Beautiful is better than ugly.
Explicit is better than implicit. Simple is better than complex. Complex is better than complicated. Flat is better than nested. Sparse is better than dense.
Readability counts. Special cases aren't special enough to

Although **practicality** beats purity. Errors should never pass silently. Unless **explicitly** silenced. In the face of ambiguity, **refuse** the temptation to guess. There should be **one**—and preferably only one—obvious way to do it. Although that way may not be obvious at first unless you're Dutch. **Now** is better than never. Although never is **often** better than right now. If the implementation is hard to explain, it's a **bad** ea. If the implementation

is easy to explain, it may be a good idea. Namespaces are one honking great idea — let's do more of those!

### IS612 Introduction To Coding Spring 2022

idea, It the implementation of the implementation is easy to explain, it is easy to explain, it is easy to explain a great and in the interest of each in the each interest of each in

Although practicality beats purity. Errors should never pass silently. Unless explicitly silenced. In the face of ambiguity, refuse the temptation to guess. There should be one — and preferably only one — obvious way to do it. Although that way may not be obvious at first unless you're Dutch. Now is better than never. Although never is often better than right nown. If the implementation is hard to explain, it's a bad

break the rules. Mithough **preak** the rules. Although **practicality** beats purity. Errors should never

ot riguona leisage

Explicit is better than ugly.

Explicit is better than implicit. Simple is better than complex. Complex is better than than complicated. Flat is better than nested. Sparse is better than dense.

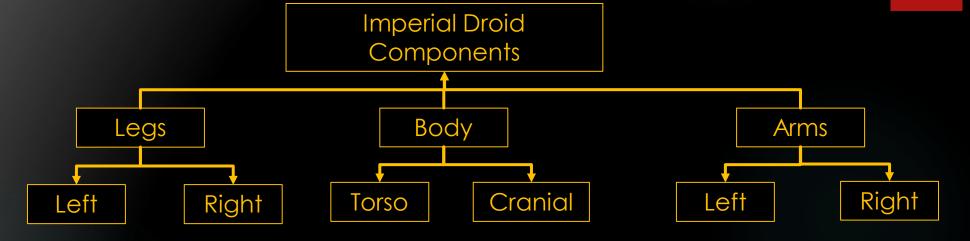
Readability counts. Special cases aren't

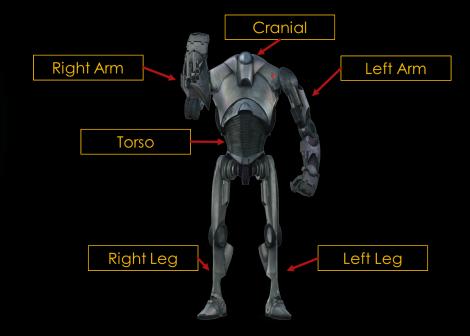
# **Python Programming**

OOP INHERITANCE EXERCISE



# Imperial Droid





### Imperial Droid Class

### **ImperialDroid**

- -DroidID: String
- -DroidType: String
- -Head: Cranial
- -UpperTorso: Torso
- -Arms: Arm ()
- -Legs: Leg()
- \_\_init\_\_(self, DroidID, DroidType)
- + displayInfo(): void
- + runDiagnostic(): void

- The ImperialDroid constructor instantiates all components of the droid when the ImperialDroid object is instantiated.
- The ID Numbers for all components at instantiation is 'X'

### SentryDroid\_A

- -DroidType = "Alpha"
- \_\_init\_\_(self, DroidID)
- + displayInfo(): void

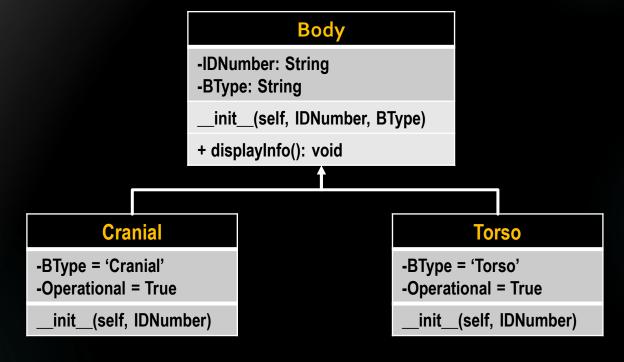
### SentryDroid\_B

- -DroidType = "Beta"
- \_\_init\_\_(self, DroidID)
- + displayInfo(): void

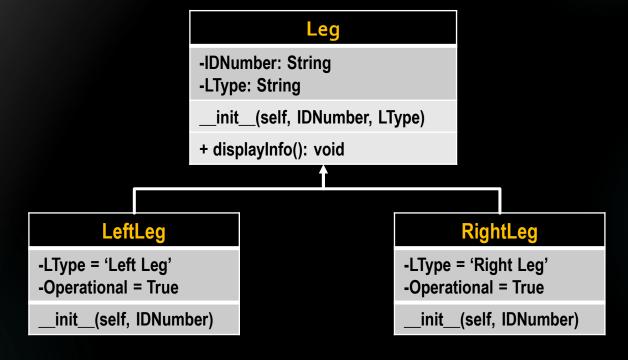
### SentryDroid\_C

- -DroidType = "Gamma"
- \_\_init\_\_(self, DroidID)
- + displayInfo(): void

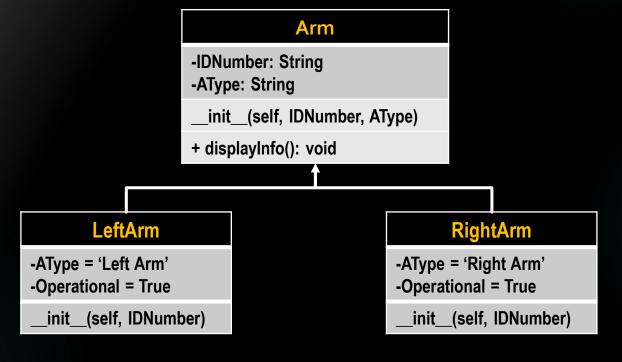
# Body Class



# Leg Class



### Arm Class



## **DroidFactory Class**

#### DroidFactory

-IDNumber: String
-BuildType: String

init (self, IDNumber, BuildType)

+ displayInfo(): void

#### Sentry\_AFactory

-BuildType = 'Alpha Droids'

\_\_init\_\_(self, IDNumber)

- + buildDroids\_A(int count): SentryDroid\_A []
- + static displayDroids(SentryDroid\_A[] dl): void

### **Sentry\_BFactory**

-BuildType = 'Beta Droids'

\_init\_\_(self, IDNumber)

- + buildDroids\_B(int count): SentryDroid\_B []
- + static displayDroids(SentryDroid\_B[ ] dl): void

#### Sentry\_CFactory

-BuildType = 'Gamma Droids'

\_\_init\_\_(self, IDNumber)

- + buildDroids\_C(int count): SentryDroid\_C []
- + static displayDroids(SentryDroid\_C[ ] dl): void

• When any type droid factory creates a droid, its DroidID is set to 'X'

# Class Methods

Method Specifications		
Method	Specification	Class
buildDroids_A(int count)	Creates the specified number of SentryDroid_As	Sentry_AFactory
buildDroids_B(int count)	Creates the specified number of SentryDroid_Bs	Sentry_BFactory
buildDroids_C(int count)	Creates the specified number of SentryDroid_Cs	Sentry_CFactory
static displayDroids(SentryDroid_A[] dl)	Calls the displayInfo() method of all droids in the list dl	Sentry_AFactory
static displayDroids(SentryDroid_B[] dl)	Calls the displayInfo() method of all droids in the list dl	Sentry_BFactory
static displayDroids(SentryDroid_C[] dl)	Calls the displayInfo() method of all droids in the list dl	Sentry_CFactory
displayInfo()	Prints to the shell all the attributes of the DroidFactory object in the format shown on slide 11	DroidFactory
displayInfo()	Prints to the shell all the attributes of the Arm object in the format shown on slide 11	Arm
displayInfo()	Prints to the shell all the attributes of the Body object in the format shown on slide 11	Body
displayInfo()	Prints to the shell all the attributes of the Leg object in the format shown on slide 11	Leg
displayInfo()	Prints to the shell all the attributes of the SentryDroid_A object in the format shown on slide 11	SentryDroid_A
displayInfo()	Prints to the shell all the attributes of the SentryDroid_B object in the format shown on slide 11	SentryDroid_B
displayInfo()	Prints to the shell all the attributes of the SentryDroid_C object in the format shown on slide 11	SentryDroid_C
displayInfo()	Prints to the shell all the attributes of the ImperialDroid object in the format shown on slide 11	ImperialDroid
runDiagnostic()	Verifies that all components of the ImperialDroid object are 'Operational' and prints '[DroidID] is Functioning Normally' to the Shell otherwise prints the part of the Droid that is not operational and '[DroidID] is Malfunctioning' to the shell.	ImperialDroid

# Special Instructions

```
def getOperational(self):
    return self.__Operational
def setOperational(self,v):
    self.__Operational = v
```

Format all getters and setters in this manner

The displayInfo() for ImperialDroid would not contain the header.

```
***Sentry Droid Alpha***
Droid ID: X
Droid Type: Alpha
Head Operational: True
Upper Torso Operational: True
Left Arm Operational: True
Right Arm Operational: True
Left Leg Operational: True
Right Arm Operational: True
```

Display droid attributes in this format for all methods that require displaying information about droids.

# Testing

```
def main():
     SAF = Sentry AFactory('F-1')
    DL = SAF.buildDroids_A(2)
     Sentry_AFactory.displayDroids(DL)
main()
                                               ***Sentry Droid Alpha***
                                               Droid ID: X
                                               Droid Type: Alpha
                                               Head Operational: True
                                               Upper Torso Operational: True
                                               Left Arm Operational: True
                                               Right Arm Operational: True
                                               Left Leg Operational: True
                                               Right Arm Operational: True
                                               ***Sentry Droid Alpha***
                                               Droid ID: X
                                               Droid Type: Alpha
                                               Head Operational: True
                                               Upper Torso Operational: True
                                               Left Arm Operational: True
                                               Right Arm Operational: True
                                               Left Leg Operational: True
```

Right Arm Operational: True