# CMSC131

Some guidelines for naming identifiers (variables, constants, etc) and formatting your code.

## NOTE

Please note that this slide deck is an at-home required reading.  The style conventions are important, but this material doesn't make sense as an in-class lecture when you can simply read these and review them while you are writing programs for this and other classes.

## Naming Identifiers (I)

There are some Java-enforced rules like…

- The first character must be a letter, the underscore or a dollar sign.
- Names are CaSeSensitivE
- There are reserved words that you cannot use (such as **if**, **else**, **class**, *etc.*)
- Names cannot contain certain special characters (such as **-**, **%**, **+**, **&**, **!**, *etc.*)

## Naming Identifiers (II)

There are naming ***conventions*** for you to follow:

- Don't start a name with a $ (this is usually an indication of a system-level identifier in Java).
- Don't differ identifiers *JUST* by CasE. `int Value, VaLue;`
- Use meaningful names (eg: the name should express what is being held in the variable).
- Use English words (at least when coding here).
- This isn't Twitter ☺ so please don't go too crazy dropping vowels and consonants.
- Use "Camel Case" name formatting.

## "Camel Case"

Variable names and method names start with **lower case letters**.  Class and interface names start with **upper case letters**.

Each word other than the first word starts with an upper case letter.

```
int  myFirstInteger;
public  class  MyFirstClass {
```

Constants are ALL UPPERCASE with underscores between words.

```
final  int  DAYS_IN_A_WEEK = 7;
```

## Speaking of CONSTANTS…

If there is a value that we will be using that will never have to change while our program is running, it should really be stored in a meaningful **constant** (also known in Java as a **final** variable).

- Even if you only plan to use it ONE TIME we want you to do this.
- Even if it is a simple "flag" condition (type 1 for cat, type 2 for dog, …) we want named constants like

```
final   int   CAT_CHOICE  =  1;
…
if (userInput == CAT_CHOICE) { …
```

## Other formatting conventions…

Use placement of braces { } as you have seen in the posted full examples.

Use indenting similar to what you have seen in my posted examples.

- If in doubt, Eclipse can help you if you highlight your code and then press `Control+I` (it will autofix your indentation).

Don't have any lines that are longer than 80 characters long (that includes spacing).

## Valid versus Good Style

1. **For** – valid but not good style
2. **success%** - invalid due to %
3. **x** – valid but might not be good style depending on context
4. **i** – valid but might not be good style depending on context
5. **o** – valid but likely not good style
6. **starting_val** – valid and likely good style