

# Simple reflex vacuum cleaner

ENG. Radwa Taher

# Python Classes/Objects

- ▶ Python is an object oriented programming language.
- ▶ Almost everything in Python is an object, with its properties and methods.
- ▶ A Class is like an object constructor, or a "blueprint" for creating objects.

# Create a Class

```
class MyClass:  
    x = 5  
  
print(MyClass)  
|
```

```
<class '__main__.MyClass'>
```

# Create Object

```
class MyClass:  
    x = 5
```

```
p1 = MyClass()  
print(p1.x)
```



5

# The `__init__()` Function

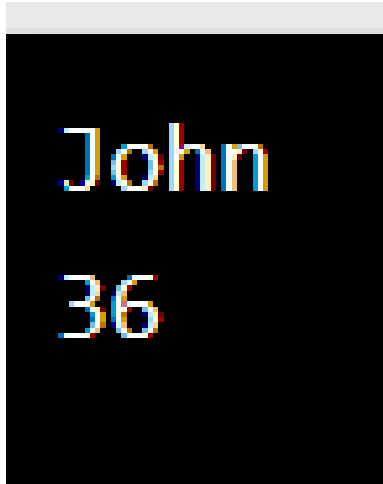
- ▶ The examples above are classes and objects in their simplest form, and are not really useful in real life applications.
- ▶ All classes have a function called `__init__()`, which is always executed when the class is being initiated.
- ▶ Use the `__init__()` function to assign values to object properties, or other operations that are necessary to do when the object is being created:

Create a class named Person, use the `__init__()` function to assign values for name and age:

```
class Person:  
    def __init__(self, name, age):  
        self.name = name  
        self.age = age
```

```
p1 = Person("John", 36)
```

```
print(p1.name)  
print(p1.age)
```



John  
36

# Object Methods

- ▶ Objects can also contain methods. Methods in objects are functions that belong to the object.
- ▶ Let us create a method in the Person class:

```
class Person:
    def __init__(self, name, age):
        self.name = name
        self.age = age

    def myfunc(self):
        print("Hello my name is " + self.name)

p1 = Person("John", 36)
p1.myfunc()
|
```

Hello my name is John

# The self Parameter

- ▶ The `self` parameter is a reference to the current instance of the class, and is used to access variables that belong to the class.
- ▶ It does not have to be named `self`, you can call it whatever you like, but it has to be the first parameter of any function in the class:



# The self Parameter

```
class Person:
    def __init__(mysillyobject, name, age):
        mysillyobject.name = name
        mysillyobject.age = age

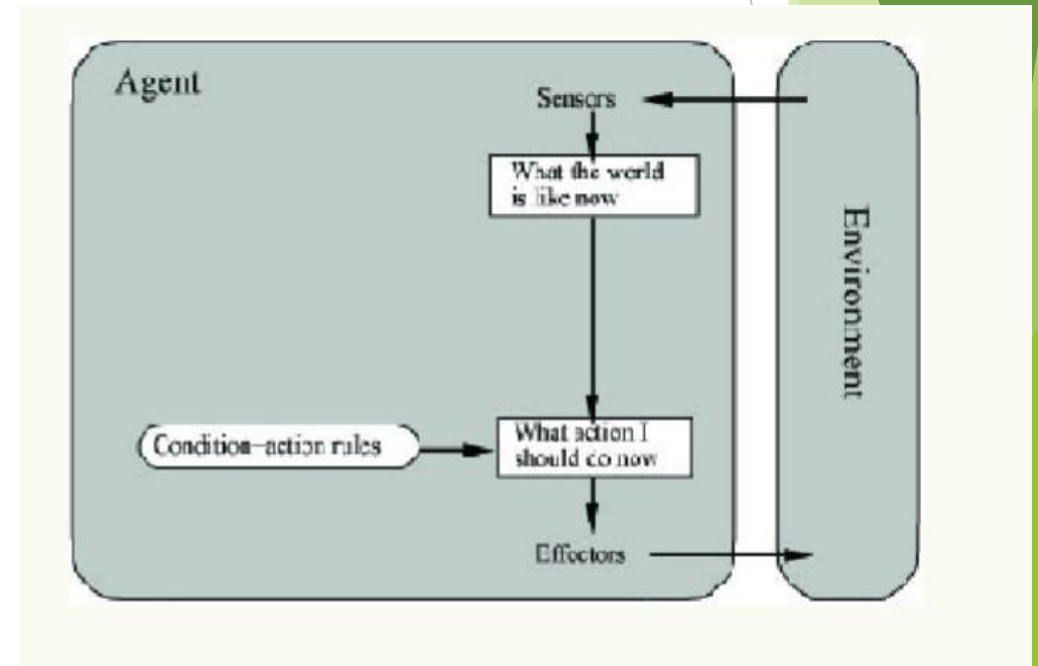
    def myfunc(abc):
        print("Hello my name is " + abc.name)

p1 = Person("John", 36)
p1.myfunc()
```

Hello my name is John

# 1. Simple Reflex Agents:

- ▶ A **Simple Reflex Agent** operates purely based on the current percept and a set of predefined rules or conditions.
- ▶ These agents do not have memory or any knowledge about the history of percepts, nor do they maintain an internal model of the environment.



## 2. Model-Based Reflex Agent

- A Model-Based Reflex Agent extends the capabilities of a simple reflex agent by maintaining an internal state or a model of the world.
- This internal model helps the agent remember what parts of the environment it has already perceived or interacted with, and it uses this information to make more informed decisions.

