**Primitive** types are not objects

**Reference** type, point to a new keyword(), instantiate, allocate memory to hold that specific object

**Local variable:**

declare within a method

Only exist during the method call

**Instance variables:**

declare on top of the class (create every time an object is created)

have longer lifespan than local variables , lass long as an object does

**Capital O object:**

ToString, inheritance

Use extends to make it a subclass of parents class: can access public and protected methods of the parents class not private, polymorphism

**Interface**: not implementation

**Overloading**: add a method of the same name as the constructor of parents class (call super.constructor for parents)

**Super**: reference to the constructor of parents class

**Polymorphism**:

sub class like the super class? make a Car object and assign that to RobberCar, every RobberCar is Car but not vice versa, Car can be CobCar and RobberCar are none of them

An array of Car (mix of RobberCar and CobCar or none of them) if Car, RobberCar and Cobcar all have update method, dynamically at runtime Java will use the update method of the most specific type, the lowest of the inheritance tree, the sub type.

**Instanceof** : is an operator check dynamically at runtime look the object and the class RobberCar instanceof RobberCar == true, instanceof Car ==true

**Shadowing**: data type of same name but different scopes. The closet variable get used, if want to use instance variables use this.variable

**Interface vs adapter:**

Interface implement method that doesn’t need body (open and close brace with empty body)

Adapter override the needed method

**Abstract class:**

can’t create new object, instantiate = new class();

**final**:

keyword: make the variable unchangeable and can be declared only one time, locked in valueS (like pi)

**static**:

instance variables is a class variable and are created every time a new object is created (private)

but static is shared among all objects of that class (only 1 copy)

class.variable => to get the static variables

**generic:**

angle brackets to specify to (T template?)

if method takes an int and pass as a short, long…