Introduction to Java

Imran A. Zualkernan

COE312

Java

- Java is very similar to C++
- It is an object-oriented language
- Relies heavily on external libraries

Java Tutorial

- We will now follow the tutorial here to explore various basic aspects of Java.

Tutorial:

https://www.w3schools.com/JAVA/default.asp

Compiler:

https://www.onlinegdb.com/online_java_compiler

For the course we will use the Eclipse compiler available for free download at.

https://www.eclipse.org/downloads/packages/

Cover the following topics in the tutorial

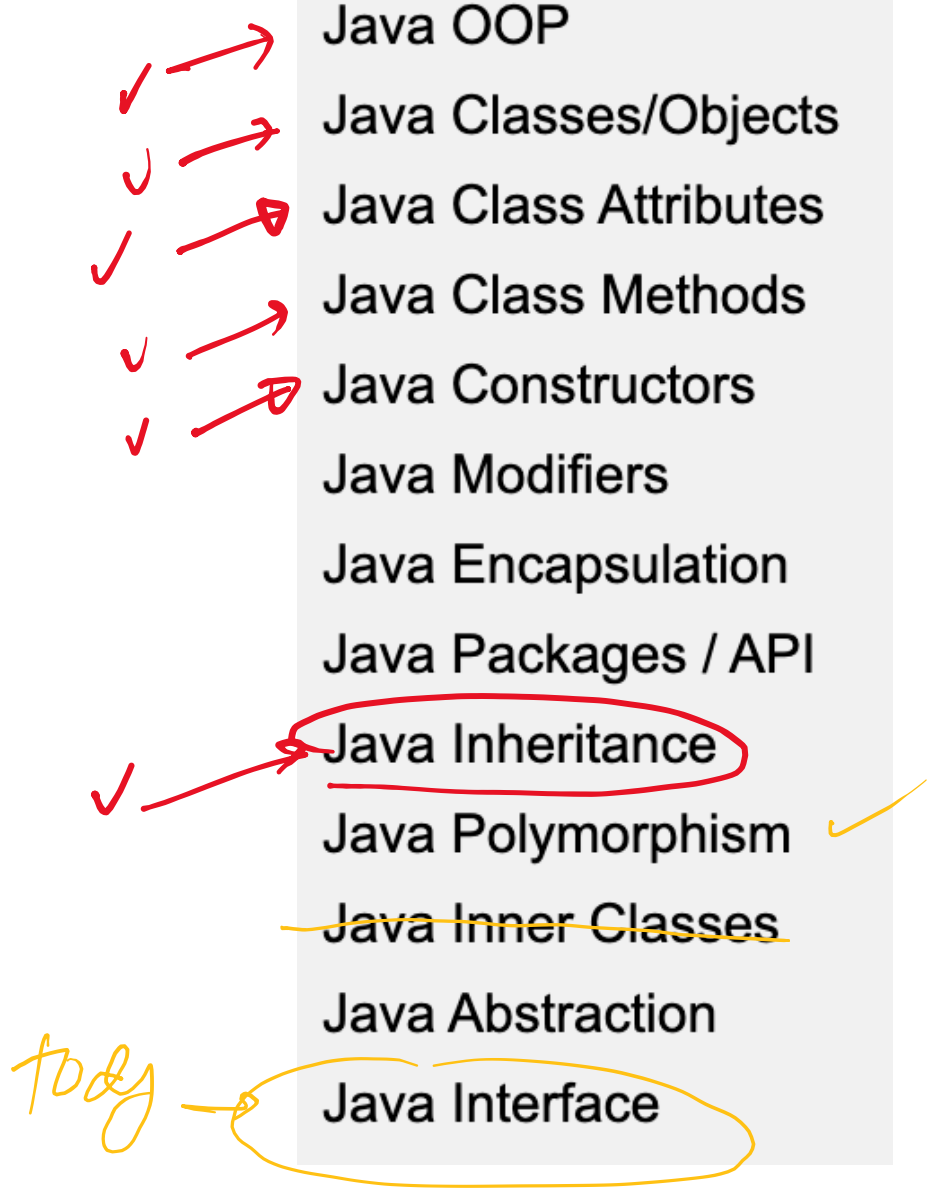
Java Tutorial

- Java HOME
- Java Intro
- Java Get Started
- Java Syntax
- Java Comments
- Java Variables
- Java Data Types
- Java Type Casting
- Java Operators
- Java Strings
- Java Math
- Java Booleans
- Java If...Else
- Java Switch
- Java While Loop
- Java For Loop
- Java Break/Continue
- Java Arrays

Cover the following in OO Programming in Java

Java Classes

- Java OOP
- Java Classes/Objects
- Java Class Attributes
- Java Class Methods
- Java Constructors
- Java Modifiers
- Java Encapsulation
- Java Packages / API
- Java Inheritance
- Java Polymorphism
- ~~Java Inner Classes~~
- Java Abstraction
- Java Interface



Review

- Unlike C++ almost everything in Java is a pointer. The only exception are primitives like int, float etc.
- Java has inheritance but does not support multiple inheritance.
- Passing values to functions is by value (or copying) only. No pass by reference is supported.
- Similar to namespaces, Java has the concept of a Package that is a collection of classes.
- There is no **delete** operator. A program called garbage collector cleans up memory automatically.
- Default access levels in Java are different than C++.
- Constructors behave slightly differently than C++.

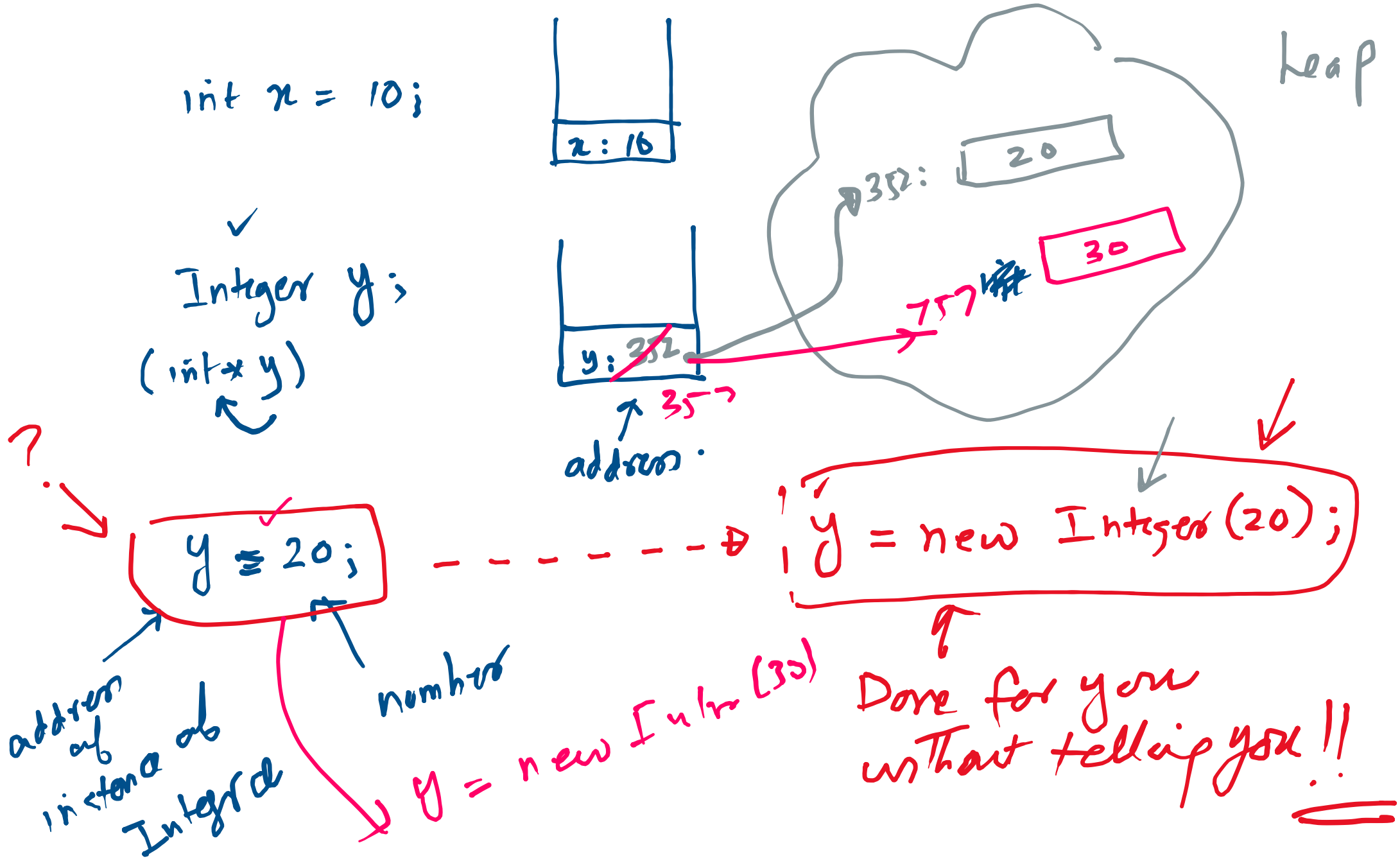
Review

- Each field and method in c class has an *access level*:
  - private: accessible only in this class
  - (package): accessible only in this package
  - protected: accessible only in this package and in all subclasses of this class
  - public: accessible everywhere this class is available
- Similarly, each class has one of two possible access levels:
  - (package): class objects can only be declared and manipulated by code in this package
  - public: class objects can be declared and manipulated by code in any package

Note: for both fields and classes, package access is the default, and is used when *no* access is specified.

Review

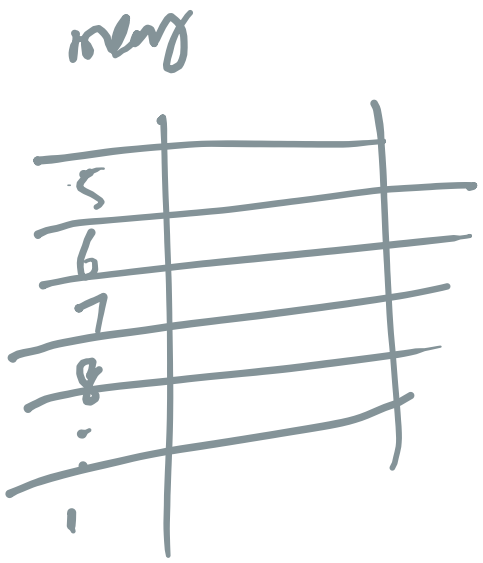
- If a constructor has arguments, you supply corresponding values when using new.
- Even if it has no arguments, you still need the parentheses (unlike C++). For example, Flower f = new Flower; is not legal in java.
- Just like C++ there can be multiple constructors, with different numbers or types of arguments. This is called constructor *overloading*.
- **Default constructor is created only if there are no constructors**. If you define *any* constructor for your class, no default constructor is automatically created.
- **this(…)** - Calls another constructor in same class.
- **super(…)**. Use *super* to call a constructor in a parent class.
- The Java compiler inserts a call to the parent constructor (super) if you don't have a constructor call as the first statement of you constructor.
- Unlike C++, you cannot overload operators.

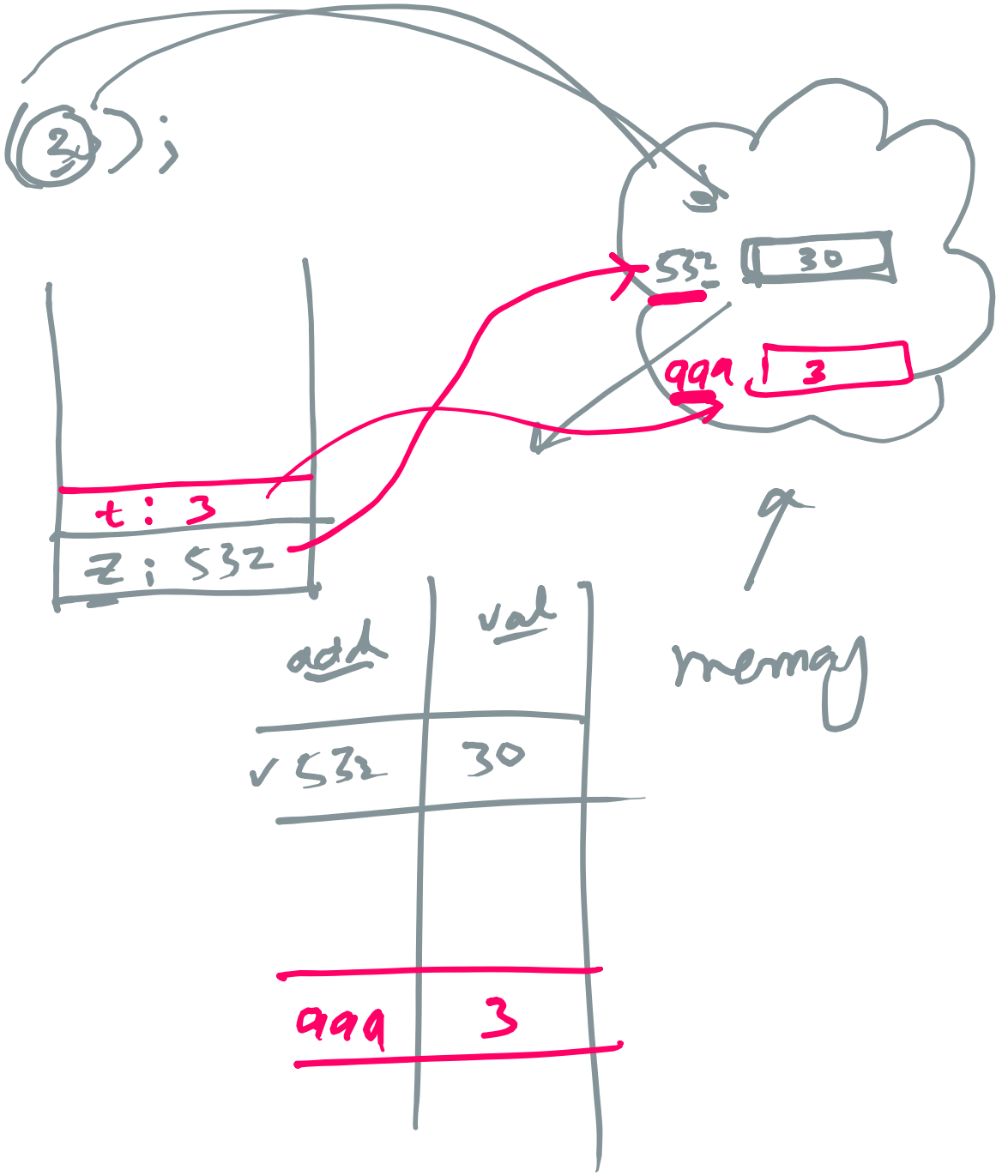

int x = 10;
x : 10
Integer y;
(int* y)
y : 352
352 :
20
30
Heap
addr.
y = 20;
y = new Integer(20);
address of instance of Integer
number
y = new Integer(30)
Done for you without telling you !!

Intger t = new Integer(3);

Integer z = new Integer (3);

allocate memory for an instance of the class called Integer





Cat is a class

C++

Cat C;

Cat c is an object instance of class

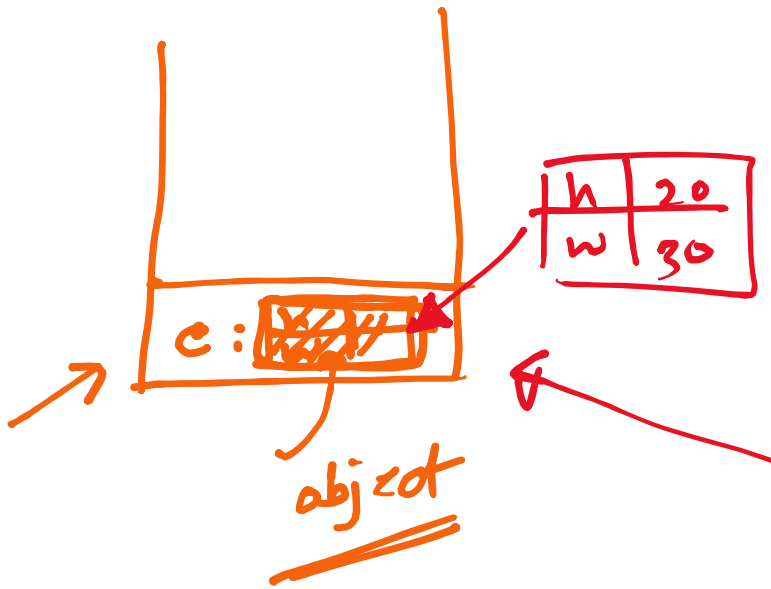
Java

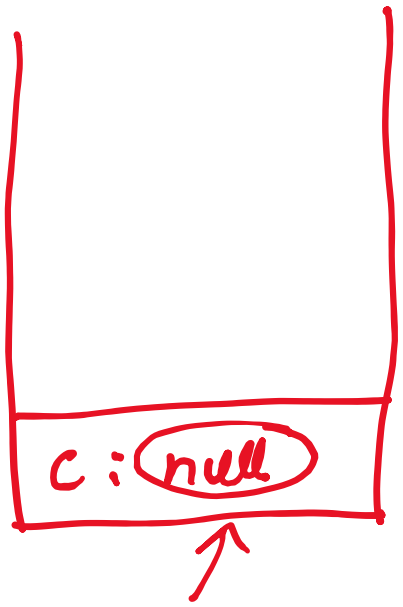
Cat C = null;

~~Pointers~~

1. C is a variable that hold the address of an instance of the class Cat

```
class C {
int height;
int weight;
}
```





null? = no address has been assigned!

how you Print

→ hex?

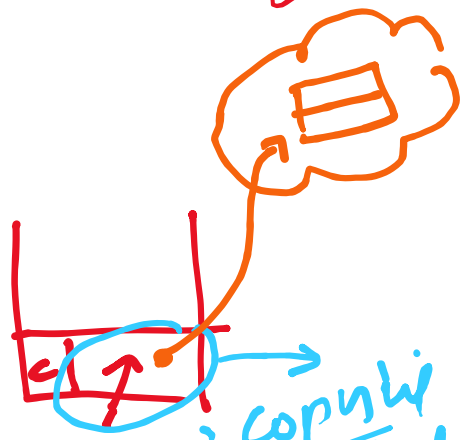
base 16

Creating an instance of a cat

C++

bad ✗ ?

```
Cat c;
```



right way

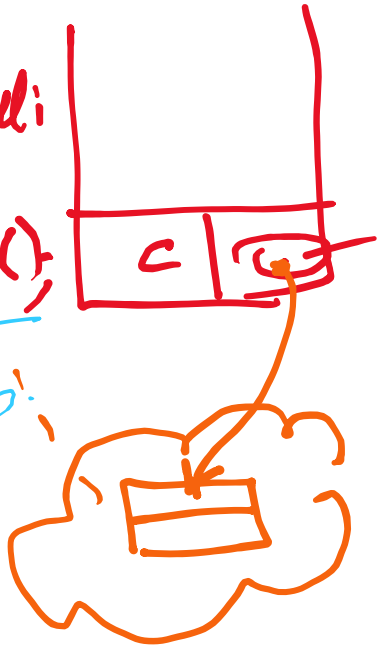
```
Cat* c = new Cat();
```

✓ ✓ ✓

- `Cat` — object
- `*` — address
- `c` — address
- `new Cat()` — address

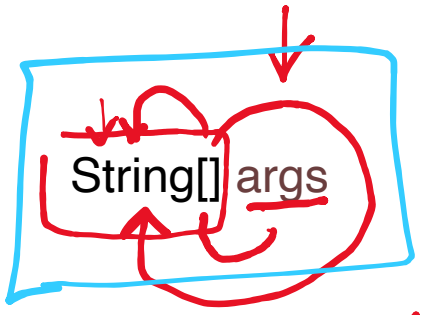
Java

```
Cat c = null;
c = new Cat();
```



✓

- `c` — addrs var
- `new Cat()` — addrs

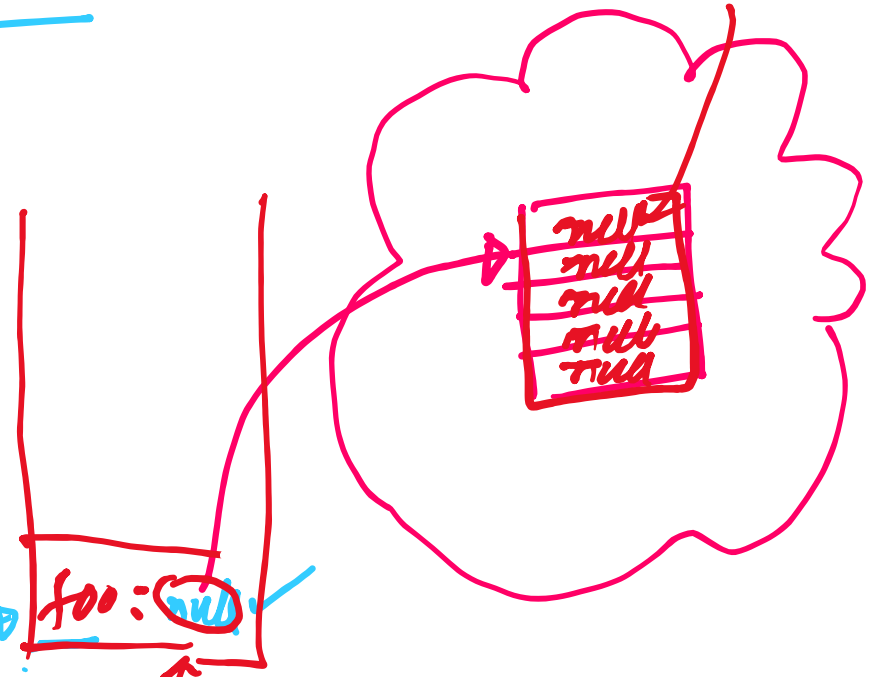


String [] foo = null

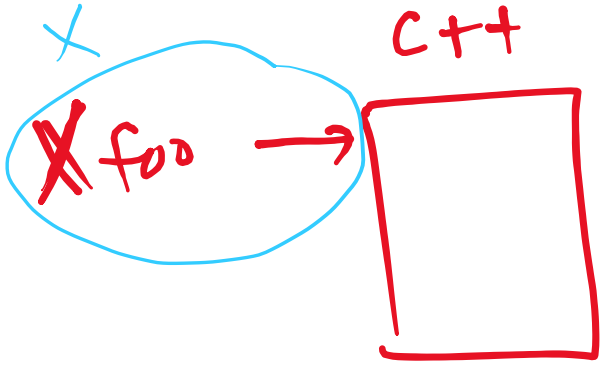
10 addr

args is an array of addresses of strings

the address of String



C++



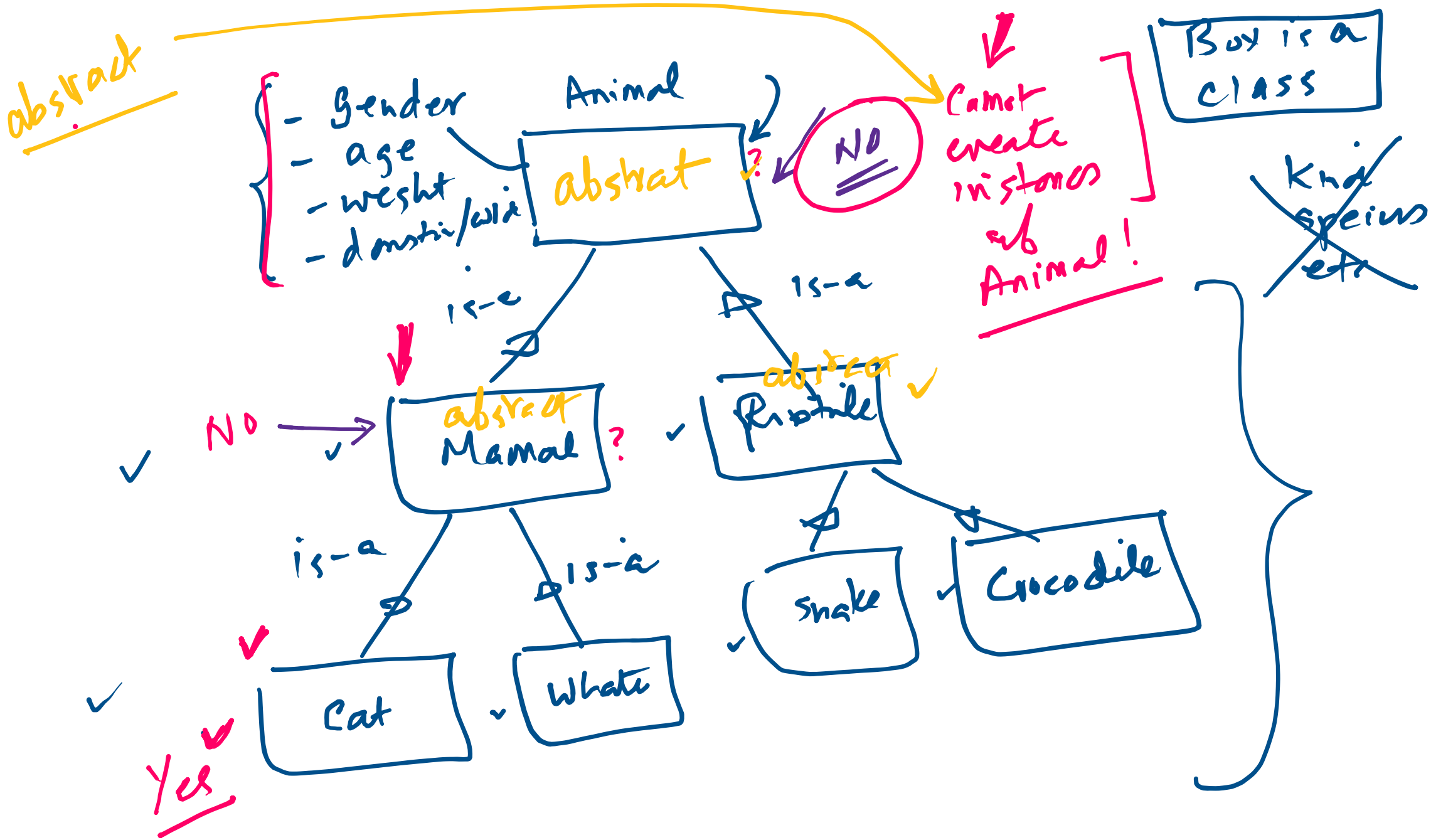
Java

stop talking about name of arg as the address of the first element in the array

foo = new String[10]; // all addresses

?

address of


abstract
Animal
- Gender
- age
- weight
- domestic/wild
abstract
?
NO
Cannot create instances of Animal!
Boy is a class
Dog
Lion
etc
is-a
is-a
NO
abstract Mamal
?
abstract Reptile
Cat
Whale
Snake
Crocodile
Yes

Constructor:

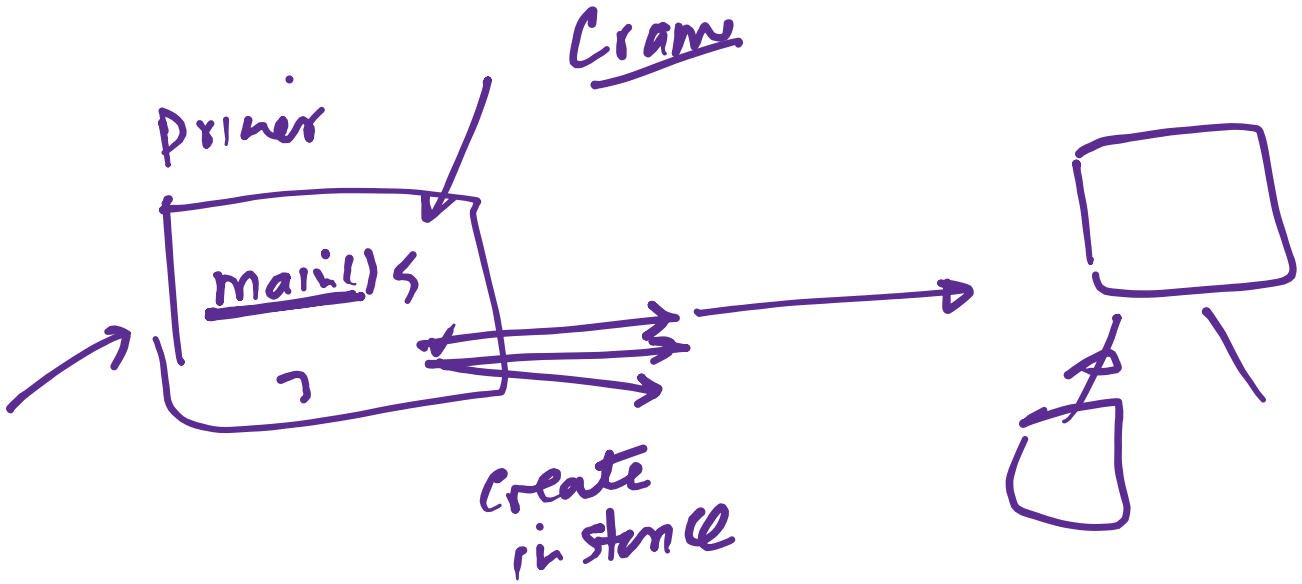
↳ [function which is typically called when you create a new instance of a class.]

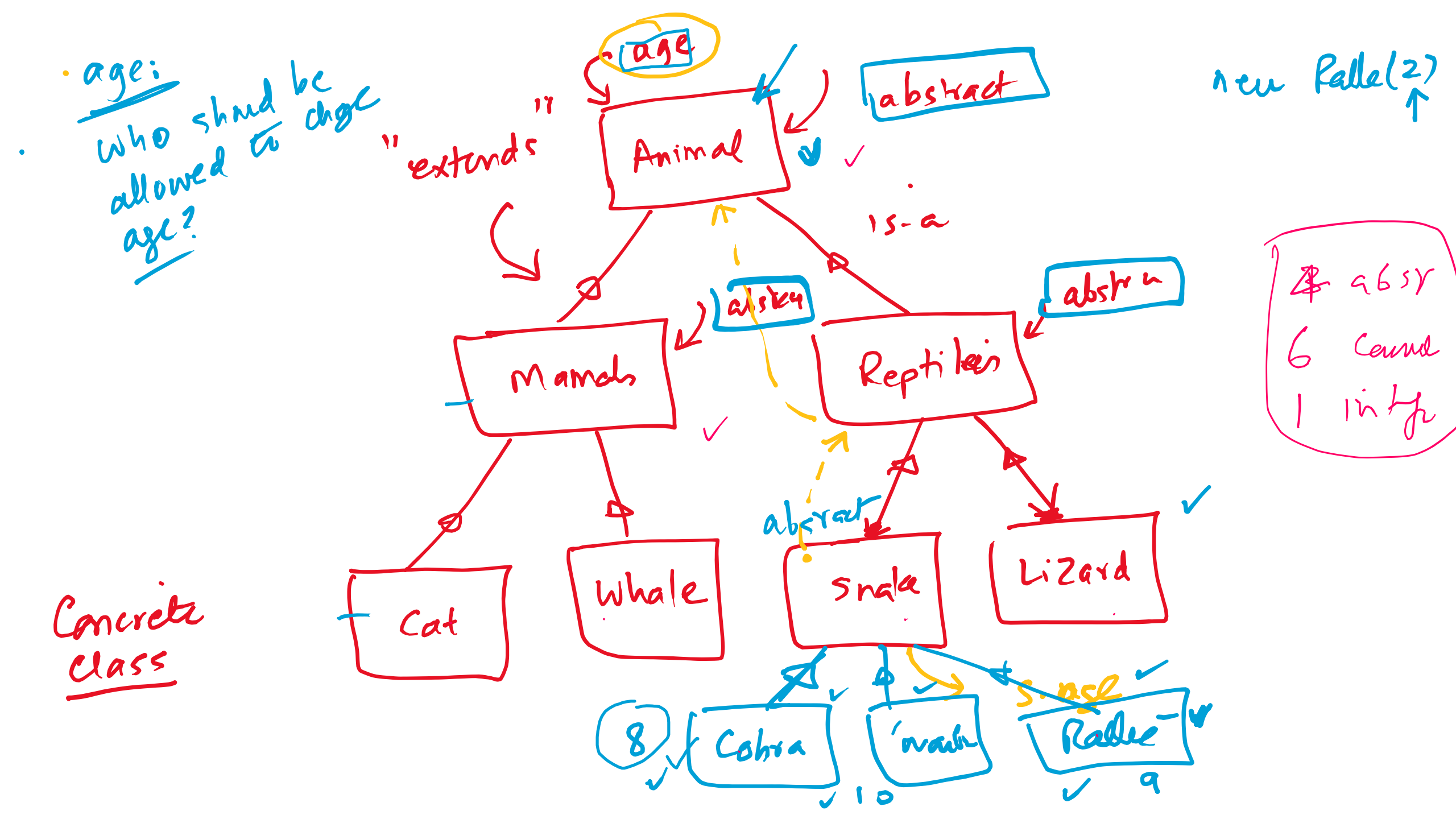
→ Animal(->-)

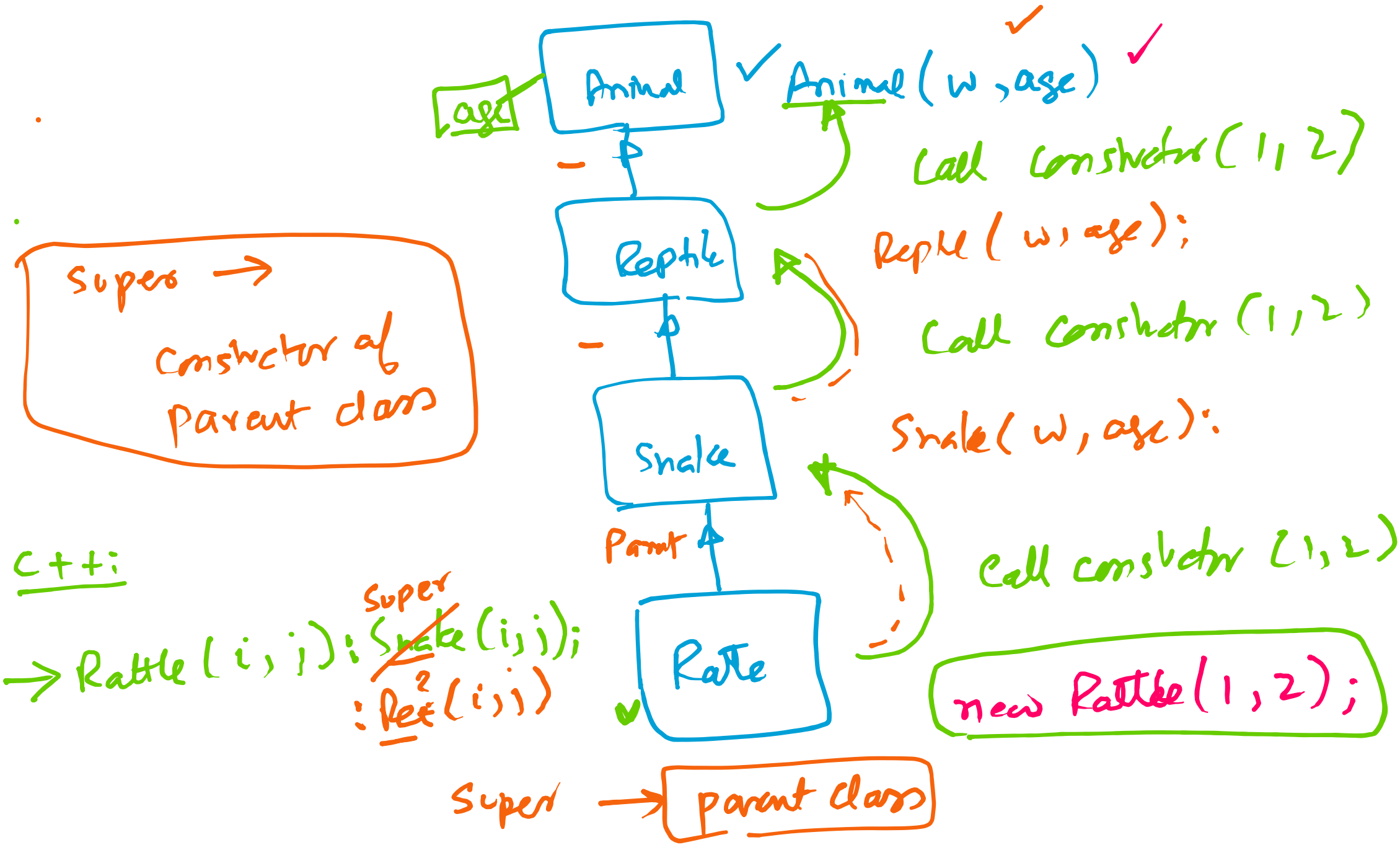
new Animal(->-)

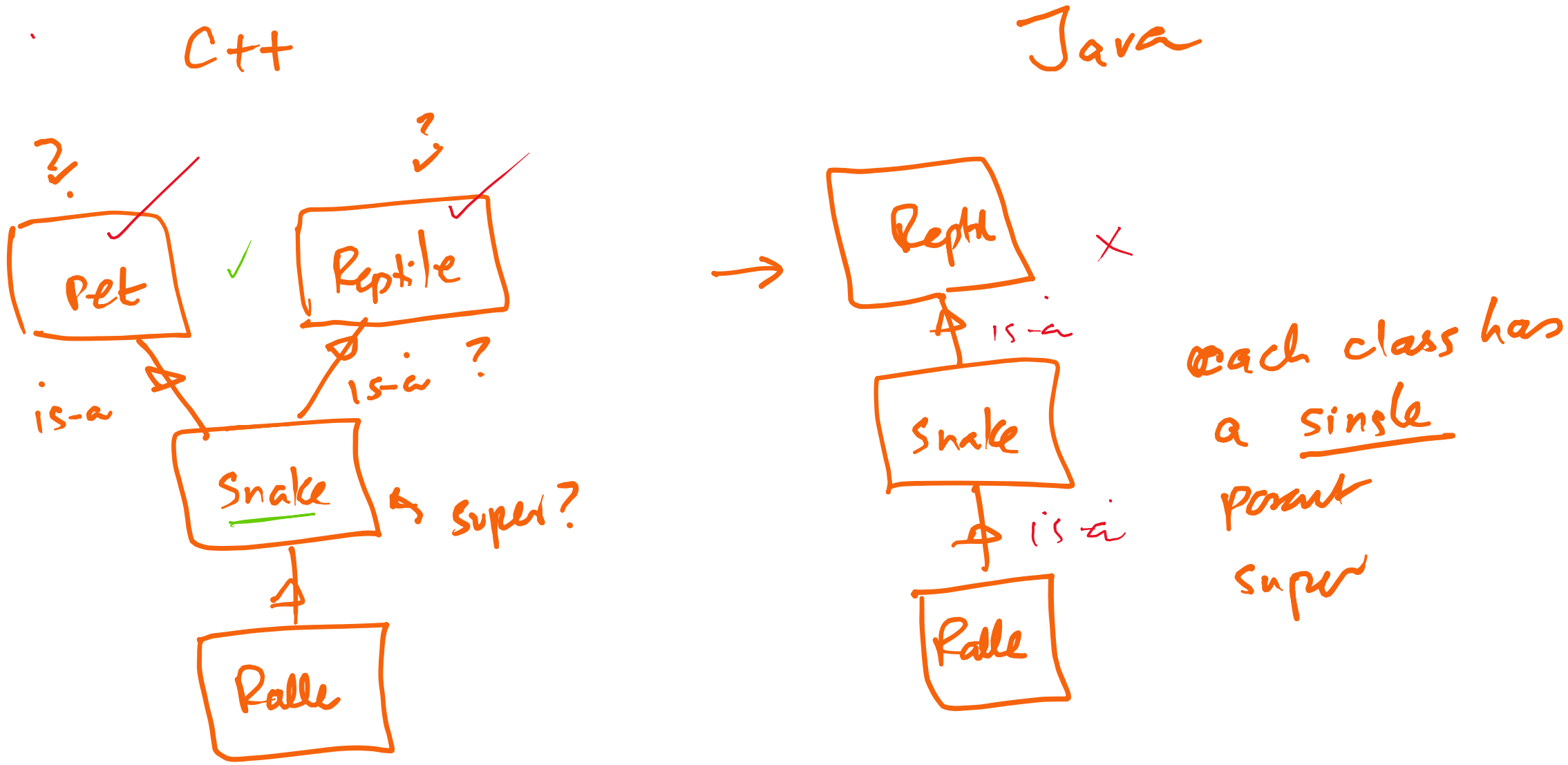
↓

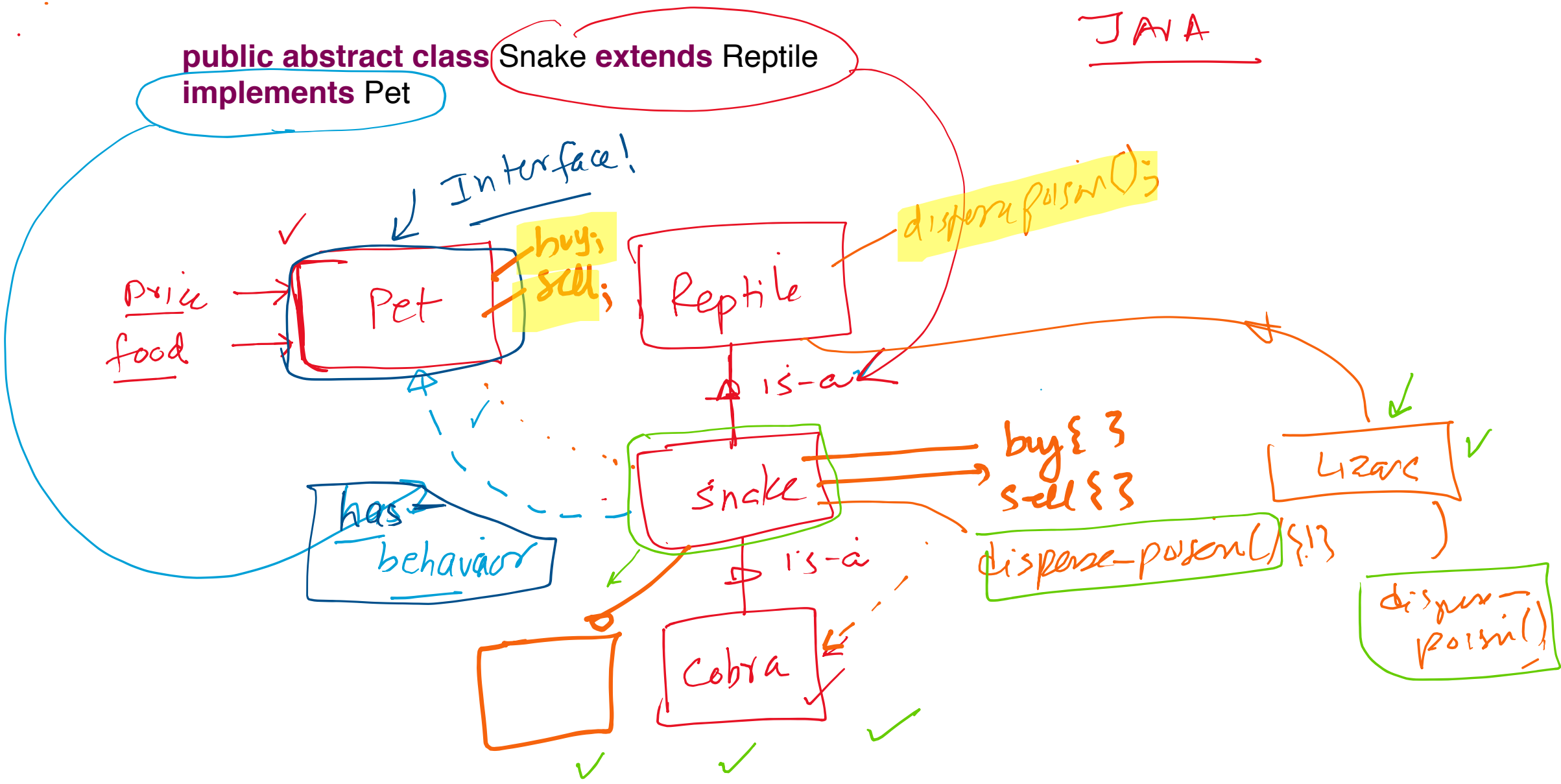
this ⇒ "me"
or "my address"


Primer
Cramr
main() {
}
Create instance

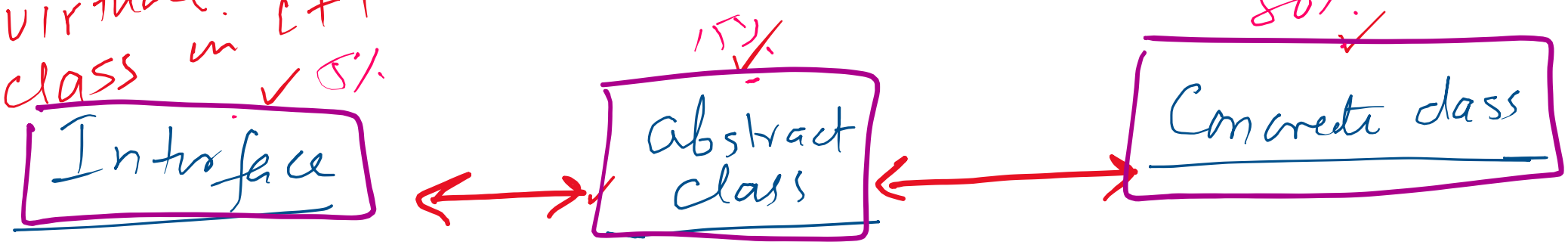

age
abstract
Animal
"extends"
is-a
Mamals
Reptiles
Cat
Whale
Snake
Lizard
Cobra
Rattle
Concrete class


Animal
age
Animal(w, age)
Reptile
Snake
Rattle
Parent
super → constructor of parent class
Call constructor (1, 2)
Reptile(w, age);
Call constructor (1, 2)
Snake(w, age);
Call constructor (1, 2)
new Rattle(1, 2);
C++:
→ Rattle(i, j); : Snake(i, j); : Reptile(i, j)
super
Super → parent class


C++
Pet
Reptile
Snake
Ralle
is-a
is-a
?
super?
Java
Repti
is-a
Snake
is-a
Ralle
each class has a single parent super


public abstract class Snake extends Reptile
implements Pet
JAVA
Interface!
Pet
buy;
sell;
price
food
Reptile
dispensePoison();
is-a
Snake
buy{ }
sell{ }
dispense_poison(){ }
has
behavior
Lizard
dispense_
poison()
is-a
Cobra

Pure virtual class in C++



- Cannot create instances
- does not have implementation of functions

[all functions are abstract]

Animal

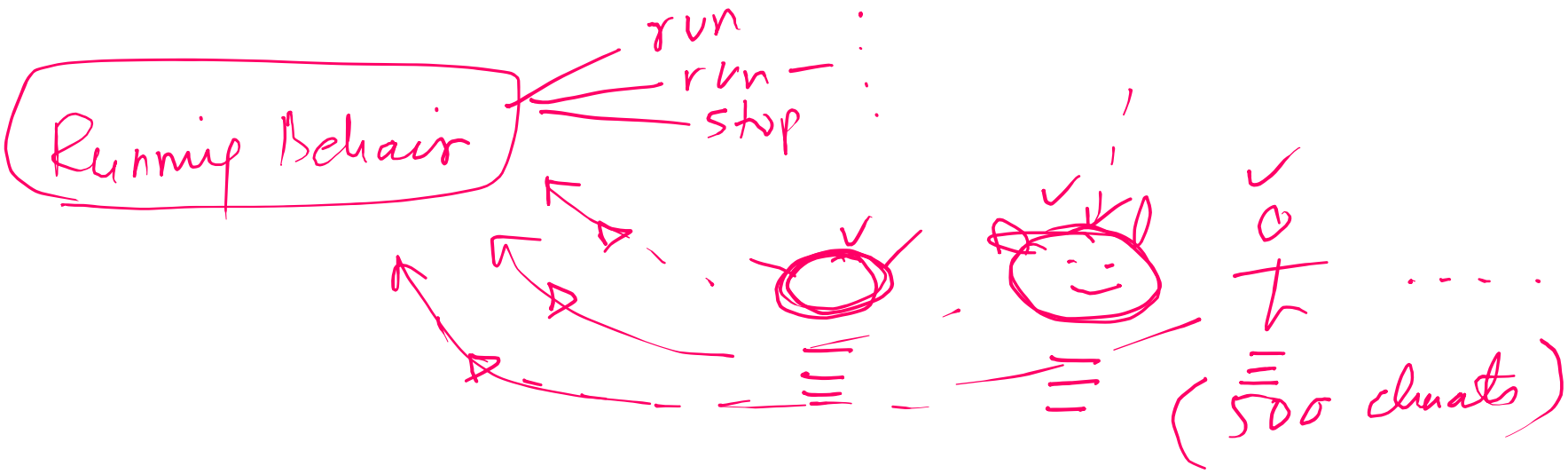
- Cannot create instances
- may have some functions [abstract fn.]
- Can you leave some fncts Undefd

Cobra

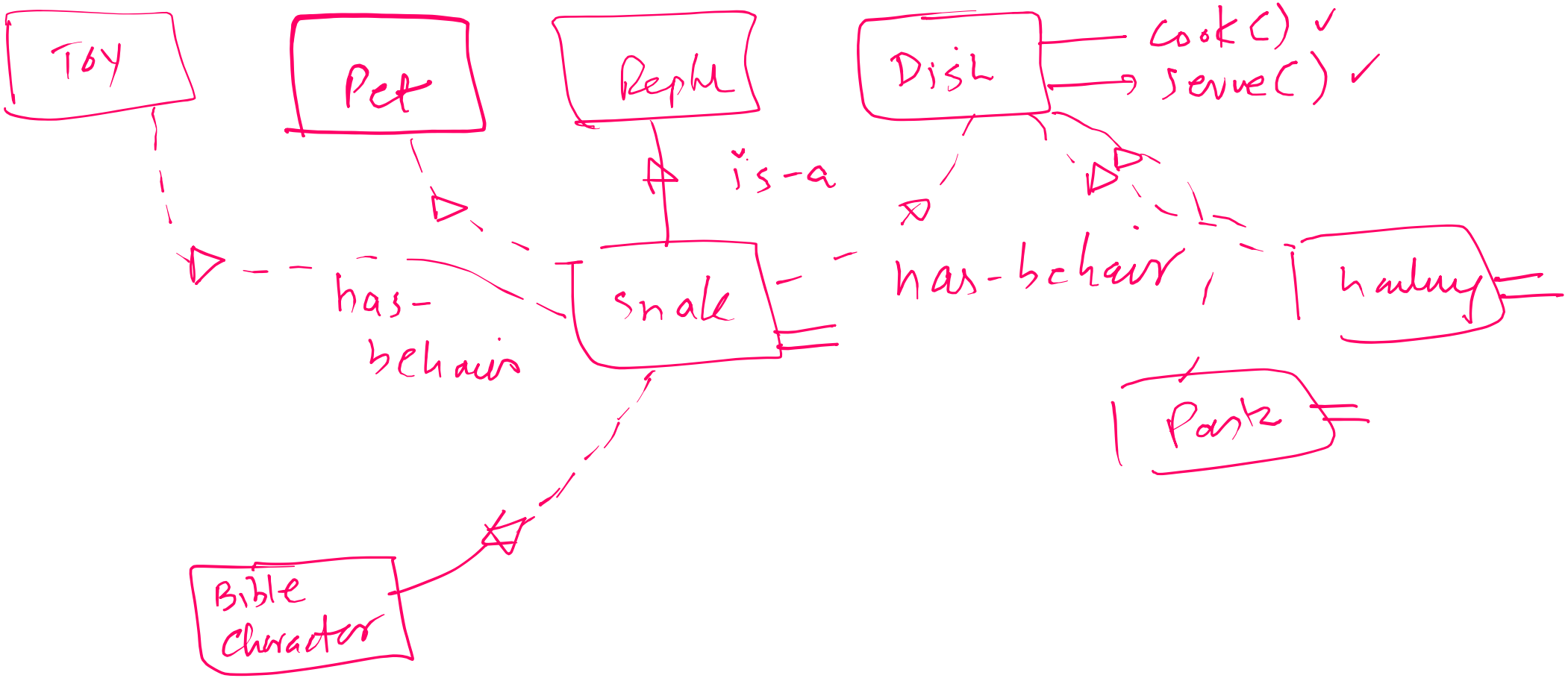
Cobra c1 = new Cobra();

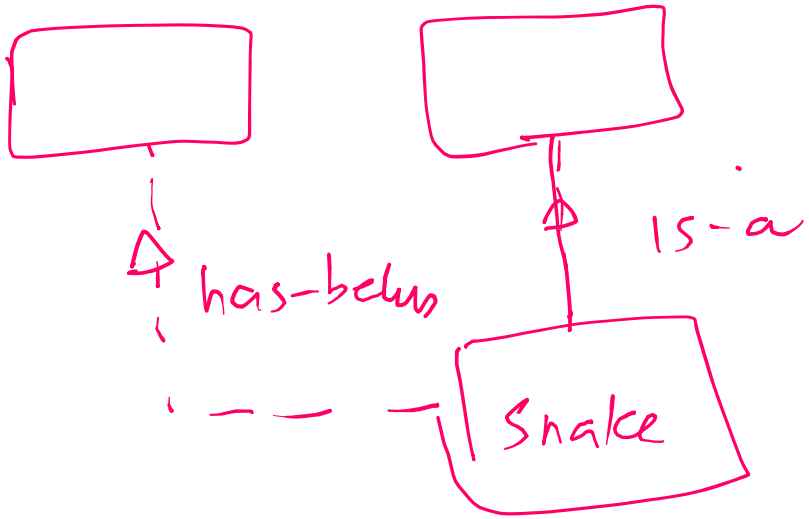
Interface is a contract

→ which function
→ which property



how many? 256

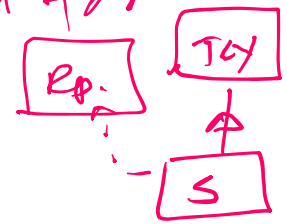

Toy
Pet
Reptile
Dish
cook() ✓
serve() ✓
is-a
has-behavior
Snake
has-behavior
Parts
Bible character



1. Reptile
2. Toy ??

Depends on your application

A. Buildy a system for Toy store



B. Buildy a system for Zoo