

# C++ Basics

- Comparison C++ vs. Java
- Today: simple C++ constructs
  - programs
  - primitive types and strings
  - computation
  - control-structures
  - functions
  - parameter passing
  - fixed-size arrays
- All files in blue are in [~csci455/code/04-06](https://github.com/csci455/code/04-06)

# Announcements

- Lab this week on C++ vectors
- PA4 due Wed 4/12
- Change to office hours this week (on web page).

# High level comparison between C++ and Java

- C++ is an older language
  - standard library is much smaller
- C++ is a hybrid: OO / structured programming
  - more features
  - But also means...larger with more complex rules
- C++ is closer to the machine:
  - runs faster
  - more flexible
  - But also...
  - harder to diagnose bugs
  - harder to port code

# Detailed outline

- What's the same in C++/Java?

**SIMS**

- how to compile and run
- form of a program

**hello.cpp**

- simple I/O

**greet.cpp**

- named constants

- conditions in if/while

**pizzacalc.cpp**

- looking up library doc

- read to EOF

- error-check input

**readVals.cpp**  
**readVals2.cpp**

- declaration order

- funcs & parameter passing

**freq.cpp**

- fixed-size arrays

- a summary of differences covered today in

**DIFFS**

# Parameter passing

Can think of three kinds of parameters:

- IN
  - OUT
  - IN-OUT
- 
- use pass by value for the first
  - can use pass by reference for the second two
  - if just one OUT param, can use return value of function

# Pass by value

- Used for primitive types (same syntax as in Java)
- ALSO for objects:
  - objects get passed by value
  - objects get returned by value
  - Example:  
**string reverseString(string s) ;**
  - string is copied on function call
  - return value is also a copy  
(not a reference to an object from the function)

# Call by reference

- Call by reference for IN-OUT mode

```
void swap(int &a, int &b) {  
    int temp = a;  
    a = b;  
    b = temp;  
}  
  
int main() {  
    int x = 10;  
    int y = 20;  
    swap(x, y);  
    cout << x << " " << y << endl;  
    return 0;  
}
```

# Passing arrays as parameters

- Array is not copied, just like in Java:

```
void foo(int myarr[], int size) {  
    for (int i = 0; i < size; i++) {  
        myarr[i] = 50;  
    }  
}
```

```
int main() {  
    int arr[20];  
    foo(arr, 20);  
    cout << arr[0] << endl;  
    return 0;  
}
```