```
class NPC {

  String name;
  int mapMurd;
  int murTol;
  int murderPropensity;
  int mapSteal;
  int stealTol;
  float theftPropensity;
  int kills;
  int wealth;
  boolean isAlive;
  boolean inRoom; //for implementing muliple rooms
  boolean witnessMurder;
  boolean beingObserved;
  String roomNameIn;

  boolean currInteracting; //for implementing observer and simultaneous interactions


  NPC(String _name, int _murTol) {

    this.name = _name;
    this.mapMurd = kills;
    this.murTol = _murTol; //how okay they are with murder
    // boolean currInteracting = false; //this is for when multiple agents are interacting with
each other
    this.murderPropensity  = 0;
    this.isAlive = true;
    this.roomNameIn = null;
  }

  void returnName() {
    String Name = name;
    print("Char name is " + Name);
  }

  void interact(NPC b) {
    if(isAlive){
```

```
    if (roomNameIn == b.roomNameIn) {
      if (b.isAlive) {
        // currInteracting = true;
        int doIt;
        doIt = int(random(1, 100));
        if (doIt <= murderPropensity || murTol == 100) { //initialises a random int out of 100
and checks if it's <= their murderPropensity. If lower they make the kill.
          b.kill();
          kills++;
          println(name + " has killed " + b.name);
        } else {
          println(name + " talks to " + b.name);
          talk(b);
        }
      } else {
        println(name + "pushes a dead body around");
      }
    }
    }
    else{
     println(name + " is dead");
    }
  }

  void interact() {
  }

  void camObserve(Room a) { //implementing a function for watching an entire room,
this function watches the entire room.
    NPC killer;             //Developing this into a function for individuals watching agents
    String killerID;        //Periodically
    int killed;
    killed = kills;
    for (int i = 0; i < a.occupants.size(); i++) {
      if (killed > 0) {
        killer = a.occupants.get(i);
        killerID = killer.name;
        println("Oh no" + killerID + " has killed someone");
      }
```

```
  }
 }


 void talk(NPC b) {
  if (isAlive) {
   if (b.isAlive) {
    int tmpMPropA = murderPropensity;
    int tmpMPropB = b.murderPropensity ; // friendliness compares how likely they are
to murder
    int difference = 0;
    String conversation = "";
    if(tmpMPropB > tmpMPropA){
    difference = tmpMPropB - tmpMPropA;
    }
    else if(tmpMPropA > tmpMPropB){
     difference = tmpMPropA - tmpMPropB;
    }

    if (difference <= 25) {
     conversation = name + " and " + b.name + " have a pleasant chat";
    } else if (difference > 25 && difference <= 50) {
     conversation = name + " and " + b.name + " exchange pleasantries";
    } else if (difference > 50 && difference <= 75) {
     conversation = name + " and " + b.name + " share some differing opinions";
    } else if (difference > 75 && difference <= 100) {
     conversation = name + " and " + b.name + " have a disagreement";
    }

    println(conversation);
   } else {
    println(b.name + " is dead");
   }
  } else {
   println(name + " is dead");
  }
 }

 void steal() {
```

```
  }

  void give() {
  }


  void kill() {
    isAlive = false;
  }

  void witnessMurder() {
  }
  void enterRoom() {
  }


    //return murderPropensity;
    // String roomAgentIn;

    //    boolean isObserved = beingObserved; //for implementing periodic checking of
interactions, rather than a constant watch

    //if (a.roomNameIn != null) {
    //  roomAgentIn = a.roomNameIn;
    //  for (int i = 0; i < roomIn.size(); i++) {
    //    if (roomAgentIn == roomIn.get(i).roomNameIn) {
    //      numOfObservers++;
    //    }
    //  }
    //}
  }
```