value?: " << &str << endl;
	printf("Address of contents: %p\n", str);
	printf("Address of variable: %p\n", &str);

	return 0;
}
```

Run the code

- Look for the code in:
  - ✓ pod0/ddgomez/public/C examples/strings_1.cpp

- Open it in Xcode and try to understand its contents.
- Open a Terminal and compile it.
- Run the code to see the result.
- What numbers are you getting in “Address of contents:” and “Address of variable:”?

Using iostream

- You can use ***cout*** to print stuff on the screen like this:

```
cout << “Hello World”;
```

- You can use ***cin*** to input data into your program, like this:

```
cin >> a >> b;
```

- This will input values to a and b variables.

Armadillo: Create a Matrix on the Fly

```
#include "../armadillo/include/armadillo"
#include <iostream>

using namespace arma;
using namespace std;

int main(){

                int rows,cols=0;

                cout << "Please enter a dimension for your matrix"<< endl;
                cin >> rows >> cols;

                cout << "The selected dimension is:" << rows << "x" << cols << endl;

                mat matrix_test(rows,cols);

                cout << matrix_test;
}
```

Compile: c++ main.c -o main.

What happened to the output of ***cout << matrix_test*** ?

More Armadillo stuff: Load a Matrix from a File

- Create a file called “mat.txt”
- Type:

```
1 2 3
4 5 6
```

- In your C++ program, add a new line:

```
matrix_test.load("mat.txt");
```

- Output the result using ***cout***
- Was your matrix the same size as the loaded data? If it wasn’t, Armadillo adapts the matrix size to match the loaded data.

Even More Armadillo stuff: Save a Matrix

- Use the ***save*** command to save a matrix to a file using Armadillo, like this:

```
matrix_test.save("mat2.txt");
```

- Open the file mat2.txt
- Can you understand its contents? Probably not, because it is in binary format. We will now save it in ASCII format.

Save a Matrix 2

- Now try the following:

```
matrix_test.save("mat2.txt", arma_ascii);
```

- Now the matrix has been saved as an ASCII file.

C++ Reference Website

- Reference guide
  - http://www.cplusplus.com/reference/
- Tutorial:
  - http://www.cplusplus.com/doc/tutorial/