**Week 4 Module 6.2 Assignment ForestFireSim 325 Group Project**

Juan Macias Vasquez, Zachary Baker, Joel Atkinson

Bellevue University

CSD325-H323 Advanced Python (2255-DD)

**Jack Lusby**

June 16th, 2025

Module 6.2 Assignment: ForestFireSim 325 Group Project

Results Document

Changes done to Code

1. Added a lake added it in Constants

WATER = '~' # New water tile character

LAKE\_WIDTH = 15

LAKE\_HEIGHT = 6

2. modified the forest to add the new lake in createNewForest()

def createNewForest():

"""Returns a dictionary for a new forest data structure."""

forest = {'width': WIDTH, 'height': HEIGHT}

lake\_start\_x = (WIDTH - LAKE\_WIDTH) // 2

lake\_start\_y = (HEIGHT - LAKE\_HEIGHT) // 2

for x in range(WIDTH):

for y in range(HEIGHT):

if lake\_start\_x <= x < lake\_start\_x + LAKE\_WIDTH and \

lake\_start\_y <= y < lake\_start\_y + LAKE\_HEIGHT:

forest[(x, y)] = WATER # Place lake

elif (random.random() \* 100) <= INITIAL\_TREE\_DENSITY:

forest[(x, y)] = TREE

else:

forest[(x, y)] = EMPTY

return forest

3. Changed the displayForest() to make sure the water was blue and not the generic white text

def displayForest(forest):

"""Display the forest data structure on the screen."""

bext.goto(0, 0)

for y in range(forest['height']):

for x in range(forest['width']):

tile = forest[(x, y)]

if tile == TREE:

bext.fg('green')

print(TREE, end='')

elif tile == FIRE:

bext.fg('red')

print(FIRE, end='')

elif tile == WATER:

bext.fg('blue')

print(WATER, end='')

else:

bext.fg('reset')

print(EMPTY, end='')

print()

bext.fg('reset')

print('Grow chance: {}% '.format(GROW\_CHANCE \* 100), end='')

print('Lightning chance: {}% '.format(FIRE\_CHANCE \* 100), end='')

print('Press Ctrl-C to quit.')

4. Final change was in main() to make sure the water doesn’t change due to fire or spawn a tree

Original was

if ((forest[(x, y)] == EMPTY)

and (random.random() <= GROW\_CHANCE)):

New version

if forest[(x, y)] == WATER:

nextForest[(x, y)] = WATER # Water never changes

elif forest[(x, y)] == EMPTY and random.random() <= GROW\_CHANCE:

Whole file preview

File name: **forestfiresim\_325.py**

# Juan Macias Vasquez, Zachary Baker, Joel Atkinson

# Date 06/15/2025

# Module 6.2

# sitka\_high\_low\_JCMV

"""Forest Fire Sim, modified by Sue Sampson, based on a program by Al Sweigart

A simulation of wildfires spreading in a forest. Press Ctrl-C to stop.

Inspired by Nicky Case's Emoji Sim http://ncase.me/simulating/model/

Updated Version by Juan Macias Vasquez, Zachary Baker, Joel Atkinson"""

import random, sys, time

try:

import bext

except ImportError:

print('This program requires the bext module, which you')

print('can install by following the instructions at')

print('https://pypi.org/project/Bext/')

sys.exit()

# Set up the constants:

WIDTH = 79

HEIGHT = 22

TREE = 'A'

FIRE = '@'

EMPTY = ' '

WATER = '~' # New water tile character

LAKE\_WIDTH = 15

LAKE\_HEIGHT = 6

# (!) Try changing these settings to anything between 0.0 and 1.0:

INITIAL\_TREE\_DENSITY = 0.20 # Amount of forest that starts with trees.

GROW\_CHANCE = 0.01 # Chance a blank space turns into a tree.

FIRE\_CHANCE = 0.01 # Chance a tree is hit by lightning & burns.

# (!) Try setting the pause length to 1.0 or 0.0:

PAUSE\_LENGTH = 0.5

def main():

forest = createNewForest()

bext.clear()

while True: # Main program loop.

displayForest(forest)

# Run a single simulation step:

nextForest = {'width': forest['width'],

'height': forest['height']}

for x in range(forest['width']):

for y in range(forest['height']):

if (x, y) in nextForest:

# If we've already set nextForest[(x, y)] on a

# previous iteration, just do nothing here:

continue

if forest[(x, y)] == WATER:

nextForest[(x, y)] = WATER # Water never changes):

elif forest[(x, y)] == EMPTY and random.random() <= GROW\_CHANCE:

# Grow a tree in this empty space.

nextForest[(x, y)] = TREE

elif ((forest[(x, y)] == TREE)

and (random.random() <= FIRE\_CHANCE)):

# Lightning sets this tree on fire.

nextForest[(x, y)] = FIRE

elif forest[(x, y)] == FIRE:

# This tree is currently burning.

# Loop through all the neighboring spaces:

for ix in range(-1, 2):

for iy in range(-1, 2):

# Fire spreads to neighboring trees:

if forest.get((x + ix, y + iy)) == TREE:

nextForest[(x + ix, y + iy)] = FIRE

# The tree has burned down now, so erase it:

nextForest[(x, y)] = EMPTY

else:

# Just copy the existing object:

nextForest[(x, y)] = forest[(x, y)]

forest = nextForest

time.sleep(PAUSE\_LENGTH)

def createNewForest():

"""Returns a dictionary for a new forest data structure."""

forest = {'width': WIDTH, 'height': HEIGHT}

lake\_start\_x = (WIDTH - LAKE\_WIDTH) // 2

lake\_start\_y = (HEIGHT - LAKE\_HEIGHT) // 2

for x in range(WIDTH):

for y in range(HEIGHT):

if lake\_start\_x <= x < lake\_start\_x + LAKE\_WIDTH and \

lake\_start\_y <= y < lake\_start\_y + LAKE\_HEIGHT:

forest[(x, y)] = WATER # Place lake

elif (random.random() \* 100) <= INITIAL\_TREE\_DENSITY:

forest[(x, y)] = TREE

else:

forest[(x, y)] = EMPTY

return forest

def displayForest(forest):

"""Display the forest data structure on the screen."""

bext.goto(0, 0)

for y in range(forest['height']):

for x in range(forest['width']):

tile = forest[(x, y)]

if tile == TREE:

bext.fg('green')

print(TREE, end='')

elif tile == FIRE:

bext.fg('red')

print(FIRE, end='')

elif tile == WATER:

bext.fg('blue')

print(WATER, end='')

else:

bext.fg('reset')

print(EMPTY, end='')

print()

bext.fg('reset')

print('Grow chance: {}% '.format(GROW\_CHANCE \* 100), end='')

print('Lightning chance: {}% '.format(FIRE\_CHANCE \* 100), end='')

print('Press Ctrl-C to quit.')

# If this program was run (instead of imported), run the game:

if \_\_name\_\_ == '\_\_main\_\_':

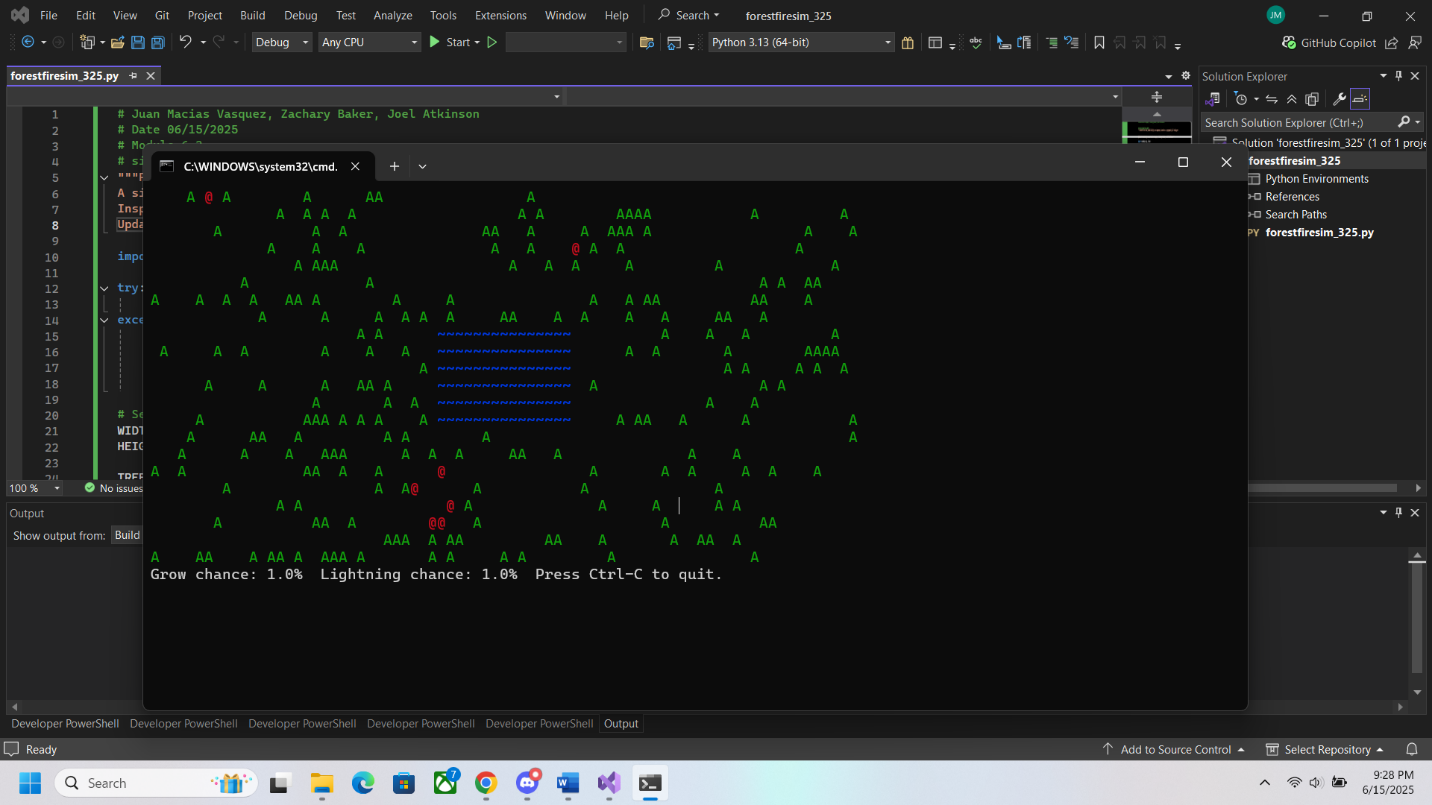
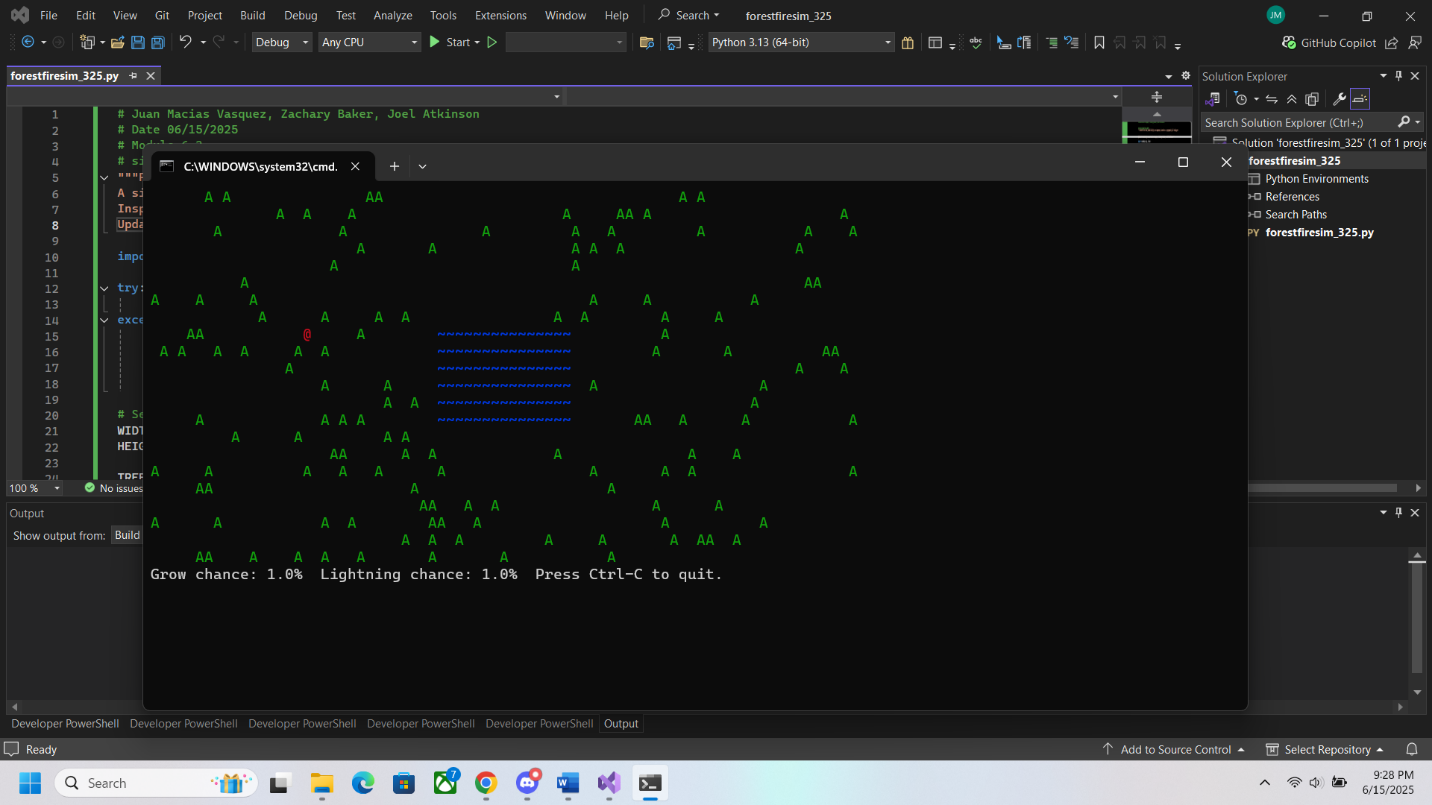
try:

main()

except KeyboardInterrupt:

sys.exit() # When Ctrl-C is pressed, end the program.

Pictures of code running



Flow Chart

A screenshot of a computer screen

AI-generated content may be incorrect.