



UNIVERSITY OF
RICHMOND

CMSC 240 Lecture 2

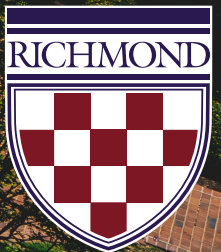
CMSC 240 Software Systems Development
Fall 2023

Today

- Some C++ Basics
- Intro to Unix/Linux
- Intro to Version Control
- Environment setup
- In-class coding exercise



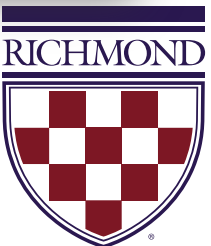
C++ Basics



Input From the Terminal

```
#include <iostream>
using namespace std;

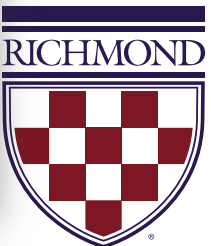
int main()
{
    cout << "Please enter your first name (followed by 'enter'):" << endl;
    string first_name;           // first_name is a variable of type string
    cin >> first_name;           // read characters into first_name
    cout << "Hello, " << first_name << "!" << endl;
}
```



YOU Kinda Already Know C++

- Conditionals: Use **if** and **else** with the same syntax

```
int random_number = rand();  
  
if (random_number >= 4) // Do something if condition1 is true  
{  
    cout << "It's greater than or equal to 4" << endl;  
}  
  
else if (random_number <= 2) // Do something if condition1 is false and condition2 is true  
{  
    cout << "It's less than or equal to 2" << endl;  
}  
  
else // Do this if both condition1 and condition2 are false  
{  
    cout << "It has to be 3!" << endl;  
}
```

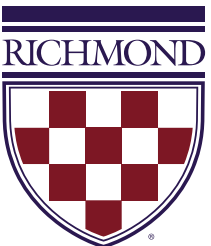


YOU Kinda Already Know C++

- Loops: Use **while** and **for** loops with the same syntax

```
int count_down = 10;
while(count_down > 0)
{
    // Run this as long as the condition is true
    cout << count_down << endl;
    count_down--; // Subtract one from count down
}

// Initialization ; Condition ; operation run on each iteration
for (int count_up = 1; count_up <= 10; count_up++)
{
    // Run this until the condition is false
    cout << count_up << endl;
}
```



YOU Kinda Already Know C++

- Basic Types: Use **int**, **float**, **double**, **char**
- Variable Declaration: You declare what each variable is

```
short eggs = 12;           // integer number : 2-bytes
int  number_of_steps = 3000; // integer number : 4-bytes
long  population = 4000000;  // integer number : 8-bytes
float temperature = 98.5;    // single-precision floating point : 4-bytes
double flying_time = 3.5;    // double-precision floating point : 8-bytes
char the_letter_a = 'a';     // char for individual characters
string name = "Annemarie";   // string for character strings
bool lights_on = true;       // bool for logical variables
```



YOU Kinda Already Know C++

- Functions/Methods: You describe a functions input/output

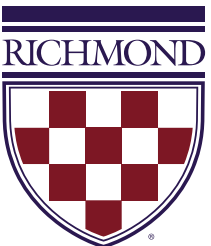
*// The square function will input an integer value
// and return the square of that integer value.*

```
int square(int value)  
{  
    return value * value;  
}
```

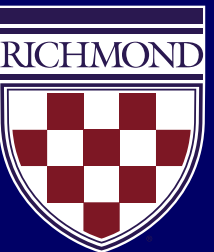

YOU Kinda Already Know C++

- Logical operators: == <= >= < > && || and ! all work the same

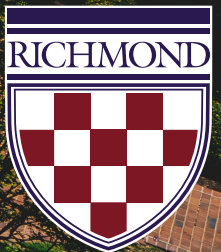
```
if(bool1 || bool2)
{
    cout << "bool1 OR bool2 is true." << endl;
}
if(bool1 && bool2)
{
    cout << "bool1 and bool2 is true" << endl;
}
if(bool1 == false)
{
    cout << "bool1 is false" << endl;
}
if(bool2 != true)
{
    cout << "bool2 is not true" << endl;
}
```



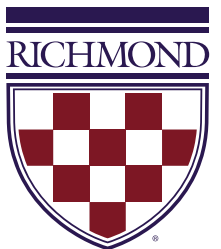
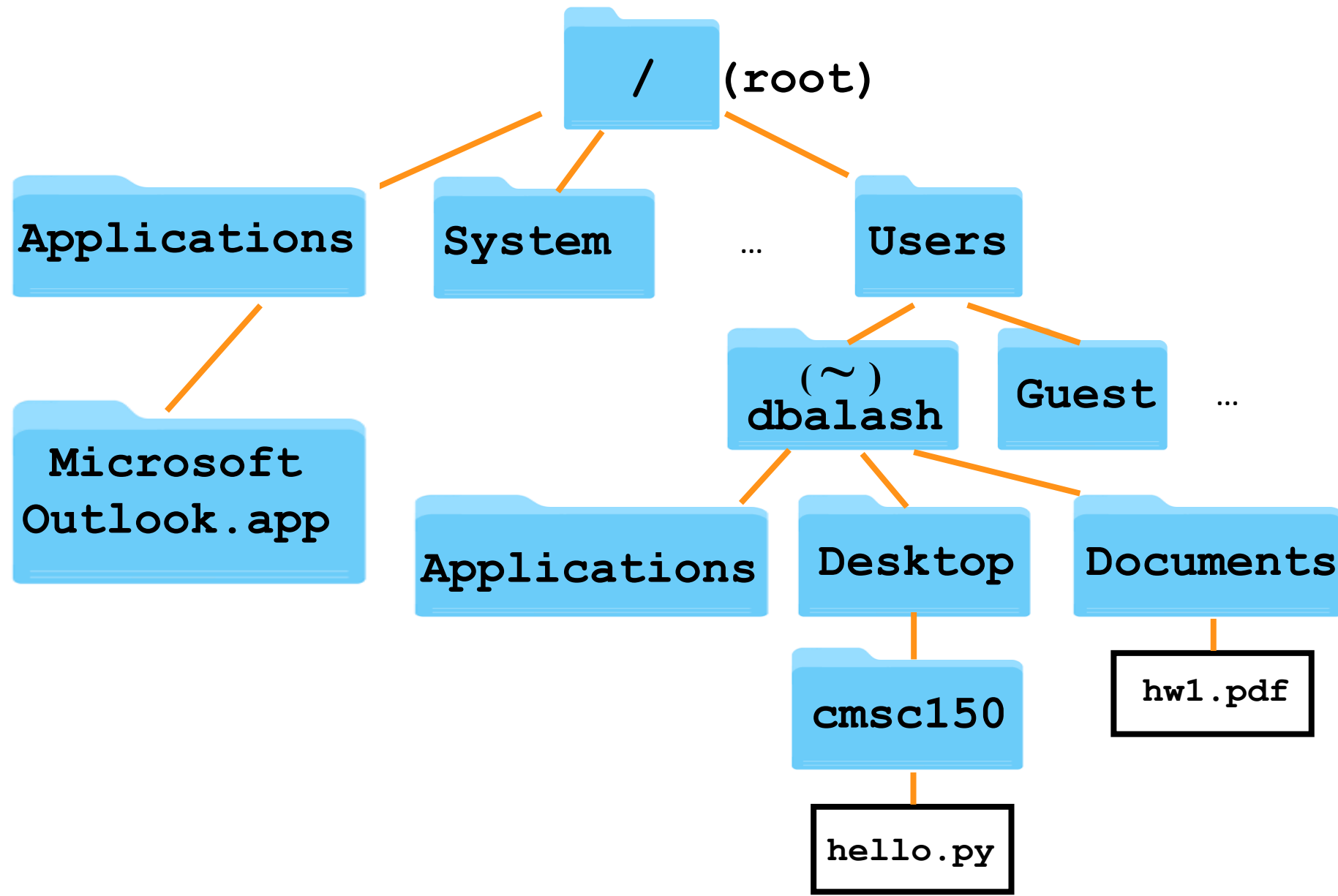
Ask me questions



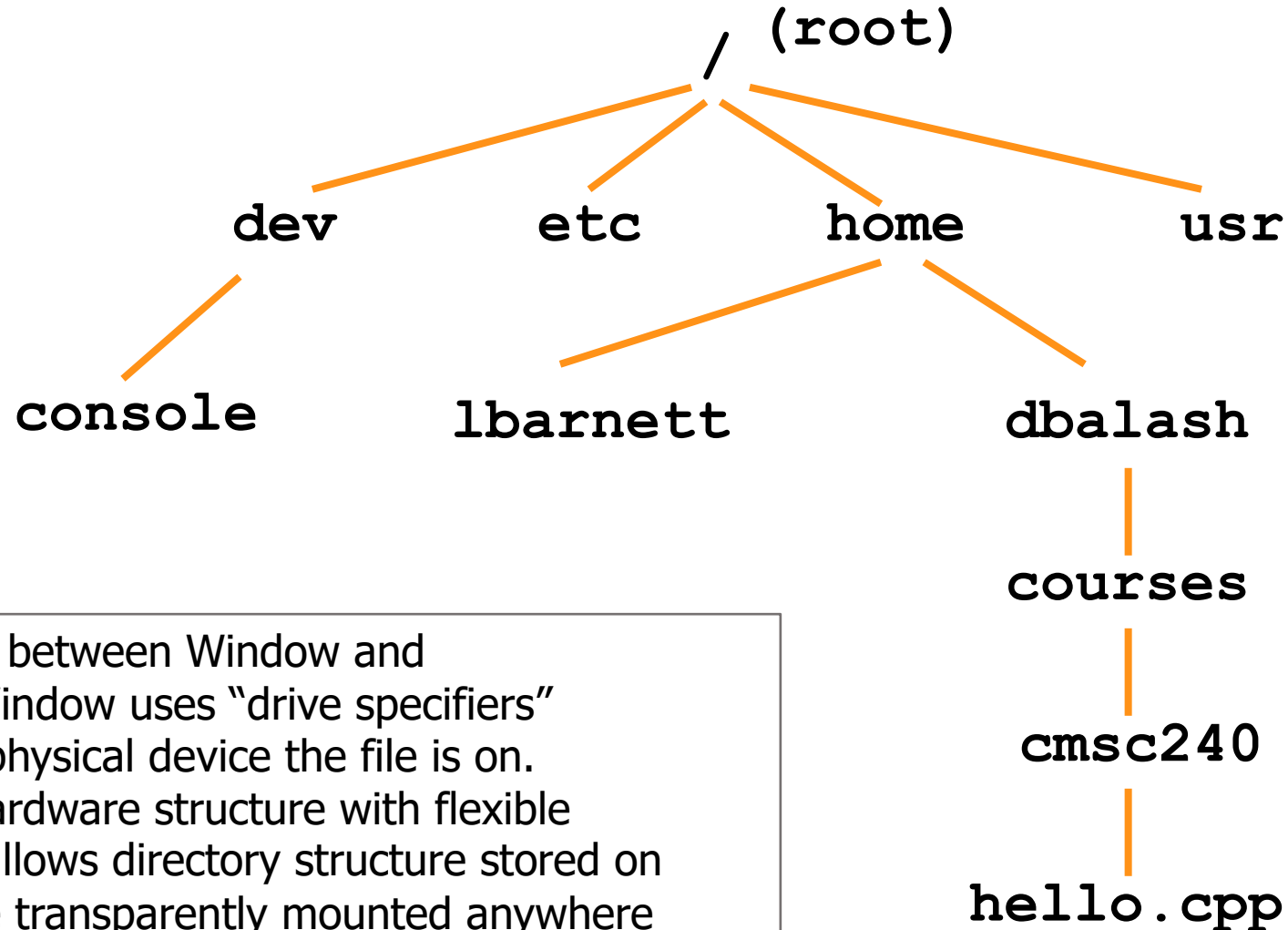
Intro to Unix



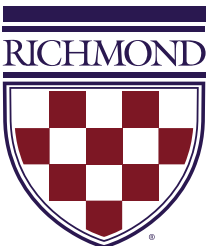
Example Unix File System (on Mac)



Example Unix File System (on Linux)



Big difference between Window and Unix/Linux: Window uses "drive specifiers" that indicate physical device the file is on. Linux hides hardware structure with flexible scheme that allows directory structure stored on a device to be transparently mounted anywhere in a tree-structured file system.



Unix/Linux File System

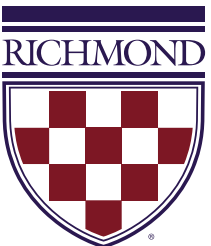
- Special directory names:
 - Root directory: /
 - Current directory: .
 - Parent directory: .. (allows you to go up)
 - User's home directory: ~
 - Some other user's home: ~sb4tc
- Two primary operations for navigating/locating:
 - `cd <name>` change directory to "name" (relative)
 - `ls` list all files/directories in current directory



Example Terminal Commands

```
$ cd ~  
$ mkdir cmsc240  
$ cd cmsc240  
$ pwd  
$ echo "Hi!" > myFile.txt  
$ cat myFile.txt  
$ cp myFile.txt yourFile.txt  
$ mv yourFile.txt ourFile.txt  
$ mkdir tmpDir  
$ mv ourFile.txt tmpDir  
$ ls  
$ cd ..
```

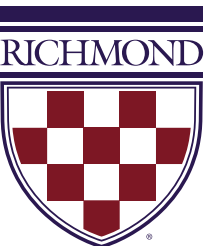
- change to home directory
- make a new cmsc240 directory
- cd to the cmsc240 directory
- print the present working directory
- redirect output to a new file
- display contents of file
- make a copy of the file
- rename the new file
- make another new directory
- move the file copy to new directory
- list current directory contents
- change to parent directory



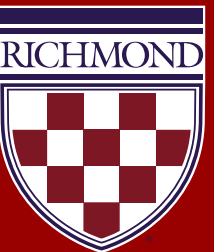
Need Help? Use “man” pages...

```
$ man ls  
$ man cd  
$ man man
```

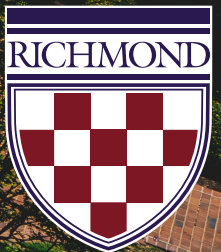
- Navigating a manual page:
 - <return> advances line at a time
 - <space> advances page at a time
 - b goes back one page at a time
 - /keyword searches for keyword
 - q quits



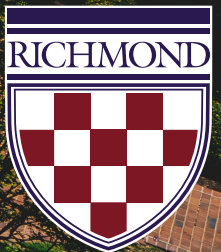
Give it a try!



Intro to Version Control with Git



Environment Setup



In-Class Coding Exercise

