Stuck in (a bit of) Traffic

By:
Jennifer Thompson
Ambarish Kunwar
Maria Graciela Molina
Freddy Vasconez
What is the Project?

Model Traffic Jams

- Include multiple different driver behaviour
- Include a graphical display of the traffic
- Object Oriented approach
Task Division

**Initial**
- Freddy: design graphics
- Graciela: structure classes
- Jennifer and Ambarish: design interface
- Work not equally divided

**Final**
- Have group meetings!!!!
- Freddy: continue with Graphics
- Ambarish: bug fixing
- Graciela: improve class functionality
- Jennifer: documentation and presentation
class Car(object):

def __init__(self):
    self.MaxSpeed=randint(1,3)
    self.CurrentSpeed=randint(1,self.MaxSpeed)
    self.MinGap=None
    self.Accel=randint(1,3)
    self.Desacc=randint(1,3)
    self.Position=None

def setPosition(self,pos):
def setCurrentSpeed(self,speed):
def setMinGap(self,gap):
def getCurrentSpeed(self):
def getAccel(self):
def getDesacc(self):
def getMaxSpeed(self):
def getPosition(self):
def Decide(self):
def doAccelerate(self,accel):
def BreakingSpace(self,speed):

class Road(object):

def __init__(self,N):
    self.road = []
    self.size=N
    self.junction=[]

def setJunction(self,pos):
def addCar(self,car,pos):
def getRoadSize(self):
def printRoad(self):
def getRoad(self):
def move(self):
def ActualGap(self):
import exceptions

class PlaceOccupied(Exception):
    def __init__(self, message):
        self.value = message
    def __str__(self):
        return repr(self.value)

def addCar(self, car, pos):
    """Update road list by adding a car object parameter"
    car: car object to be inserted in the road
    pos: position of the car in the road
    """
    for c in self.road:
        if c.getPosition() == pos:
            raise PlaceOccupied("There is a car here")
    car.setPosition(pos)
    self.road.append(car)
Example - Exceptions

• We add Error Handling

```python
while not success:
    fail=False
    try:
        L=input("Enter a road length: ")
    except NameError:
        print "Incorrect input."
        print "Input must be an integer."
        print "press enter to exit\n"
        continue
```
from nose.tools import *

from classes import Car

def test_constructor():
    c = Car()
    assert c.Position == None

def test_set_pos():
    c = Car()
    c.setPosition(5)
    assert c.getPosition() == 5
    c.setPosition(5)
    assert c.getPosition() == 5

def test_decide():
    c = Car()
    c.setMinGap(0)
    assert c.getMinGap()==0
    csetCurrentSpeed(1000)
    c.Decide()
    assert c.getCurrentSpeed()==0
    c.setMinGap(1)
    c.Decide()
    assert c.getCurrentSpeed()==1

def test_doaccelerate():
    c = Car()
    c.setCurrentSpeed(1)
    c.doAccelerate(2)
    assert c.getCurrentSpeed()==min(3,c.getMaxSpeed())
First Milestone

- 1 car in a circle

After debugging, next steps
- new graphics approach
- Add more cars
- Add feature to allow acceleration and deceleration
- Testing
- Documentation
Evolution

- Many cars!!!!
- Not crashing cars!!!
- Actually accelerate and decelerate!!!
- :)
Second Milestone – Two cars

- Run code!!!!!
Using Pydoc/Sphinx

About Traffic Jam

Version 1.0

Traffic Jam 1.0 is a program to model traffic flow. It calculates the behaviour of 2 cars on a circular road. Velocities, starting positions and rate of acceleration for aggressive, normal and cautious drivers on a road.

Car

class Car

This class creates a Car object. Properties: MaxSpeed: Driver’s desired speed. According this value you can have different behaviour.

- 3-Aggressive driver: This means that driver moves at speed of 3 if possible. 2-Average driver: This means that driver moves at speed of 2 if possible. 1-Careful driver: This means that driver moves at speed of 1 if possible.

CurrentSpeed: Current car’s speed MinGap: Gap to the next car Accel: Car acceleration rate Desacc: Car deceleration rate Positon: Car position in the road

- Decide()
  decide if we should accelerate

- getAccel()
  Obtain car’s acceleration value

- getCurrentSpeed()
  Obtain current car’s speed
Possible new features

- Add classes for different cars (to inherit from Car())
- Add classes for behaviour of driver
- More realistic modelling – inputs from experiment
- More detailed output
- Vertices on road
Our job in gource ...