# C++ Coding Standards for Computer Science 2308

## Class Names

* Class names are nouns : Dog, Cat, LinkedList ***not*** RunProgram
* Class names are singular: Amphibian ***not*** Amphibians
* Class names observe “camel back” notation, i.e., each word in the name is capitalized, e.g., IstanbulNotConstantinople ***not*** istanbulnotconstantinople
* Class names do not have underscores “\_” , e.g., IstanbulNotConstantinople not Istanbul\_Not\_Constantiople

## Variable Names

* Variable names adhere to the same conventions as class names except they are not capitalized, e.g., aFilm ***not*** AFilm

## Method Names

* Methods names adhere to the same conventions as variables names except they are verbs not nouns, e.g., getName ***not*** theNameGetter;

## Constants

* Constants are declared as all caps with underscores (“\_”) between words, e.g., PRICE\_OF\_BEANS ***not*** PRICEOFBEANS.
* Constants are preferred over using a raw values, see below

**const int** NUMBER\_OF\_VALUES = 20;

int fibonacciValues [NUMBER\_OF\_VALUES];

rather than

int fibonacciValues [20];

## Primitive Data Types

* For the short term, we will confine our use of primitive data types to the following:
  + Int
  + float
  + double
  + char (occasionally)
  + string (not a primitive, but often treated like one)

## Naming

* Use mnemonic names for classes, variables, methods; longer names that make clear what your class, variable, or method represents are preferred over shorter names that are less clear. See below for an example

int wednesday = 4;

rather than

int w = 4;

## Comments

* Comments your code!

## Code Formatting

* Use whitespace to make code more easily readable.
* Align brackets to illustrate scope

for (int i=0; i<10; i++ ) {

// within the scope of the for loop

} // end of the for loop scope

* Use brackets to delineate scope even when only one statement is within scope

if (primeCandidate < = 0) {

return false;

}

rather than

if (primeCandidate < = 0) return false;