Java Basics cheatsheet

Key Terms

- Most Java programming involves calling **methods** of **objects**. The syntax for this is obj.method(param1, param2).
- Objects are created when we **instantiate classes**. We do so with the *new* keyword: new Cat("Ziggy"). This calls the class **constructor**.
- Classes can **extend** other classes, which means that they inherit all the functionality from those other classes (but they can also overwrite some of it).
- We also have **interfaces** which are a set of method signatures. A class can **implement** an interface if it has implementations for all the methods indicated in the interface.
- The this keyword is used in an object method to refer to the object itself, and to provide access to its **fields**.

Visibility

Classes, Variables, and Methods etc have *visibility* indicated by a keyword:

public Can be accessed by anyone.

private Can only be accessed from objects of the class that contains them.

package-private Can be accessed by any classes that are within the same package. This is also the default, if no specific word is used.

protected Can be accessed by subclasses of the class.

Variables

Methods and objects work with a number of different "variable" symbols. They vary in their **scope**, i.e. the specification of all the parts of the code where they exist.

Instance variables also called **fields**, are unique to each object, typically created when the object is instantiated. Their values are shared amongst all the methods of the object.

Static variables or **static fields** are properties associated with a class and are shared amongst all object instances of that class. Similarly static methods can be called just using the class name and without requiring a class instance.

Parameters or **arguments** are passed to the method from its caller. Their value only extends to the end of the specific function call.

Local variables or simply **variables** are defined within a function and exist only within the innermost set of curly braces that contains their declaration.

Type of a variable

- Each variable has a *type* specified at the moment of its declaration.
- This can be a built-in datatype like int or char, or it can be a class or interface.
- When assigning a value to a variable, the type of

Syntax elements:

- function definitions
-