Project Description:

- Name: AI RISK
- My term project will be the game RISK where a user competes with the computer. It will have an interactive user interface where the user can decide where to deploy their troops, whether to attack, and whether to maneuver troops. The computer will calculate the best strategy based on the current state of the map.

Structural Plan:

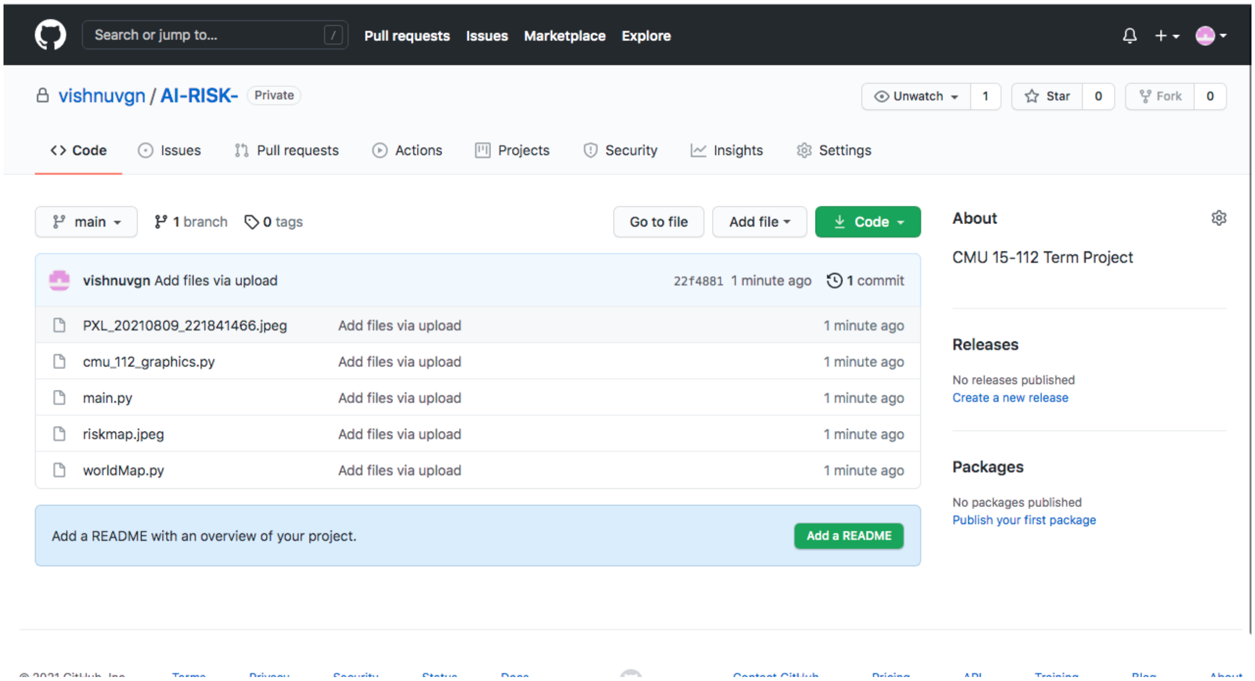
- This project will use object orientation to simplify and better categorize the complex, moving pieces. For instance, I have written a class for the territories so I can store information (like who is currently in that region, how many troops are on, etc.) as attributes of the object. I have also created multiple files in my project to break up my code into smaller and more manageable pieces. For instance I have a drawProject.py file that does all of the drawing components but does not contain the definitions of classes or instantiations of objects.

Algorithmic Plan:

- The decision making of the computer is the trickiest part of the project. The decision making algorithm should scan the current state of the map and take into account many pieces of information such as where enemy troops are, how many of them there are, and if any territories that the computer controls are weak. It should then decide where to place the troops that it gets at the beginning of its turn, decide whether to attack or not (if so, where), and, finally, whether to maneuver its troops or not. I will write this algorithm as if I were the one making the decision. So, I will try to mathematically rate how strong or weak a position is taking in the number of troops the enemy has and the number of troops the computer has. If it is not enough to withstand an attack, then the computer should add troops in that area. Similarly, I write the algorithm so that the computer can pinpoint areas of weakness in its enemies territories and decide how to attack.

Version Control:

For version control, I am using GitHub and uploading my files in the cloud after I am done coding (when I stop for the time being and close VSCode).



vishnuvgn / AI-RISK- Private

vishnuvgn Add files via upload 22f4881 1 minute ago 1 commit

PXL_20210809_221841466.jpeg	Add files via upload	1 minute ago
cmu_112_graphics.py	Add files via upload	1 minute ago
main.py	Add files via upload	1 minute ago
riskmap.jpeg	Add files via upload	1 minute ago
worldMap.py	Add files via upload	1 minute ago

Add a README with an overview of your project.

About

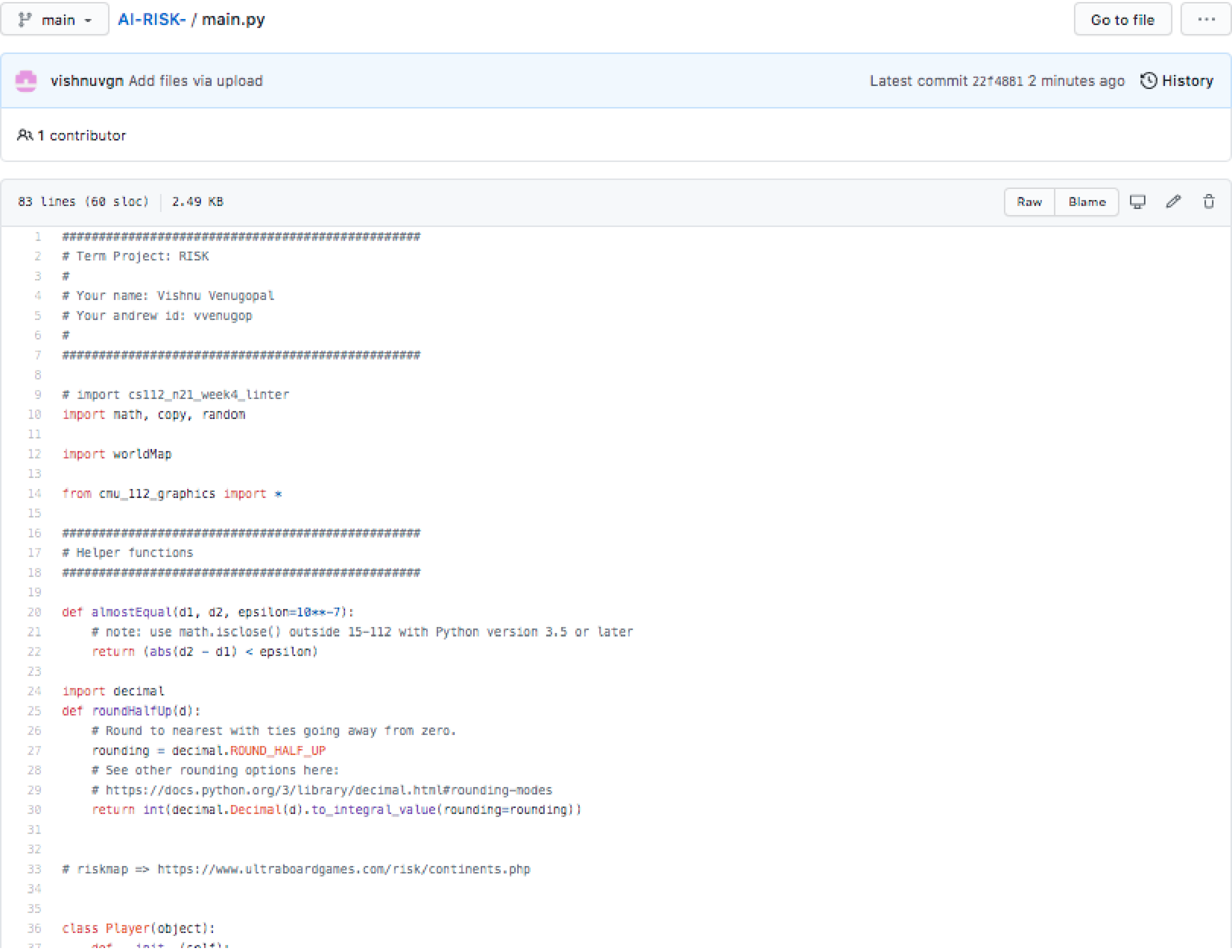
CMU 15-112 Term Project

Releases

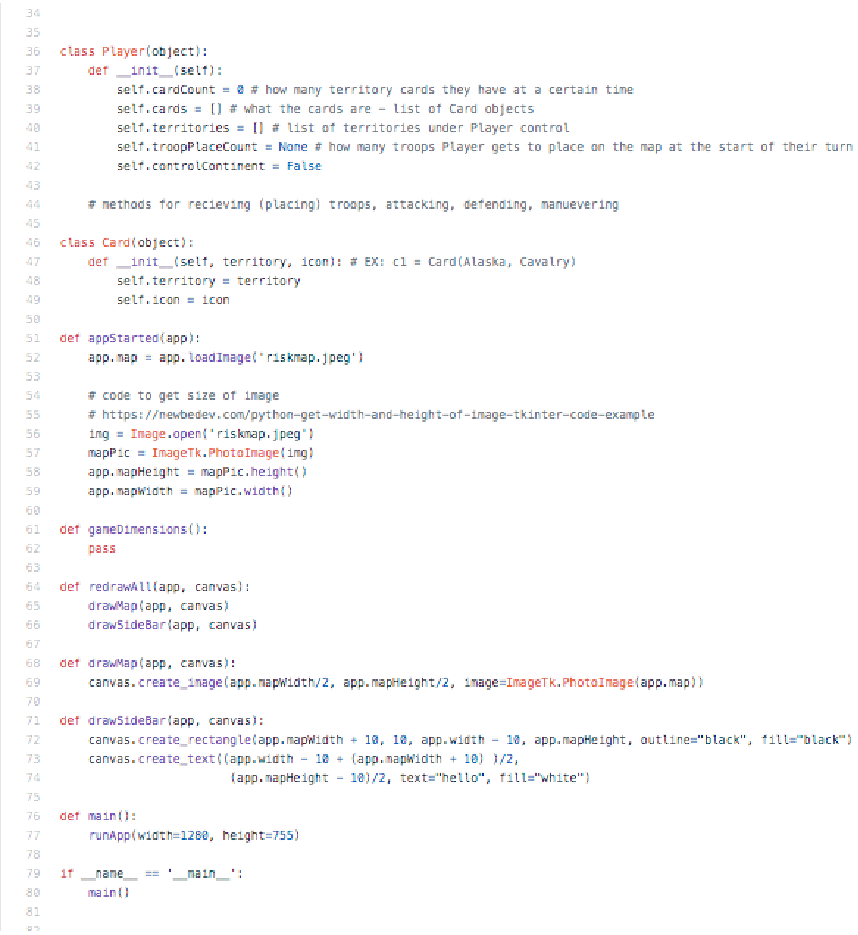
No releases published
Create a new release

Packages

No packages published
Publish your first package


main
AI-RISK- / main.py
Go to file
···
vishnuvgn Add files via upload
Latest commit 22f4881 2 minutes ago
History
1 contributor
83 lines (60 sloc) | 2.49 KB
Raw
Blame

```python
#################################################
# Term Project: RISK
#
# Your name: Vishnu Venugopal
# Your andrew id: vvenugop
#
#################################################

# import cs112_n21_week4_linter
import math, copy, random

import worldMap

from cmu_112_graphics import *

#################################################
# Helper functions
#################################################

def almostEqual(d1, d2, epsilon=10**-7):
    # note: use math.isclose() outside 15-112 with Python version 3.5 or later
    return (abs(d2 - d1) < epsilon)

import decimal
def roundHalfUp(d):
    # Round to nearest with ties going away from zero.
    rounding = decimal.ROUND_HALF_UP
    # See other rounding options here:
    # https://docs.python.org/3/library/decimal.html#rounding-modes
    return int(decimal.Decimal(d).to_integral_value(rounding=rounding))


# riskmap => https://www.ultraboardgames.com/risk/continents.php


class Player(object):
    def __init__(self):
```



```
class Player(object):
    def __init__(self):
        self.cardCount = 0 # how many territory cards they have at a certain time
        self.cards = [] # what the cards are - list of Card objects
        self.territories = [] # list of territories under Player control
        self.troopPlaceCount = None # how many troops Player gets to place on the map at the start of their turn
        self.controlContinent = False

    # methods for recieving (placing) troops, attacking, defending, manuevering

class Card(object):
    def __init__(self, territory, icon): # EX: c1 = Card(Alaska, Cavalry)
        self.territory = territory
        self.icon = icon

def appStarted(app):
    app.map = app.loadImage('riskmap.jpeg')

    # code to get size of image
    # https://newbedev.com/python-get-width-and-height-of-image-tkinter-code-example
    img = Image.open('riskmap.jpeg')
    mapPic = ImageTk.PhotoImage(img)
    app.mapHeight = mapPic.height()
    app.mapWidth = mapPic.width()

def gameDimensions():
    pass

def redrawAll(app, canvas):
    drawMap(app, canvas)
    drawSideBar(app, canvas)

def drawMap(app, canvas):
    canvas.create_image(app.mapWidth/2, app.mapHeight/2, image=ImageTk.PhotoImage(app.map))

def drawSideBar(app, canvas):
    canvas.create_rectangle(app.mapWidth + 10, 10, app.width - 10, app.mapHeight, outline="black", fill="black")
    canvas.create_text((app.width - 10 + (app.mapWidth + 10) )/2,
                        (app.mapHeight - 10)/2, text="hello", fill="white")

def main():
    runApp(width=1280, height=755)

if __name__ == '__main__':
    main()
```

TP2 Update:

- I did not make any design changes.

TP3 Update:

- I made my project such that a person can play with another person in addition to playing with a computer. (Two styles: Person vs. Person and Person vs. Computer) Also, I did not have enough time to have the computer make “smart” moves. It simply makes decisions randomly based on what moves are legal.