Nigel Nicholas
1155088791

Advantages and Disadvantages of Dynamic Typing

1. Advantages of Dynamic Typing:

- More generic code can be written. In other words, functions can be applied on arguments of different types.Example:

```
def function(num1, num2):
        return num1/num2
function(5, 2.0)
```

Since, there aren't any data type included, upon passing the 2 parameters to the function provided above, it is indeed valid. It makes it easier for you to pass numbers especially, since you can have either integer or floating point number in several cases. It makes it easier for you to input some numbers and do operations in the above case.

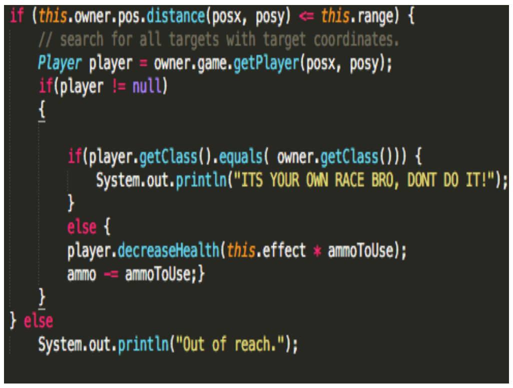
- Possibilities of mixed type collection data structures.
  With dynamic typing, and in python, there is the existence of tuple. Tuple is a list of elements that may contain different type. Example:

```
Tuple1 = ( 'math' , 3 , 'hello' , 2.0)
Tuple1[2]  #'hello'
```

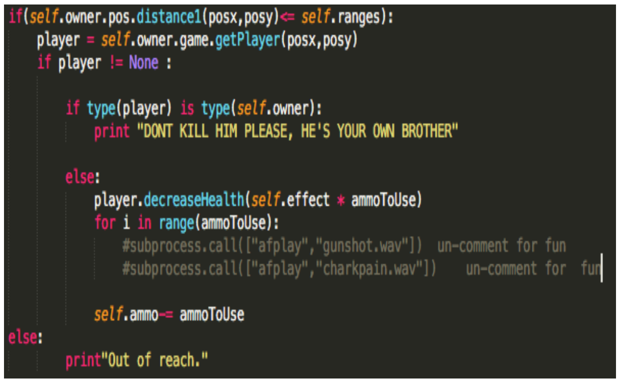
  With this, a list of variable can be of any data type.

2. Why Python is better than Java:

a) Use of indentation in Python



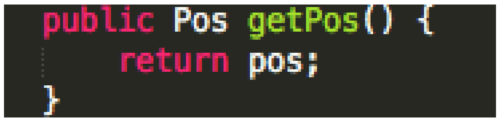
```
if (this.owner.pos.distance(posx, posy) <= this.range) {
    // search for all targets with target coordinates.
    Player player = owner.game.getPlayer(posx, posy);
    if(player != null)
    {

        if(player.getClass().equals( owner.getClass())) {
            System.out.println("ITS YOUR OWN RACE BRO, DONT DO IT!");
        }
        else {
        player.decreaseHealth(this.effect * ammoToUse);
        ammo -= ammoToUse;}
    }
} else
    System.out.println("Out of reach.");
```



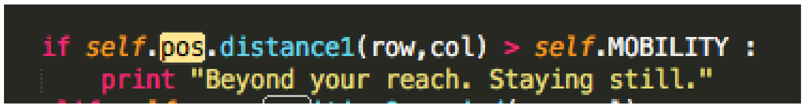
```
if(self.owner.pos.distance1(posx,posy)<= self.ranges):
    player = self.owner.game.getPlayer(posx,posy)
    if player != None :

        if type(player) is type(self.owner):
            print "DONT KILL HIM PLEASE, HE'S YOUR OWN BROTHER"

        else:
            player.decreaseHealth(self.effect * ammoToUse)
            for i in range(ammoToUse):
                #subprocess.call(["afplay","gunshot.wav"])  un-comment for fun
                #subprocess.call(["afplay","charkpain.wav"])    un-comment for  fun

            self.ammo-= ammoToUse
else:
        print"Out of reach."
```

These 2 codes when run are the same. This is taken from source code, rifle.java and rifle.py respectively. 1 of the advantages of using python is that code blocks are seen by its indentation that improves readability. Unlike Java where you have to include Opening and Closing braces to represent compound statement, it may make user slightly confused with too many braces especially in very nested loops.

b) Setter and Getter Methods:



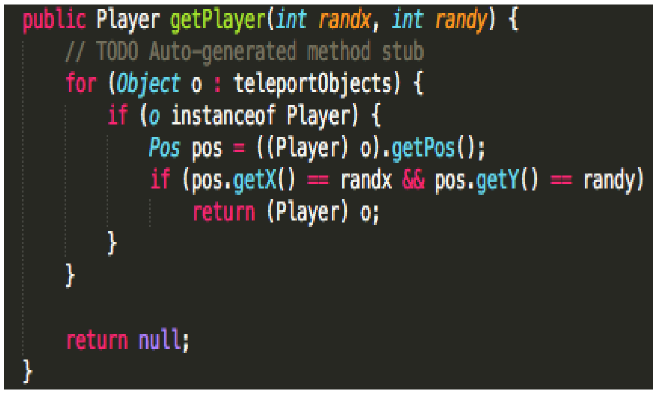
```
public Pos getPos() {
    return pos;
}
```



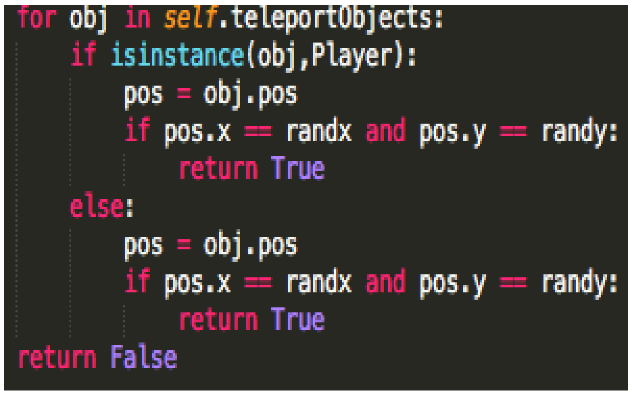
```
if self.pos.distance1(row,col) > self.MOBILITY :
    print "Beyond your reach. Staying still."
```

In Java, it's well known that there is data encapsulation such that a variable may be hidden or restricted of access from other class, and therefore to get these private variables, you need to access the public method setter and getter. This doesn't happen in python, since there aren't any data encapsulation. What happens in python is that you access attributes directly like the picture on the right. This saves time because calling a function takes time.

3. Further advantages of Dynamic and Duck Typing



```
public Player getPlayer(int randx, int randy) {
    // TODO Auto-generated method stub
    for (Object o : teleportObjects) {
        if (o instanceof Player) {
            Pos pos = ((Player) o).getPos();
            if (pos.getX() == randx && pos.getY() == randy)
                return (Player) o;
        }
    }

    return null;
}
```



```
for obj in self.teleportObjects:
    if isinstance(obj,Player):
        pos = obj.pos
        if pos.x == randx and pos.y == randy:
            return True
    else:
        pos = obj.pos
        if pos.x == randx and pos.y == randy:
            return True
return False
```

In Java, like the code above, you use type casting to object o in order to use the superclass' main function. Meanwhile, in python, they would simply access the object's method immediately. This is the characteristic of polymorphism without inheritance.