**Object-oriented framework**

* **Two basic programming paradigms:**
* **Procedural – step by step structures**
* **Object-oriented**

**Combining data and functionally and wrap it around what is called object.**

* **Classes and objects are the main aspects of [programming**
* **Class creates new type**
* **Objects are instances of the class**
* **Create an object and then modify it**
* **Instances(object)**
* **Must be a beloved variable. And it can store**
* **Variables that belong to an object or class are called fields**
* **Function is called methods**
* **This terminology is very important because it helps us to differentiate**
* **Blueprint- class instances**
* **Can create int in int group**
* **Functions-what object do**
* **Every object has a unique self**
* **Fields are tow type- belongs to each instance(object) of the class**
* **Or they belong to the class itself**
* **Instance variables and class respectively**
* **Class defined with class**
* **Class methods and fields are listed in indented block**

**Python scopes and namespace**

* **A namespace is a mapping from names to objects**
* **Implemented as python dictionaries, but normally not noticeable in any way**
* **Dic- curly brackets, key value pairs**

**Example**

* **Built-in names built- in exception names**
* **Global names are in module**
* **No relation between different namespace**
* **In the expression modname.funcname**
* **Mod name is a module object and funcname is an attribute of it**
* **Created at different moment and different lifetime**
* **When the python interpreter starts up, and it never deleted**
* **Read from a script file or interactively are considered part of a module called \_main\_ - so they have their own global namespace**
* **Built-in names actually also live in a module, this is called \_builtin\_**
* **Local namespace for a function is created- calling function**
* **And deleted-when the function returnes or raises an eception that is not handled within the function**
* **Recursive invocation each have their own local namespace**
* **Without comparing any other value, a function call itself**
* **A scope is a textual region of a python programming where namespace**
* **Enclosing scope, the middle scope**
* **Local scope references the local names of the current function**
* **New names will be local scope**

**The self**

* **The have an extra variable that has to be added to the beginning of a perimeter list**
* **But we do not give a value for this perimeter when we call the method**
* **This particular variable refers to the object itself**
* **And by convention, it is given the name self**
* **Can give any name for this perimeter**
* **We use the name self**
* **Python provide automatically this value in this function perimeter list**

**The \_init\_ method**

* **\_init is called immediately after an instance of the class is created**
* **Look like a constructor**
* **But \_init\_is the closest**

**Class and object variables**

* **Def howMany(self):**
* **“print the current population”**

**#ther will always be at least one person**

**If person.population==1:**

**Print”I am the one person here”**

**Special class methods**

**Inheritant**

* **Once of the majoe benefits of object oriented programming**

**Multiple inheritance**

* **Python suppots alimiyed form of a multiple inheritance as well**
* **A class definition with multiple base classes looks as follows:**
* **Class DerivedClassName(base1, base2, base3):**
* **<statement-1>**

**<statement-N>**

* **The only rule necessary to explain the semantics is the resolution rule used for class references**

**Requirement analysis**

Sho