Open Source Software — CSCI-4470 — Spring 2021
Test 2
April 20, 2021

SOLUTIONS

1. (20 points) The code below is extracted from the **Angry Birds** example we did in class. Refer to it to answer the questions below.

```
1  import os
2  import sys
3  import math
4  import time
5  import pygame
6  current_path = os.getcwd()
7  import pymunk as pm
8  from characters import Bird
9  from level import Level
10 pygame.init()

   # ... Deleted code

11 clock = pygame.time.Clock()
12 running = True
   # the base of the physics
13 space = pm.Space()
14 space.gravity = (0.0, -700.0)

   # ... Deleted Code

15 static_body = pm.Body(body_type=pm.Body.STATIC)
16 static_lines = [pm.Segment(static_body, (0.0, 060.0), (1200.0, 060.0), 0.0)]
17 static_lines1 = [pm.Segment(static_body, (1200.0, 060.0), (1200.0, 800.0), 0.0)]

   # ... Deleted Code

18 space.add_collision_handler(0, 1).post_solve=post_solve_bird_pig
19 space.add_collision_handler(0, 2).post_solve=post_solve_bird_wood
20 space.add_collision_handler(1, 2).post_solve=post_solve_pig_wood

   # ... Deleted Code

21 while running:
22  for event in pygame.event.get():
23         if event.type == pygame.QUIT:
24             running = False
25         elif event.type == pygame.KEYDOWN and event.key == pygame.K_ESCAPE:
26             running = False
27         elif event.type == pygame.KEYDOWN and event.key == pygame.K_w:
28             if wall:
29                 space.remove(static_lines1)
30                 wall = False
```

```
31              else:
32                  space.add(static_lines1)
33                  wall = True
34
35          elif event.type == pygame.KEYDOWN and event.key == pygame.K_s:
36              space.gravity = (0.0, -10.0)
37              level.bool_space = True
38          elif event.type == pygame.KEYDOWN and event.key == pygame.K_n:
39              space.gravity = (0.0, -700.0)
40              level.bool_space = False
41          if (pygame.mouse.get_pressed()[0] and x_mouse > 100 and
42                  x_mouse < 250 and y_mouse > 370 and y_mouse < 550):
43              mouse_pressed = True
44          if (event.type == pygame.MOUSEBUTTONUP and
45                  event.button == 1 and mouse_pressed):

          # ... Deleted Code

46      for beam in beams:
47          beam.draw_poly('beams', screen)
48      dt = 1.0/50.0/2.
49      for x in range(2):
50          space.step(dt) # make two updates per frame for better stability

      # ... Deleted Code

51      pygame.display.flip()
52      clock.tick(50)
53      pygame.display.set_caption("fps: " + str(clock.get_fps()))
```

(a) (2 of 20 points) Which python package encapsulates the game play?

pygame

(b) (2 of 20 points) Which python module ecapsulates the game physics?

pymunk

(c) (4 of 20 points) Assume you want to add a headwind blowing from right to left across the screen. The force due to the headwind is one tenth $\frac{1}{10}$ of the force of gravity in the simulation. Write a short code snippet to start the headwind and reset the gravity to Earth normal, as currently defined in the simulation, when the `'h'` key is pressed. (Headwinds don't occur in space.) Indicate where your code should be inserted in the snippet provided. (Use the provided line numbers.)

Insert the following ater line 40:

```
elif event.type == pygame.KEYDOWN and event.key == pygame.K_n:
   space.gravity = (-70.0, -700.0)
   level.bool_space = False
```

(d) (2 of 20 points) What would you change in your code if there was a tailwind instead (blowing from left to right.)

```
elif event.type == pygame.KEYDOWN and event.key == pygame.K_n:
   space.gravity = (70.0, -700.0)
   level.bool_space = False
```

(e) (3 of 20 points) Which **pymunk** variable represents the *toggleable* `wall` at the right end of the laying field?

static_lines1

(f) (3 of 20 points) Which **pymunk** variable represents the `floor` at the bottom of the laying field?

static_lines

(g) (2 of 20 points) Which python module manages the frames per second?

pygame

(h) (2 of 20 points) Which python module manages the collisions by invoking the collision handlers?

pymunk

2. Assume you have 2 files of data. On each line of the first file is an **identifier** and a single **name** field. On each line of the second file is an **identifier** and a single favorite **drink** field.

   Example files are shown below:

   file1.txt

```
0001a2 Turner
0001a3 Turner
0001a4 Gleyber
0001a5 Voit
0001a6 Stanton
0001a7 Gardner
0001a8 Sanchez
0001a9 Judge
0001b1 Boone
```

   file2.txt

```
0001a2 Tea
0001a3 Coffee
0001a4 Gatorade
0001a6 Water
0001a7 Water
0001a8 Water
0001a9 RC
0001b1 Pepsi
0001b2 Coke
```

   Write a simple python program using `'pymongo'` that reads both files and writes the data to a mongo collection. Data with the same **identifier** should be merged into the same document with **identifier** as its unique document identifier. Use database **yankees** and collection name **menu**. After running your code on the sample data, the following should be stored in the database.

```
> use yankees
switched to db yankees
> db.menu.find()
{"_id":"0001a2", "name":"Turner", "drink":"Tea"}
{"_id":"0001a3", "name":"Turner", "drink":"Coffee"}
{"_id":"0001a4", "name":"Gleyber", "drink":"Gatorade"}
{"_id":"0001a5", "name":"Voit"}
{"_id":"0001a6", "name":"Stanton", "drink":"Water"}
{"_id":"0001a7", "name":"Gardner", "drink":"Water"}
{"_id":"0001a8", "name":"Sanchez", "drink":"Water"}
{"_id":"0001a9", "name":"Judge", "drink":"RC"}
{"_id":"0001b1", "name":"Boone", "drink":"Pepsi"}
{"_id":"0001b2", "drink": "Coke"}
```

```
from pymongo import MongoClient

def insert_or_update(collection, ident, field, value):
    one = collection.find_one({"_id":ident})
    if one != None:
        collection.update_one({"_id":ident}, {"$set":{field: value}})
    else:
        collection.insert_one({"_id":ident, field:value})


if __name__ == '__main__':
    client = MongoClient()
    db = client.yankees
```

```
collection = db.menu

for line in open("file1.txt"):
    name = line.strip().split()
    insert_or_update(collection, name[0], "name", name[1])

for line in open("file2.txt"):
    drink = line.strip().split()
    insert_or_update(collection, drink[0], "drink", drink[1])
```

3. (20 points) Refer to the **Github Action** template for **cmake** shown below while answering the following questions.

```
name: CMake

on: [push, pull_request]

env:
  BUILD_TYPE: Release

jobs:
  build:
    runs-on: ubuntu-latest

    steps:
    - uses: actions/checkout@v2

    - name: Create Build Environment
      run: cmake -E make_directory ${{github.workspace}}/build

    - name: Configure CMake
      shell: bash
      working-directory: ${{github.workspace}}/build
      run: cmake $GITHUB_WORKSPACE -DCMAKE_BUILD_TYPE=$BUILD_TYPE

    - name: Build
      working-directory: ${{github.workspace}}/build
      shell: bash
      run: cmake --build . --config $BUILD_TYPE

    - name: Test
      working-directory: ${{github.workspace}}/build
      shell: bash
      run: ctest -C $BUILD_TYPE
```

(a) (4 of 20 points) What is the command that **builds** the test executables?
Answer:

```
run: cmake --build . --config $BUILD_TYPE
```

(b) (4 of 20 points) What platform and version do the tests run on?
Answer:

ubuntu-latest

(c) (4 of 20 points) Assuming actions are set up correctly, what repository actions cause this testing file to run?
Answer:

When someone pushes or makes a pull request to the repository.

(d) (4 of 20 points) Modify the line that actually executes the tests so that it only runs tests 50 though 100.
Answer:

```
run: ctest -C $BUILD_TYPE -I 50,100
```

(e) (4 of 20 points) In our class notes we discussed errors, faults, and failures. Please discuss the meanings of the 3 terms and say which one testing is designed to discover.

Answer:

An error is made by an Engineer/Algorithm Designer/Implementor. A fault is manifestation of that error in the code. A failure is an incorrect output behavior that is caused by executing a fault. Testing attempts to discover failures in software systems.

4. (20 points) I need a new Docker container to run my favorite programs. Write a `Dockerfile` based off of the latest `python` release. I want it to have:

   programs:

   - `git`
   - `emacs`

   python modules:

   - tensorflow
   - matplotlib

   Finally, I need:

   - A directory `/home/turner` to work in
   - This should be the working directory
   - The container should run `bash` when it is started

   (a) (16/20 points) Write a `Dockerfile` to provision the container described above.

```
FROM python:latest

RUN apt update
RUN apt install --yes git
RUN apt install --yes emacs

# Installing python-pip is optional.
# The python image already has it.

RUN apt install --yes python-pip

RUN pip install tensorflow
RUN pip install matplotlib

RUN mkdir -p /home/turner

WORKDIR /home/turner

CMD ["bash"]
```

   (b) (2/20 points) What command would be used to generate a container named `favorites` from this Dockerfile? You can assume you are in the same directory as the file.

```
docker build -t favorites .
```

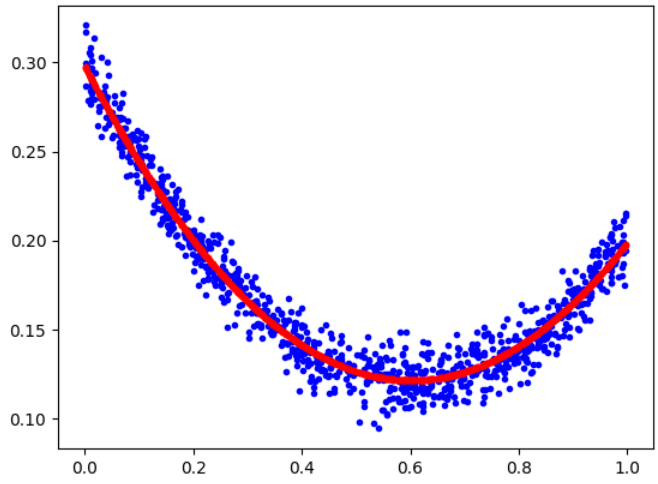
   (c) (2/20 points) What command would be used to run the container you just created so that it comes up in interactive mode with a terminal prompt?

```
docker run -it favorites
```

5. (20 points) In class we used Tensorflow to find the 2 coefficients m and b of the linear equation $y = m * x + b$. For this question, you will modify that code to find the three coefficients a, b, and c of the quadratic equation $y = a * x^2 + b * x + c$. A (slightly) modified version of the code from class is shown below along with a plot of my solution to the quadratic equation $y = 0.5 * x^2 - 0.6 * x + 0.3$. The number of iterations for the plot was 2000 and the learning rate was 0.4. Note that I made two small changes to improve plotting.

   - I changed the number of points in the test data from 100 to 1000.
   - I changed the plot command to use red dots instead of blue lines.

   Modify the code below to solve quadratics. Make sure that you modify the code to generate the noisy data along with the code to calculate the solution.



```
import tensorflow as tf
from tensorflow.keras import Model
import matplotlib.pyplot as plt

def make_noisy_data(m=0.1, b=0.3, n=1000):
  x = tf.random.uniform(shape=(n,))
  noise = tf.random.normal(shape=(len(x),), stddev=0.01)
  y = m * x + b + noise
  return x, y

def predict(x):
  y = m * x + b
  return y

def squared_error(y_pred, y_true):
  return tf.reduce_mean(tf.square(y_pred - y_true))

x_train, y_train = make_noisy_data()
plt.plot(x_train, y_train, 'b.')
plt.show()

m = tf.Variable(0.)
b = tf.Variable(0.)

loss = squared_error(predict(x_train), y_train)
loss_vec = []
print("Starting loss {:.6f}".format(loss.numpy()))

learning_rate = 0.05
steps = 200

for i in range(steps):

  with tf.GradientTape() as tape:
      predictions = predict(x_train)
      loss = squared_error(predictions, y_train)

  loss_vec.append(loss)
  gradients = tape.gradient(loss, [m, b])

  m.assign_sub(gradients[0] * learning_rate)
  b.assign_sub(gradients[1] * learning_rate)


  if i % 20 == 0:
      print("Step {:d}, Loss {:.6f}".format(i, loss.numpy()))

print("Solution: y = {:.6f} * x + {:.6f}".format(m.numpy(), b.numpy()))
plt.plot(range(steps), loss_vec)
plt.show()

plt.plot(x_train, y_train, 'b.')
plt.plot(x_train, predict(x_train), 'r.')
plt.show()
```

Answer:

```
import tensorflow as tf
from tensorflow.keras import Model
import matplotlib.pyplot as plt
```

```
def make_noisy_data(a=.5, b=-0.6, c=0.3, n=1000):
  x = tf.random.uniform(shape=(n,))
  noise = tf.random.normal(shape=(len(x),), stddev=0.01)
  y = a * x * x + b * x + c + noise
  return x, y

def predict(x):
  y = a * x * x + b * x + c
  return y

def squared_error(y_pred, y_true):
  return tf.reduce_mean(tf.square(y_pred - y_true))

x_train, y_train = make_noisy_data()
plt.plot(x_train, y_train, 'b.')
plt.show()

a = tf.Variable(0.)
b = tf.Variable(0.)
c = tf.Variable(0.)

loss = squared_error(predict(x_train), y_train)
loss_vec = []
print("Starting loss {:.6f}".format(loss.numpy()))

learning_rate = 0.4
steps = 2000

for i in range(steps):

  with tf.GradientTape() as tape:
      predictions = predict(x_train)
      loss = squared_error(predictions, y_train)

  loss_vec.append(loss)
  gradients = tape.gradient(loss, [a, b, c])

  a.assign_sub(gradients[0] * learning_rate)
  b.assign_sub(gradients[1] * learning_rate)
  c.assign_sub(gradients[2] * learning_rate)


  if i % 20 == 0:
      print("Step {:d}, Loss {:.6f}".format(i, loss.numpy()))

print("Solution: y = {:.6f} * x^2 + {:.6f} * x + {:.6f}".format(a.numpy(), b.numpy(), c.numpy()))
plt.plot(range(steps), loss_vec)
plt.show()

plt.plot(x_train, y_train, 'b.')
plt.plot(x_train, predict(x_train), 'r.')
plt.show()
```