Project 1 Documentation

Cover Page:

Matt Brown
SpongeBob SquarePants Adventures in Python
Pace University - CS 632P
Project 1
10/30/19

Table of Contents:

Introduction

Welcome to the SpongeBob SquarePants Adventure video game! In this game, you'll be playing the role of SpongeBob SquarePants - the happiest and most fun invertebrate of Bikini Bottom! You'll get to choose from a few different adventures to go on with familiar friends from Bikini Bottom. Each adventure you choose will bring you on a different path - from jelly fishing with Patrick, to practicing karate with Sandy. As SpongeBob himself would say, "I'm Ready!!". Have fun!

What's Included:

Folders and files included in the package:

CS 632P Project 1 Flow Chart.pdf - A flow chart showing the basic flow of the game. Text that appears in the flow chart is not meant to be an exact match of the text the user will see, but rather a visual representation of how the game flows.

Spongebob.py - The main Python file to fun the game (Python source code).

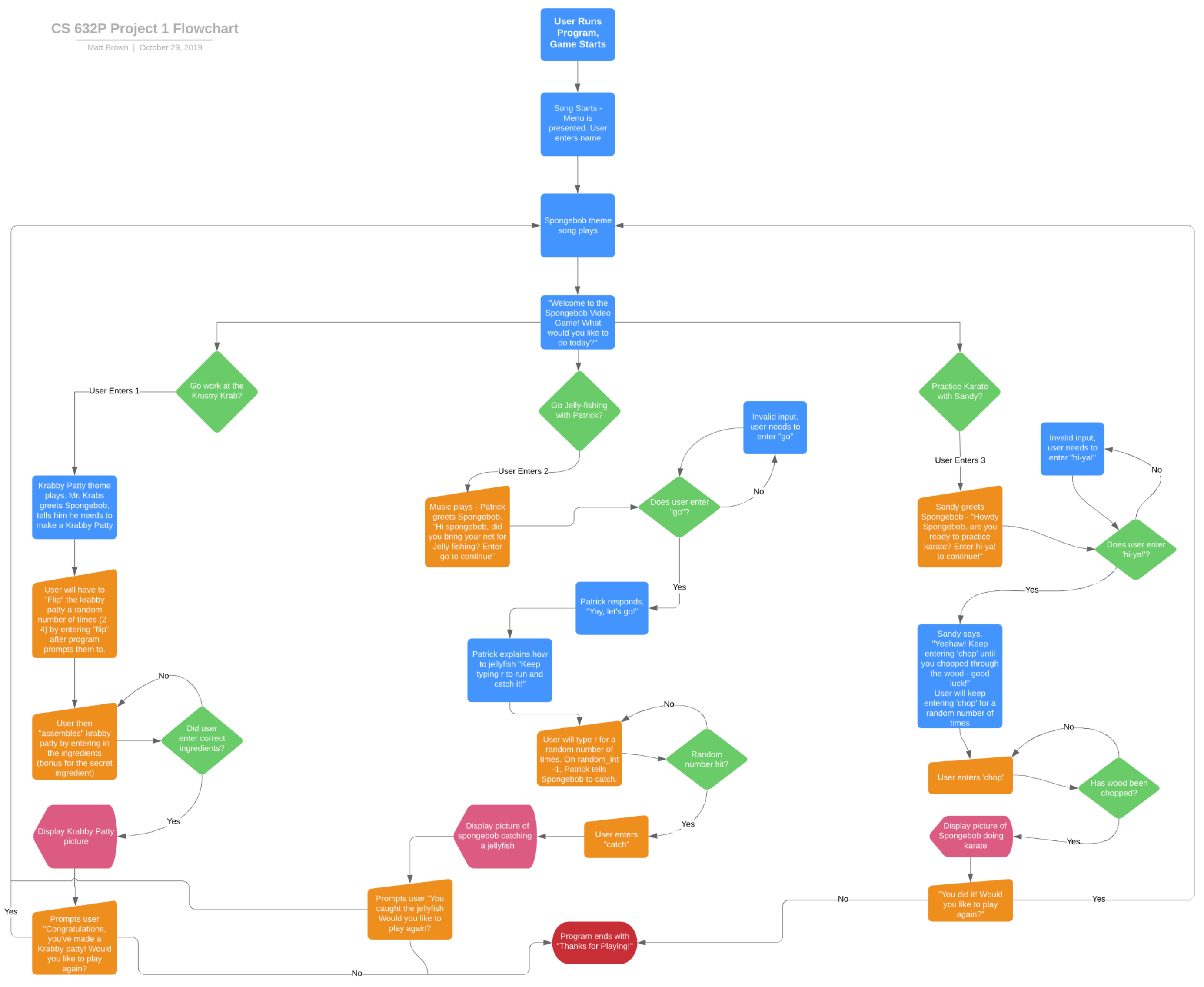
Requirements.txt - Contains a list of package names required to play the game. We will run this file later in a virtual environment so that the game can run properly on any machine.

Images Folder - Folder that contains the necessary images that pop up during the game.

Songs Folder - Contains the song files that are played throughout the game (all in .mp3 format).

`venv` Folder - Virtual environment folder that contains all of the libraries and dependencies needed to run the game. We will go over how to create a venv on any machine running Python 3.7.4 or greater, so that the game can run properly.

Flowchart of gameflow:


CS 632P Project 1 Flowchart
Matt Brown | October 29, 2019
User Runs Program, Game Starts
Song Starts - Menu is presented. User enters name
Spongebob theme song plays
"Welcome to the Spongebob Video Game! What would you like to do today?"
Go work at the Krustry Krab?
User Enters 1
Krabby Patty theme plays. Mr. Krabs greets Spongebob, tells him he needs to make a Krabby Patty
User will have to "Flip" the krabby patty a random number of times (2 - 4) by entering "flip" after program prompts them to.
User then "assembles" krabby patty by entering in the ingredients (bonus for the secret ingredient)
Did user enter correct ingredients?
No
Yes
Display Krabby Patty picture
Prompts user "Congratulations, you've made a Krabby patty! Would you like to play again?
Yes
No
Go Jelly-fishing with Patrick?
User Enters 2
Music plays - Patrick greets Spongebob, "Hi spongebob, did you bring your net for Jelly fishing? Enter go to continue"
Does user enter "go"?
Invalid input, user needs to enter "go"
No
Yes
Patrick responds, "Yay, let's go!"
Patrick explains how to jellyfish "Keep typing r to run and catch it!"
User will type r for a random number of times. On random_int -1, Patrick tells Spongebob to catch.
Random number hit?
No
Yes
User enters "catch"
Display picture of spongebob catching a jellyfish
Prompts user "You caught the jellyfish Would you like to play again?
Program ends with "Thanks for Playing!"
Practice Karate with Sandy?
User Enters 3
Sandy greets Spongebob - "Howdy Spongebob, are you ready to practice karate? Enter hi-ya! to continue!"
Does user enter 'hi-ya!'?
Invalid input, user needs to enter "hi-ya!"
No
Yes
Sandy says, "Yeehaw! Keep entering 'chop' until you chopped through the wood - good luck!" User will keep entering 'chop' for a random number of times
User enters 'chop'
Has wood been chopped?
No
Yes
Display picture of Spongebob doing karate
"You did it! Would you like to play again?"
No
Yes

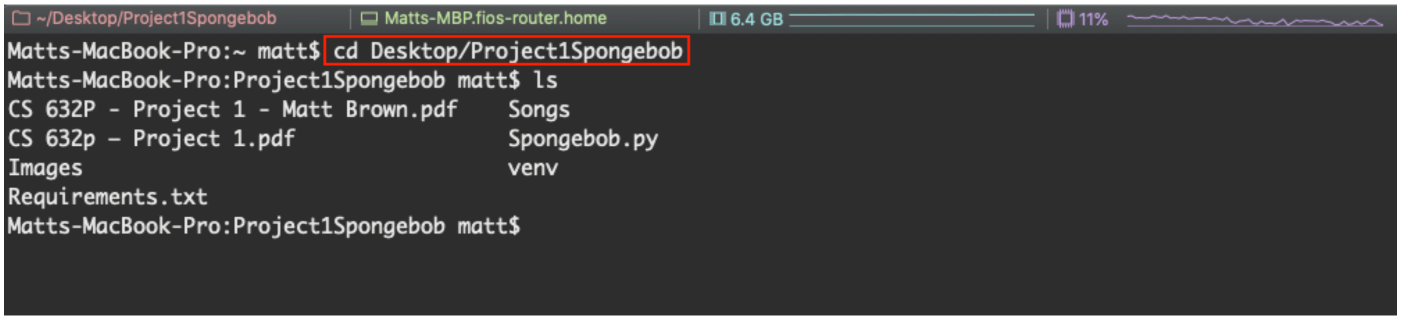
Getting Started/Preparing your environment:

Save the folder titled SpongebobProject1 onto your computer (I recommend the Desktop). The use of this program involves using the terminal(mac/linux) or command prompt(windows). Lets start the virtual environment, which is included in the software package:

1. On macOS, open the terminal applications. To do this, go to Applications --> Utilities folder --> Terminal. You can always search for it using spotlight as well.
2. On Windows, open the start menu and search for cmd. Open the application titled Command Prompt.
3. In the terminal/command prompt, make the current directory the Project1Spongebob folder that was saved by typing in `cd Path/to/file` on macOS, or `cd Path\to\file` on Windows. The below example shows if it was saved on a Desktop (*Tip: to see what's in the folder, you can type*

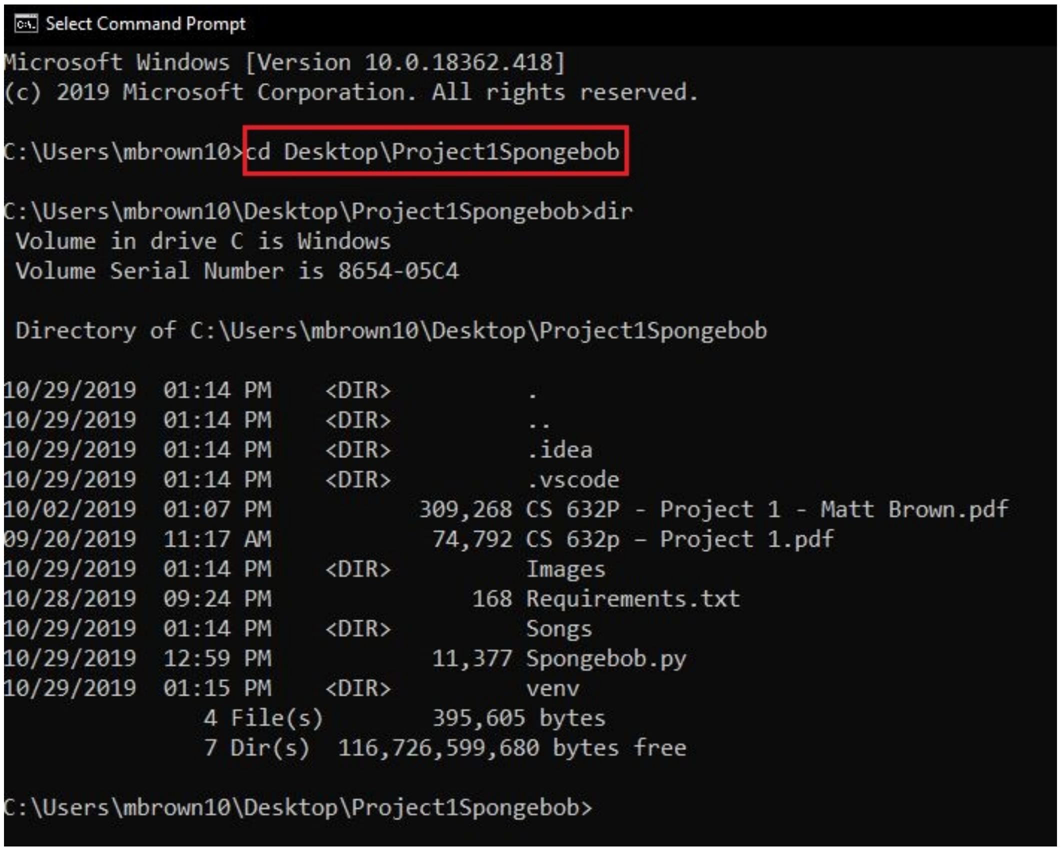
and enter `ls` for macOS and `dir` for Windows):

On macOS:



```
Matts-MacBook-Pro:~ matt$ cd Desktop/Project1Spongebob
Matts-MacBook-Pro:Project1Spongebob matt$ ls
CS 632P - Project 1 - Matt Brown.pdf    Songs
CS 632p – Project 1.pdf                 Spongebob.py
Images                                  venv
Requirements.txt
Matts-MacBook-Pro:Project1Spongebob matt$
```

On Windows:



```
Microsoft Windows [Version 10.0.18362.418]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\mbrown10>cd Desktop\Project1Spongebob

C:\Users\mbrown10\Desktop\Project1Spongebob>dir
 Volume in drive C is Windows
 Volume Serial Number is 8654-05C4

 Directory of C:\Users\mbrown10\Desktop\Project1Spongebob

10/29/2019  01:14 PM    <DIR>          .
10/29/2019  01:14 PM    <DIR>          ..
10/29/2019  01:14 PM    <DIR>          .idea
10/29/2019  01:14 PM    <DIR>          .vscode
10/02/2019  01:07 PM           309,268 CS 632P - Project 1 - Matt Brown.pdf
09/20/2019  11:17 AM            74,792 CS 632p – Project 1.pdf
10/29/2019  01:14 PM    <DIR>          Images
10/28/2019  09:24 PM               168 Requirements.txt
10/29/2019  01:14 PM    <DIR>          Songs
10/29/2019  12:59 PM            11,377 Spongebob.py
10/29/2019  01:15 PM    <DIR>          venv
               4 File(s)        395,605 bytes
               7 Dir(s)  116,726,599,680 bytes free

C:\Users\mbrown10\Desktop\Project1Spongebob>
```

4. Now, we need to activate our virtual environment, or venv for short. This step will allow us to run the game, whether it's macOS, Windows, or Linux. It will also allow us to run the game in case there are different versions of Python or packages from machine to machine. While still in the project folder, enter the following command to activate the virtual environment:

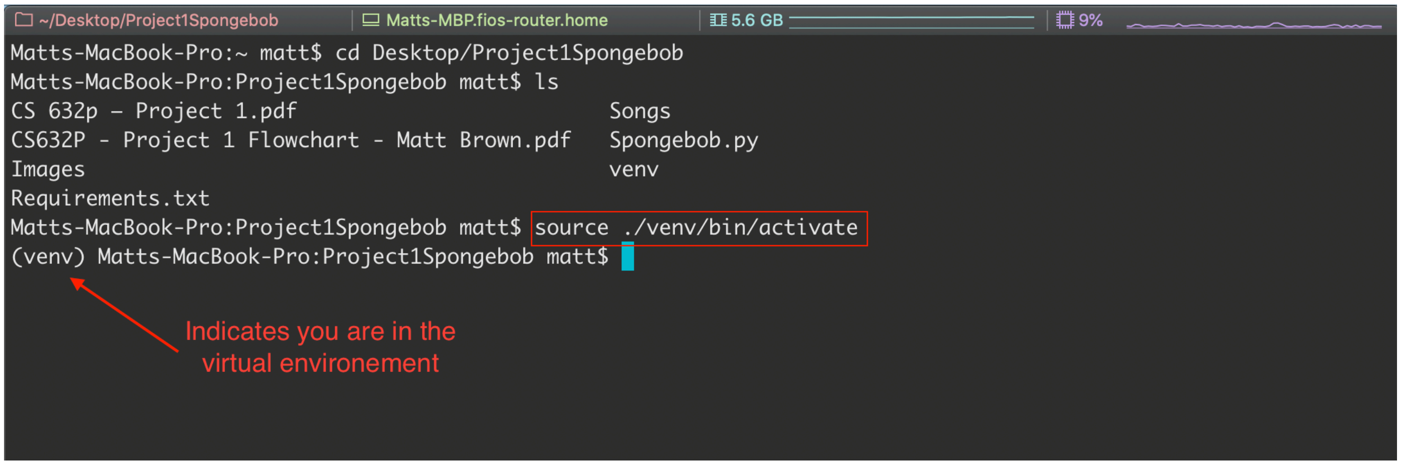
On macOS:

```
source ./venv/bin/activate
```

On Windows:

```
source ./venv/Scripts/activate
```

- You should see something similar to the screenshot below:



```
Matts-MacBook-Pro:~ matt$ cd Desktop/Project1Spongebob
Matts-MacBook-Pro:Project1Spongebob matt$ ls
CS 632p – Project 1.pdf                 Songs
CS632P - Project 1 Flowchart - Matt Brown.pdf   Spongebob.py
Images                                  venv
Requirements.txt
Matts-MacBook-Pro:Project1Spongebob matt$ source ./venv/bin/activate
(venv) Matts-MacBook-Pro:Project1Spongebob matt$
```

5. Now that we have activated our virtual environment, we need to install the proper packages. To do this, enter the command below while in the virtual environment:

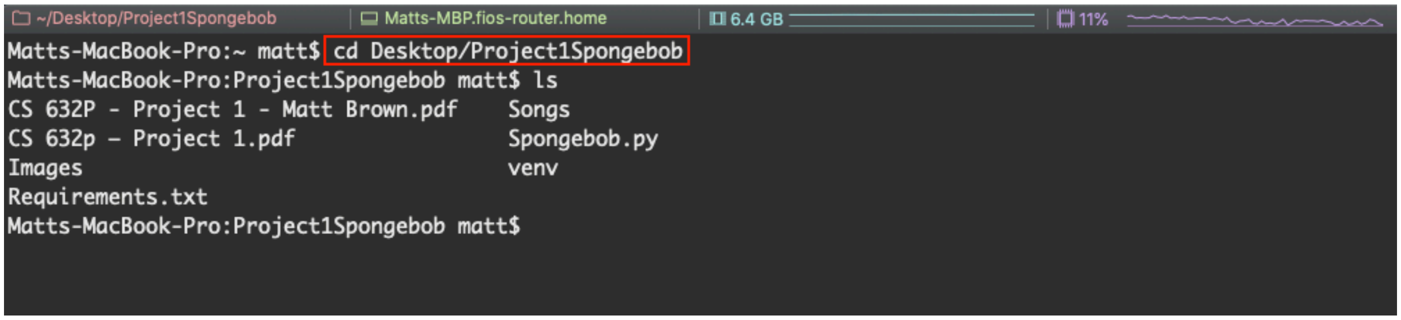
```
pip3 install -r Requirements.txt
```

- This will install the necessary packages automatically. You are now ready to play!
- The steps above only need to be run the first time you set up the virtual environment. After the first time, you can skip right to playing the game.

Running the game

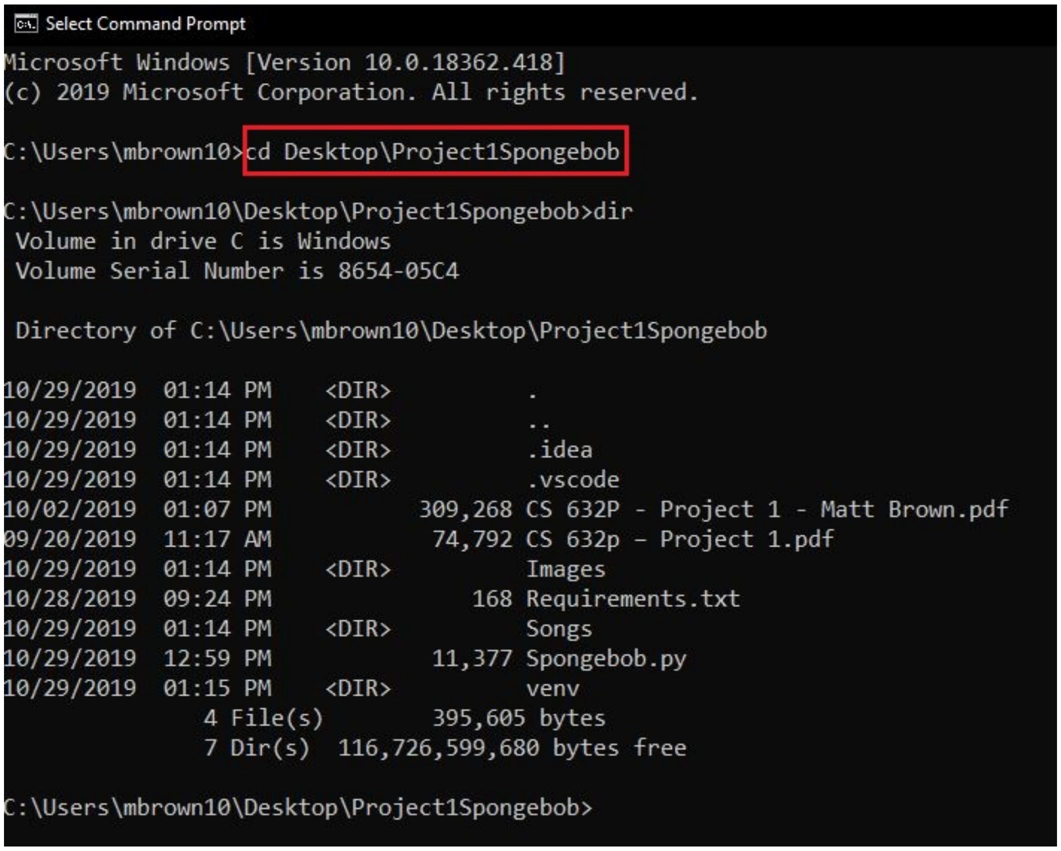
- It's best to start and run the game in the terminal (macOS)/cmd prompt (Windows). To start playing:

    a. Open a new terminal window.
    b. Find the location where you saved `Spongebob.py` (from a previous step in this document, it should be on your Desktop).
    c. In the terminal/command prompt, make the location of the `Spongebob.py` the current directory:
    - For macOS & Linux, enter in: `cd path/to/Project1Spongebob`(Example below):



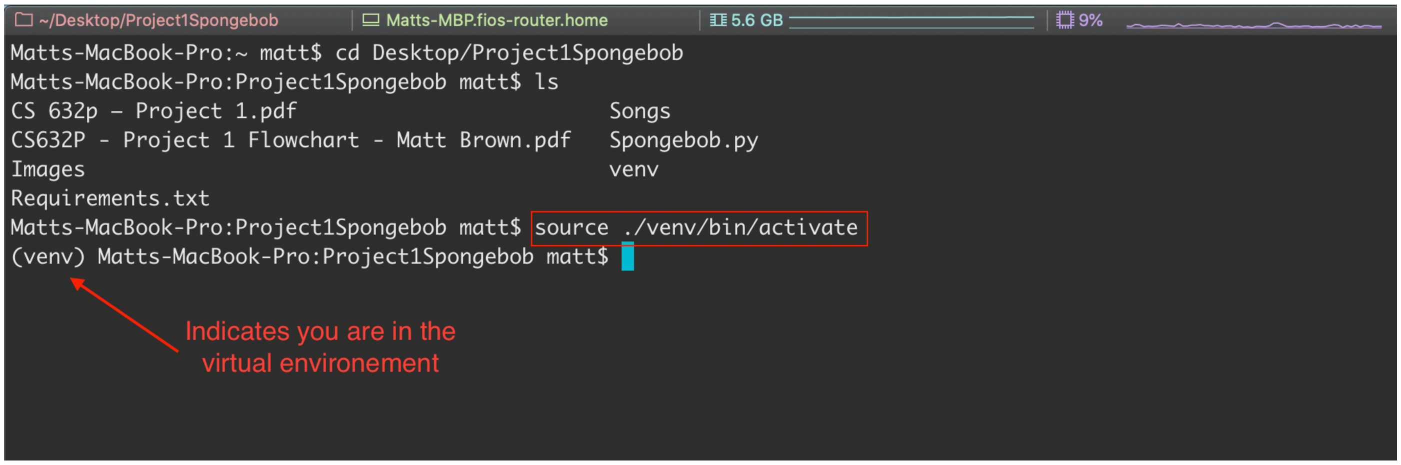
```
Matts-MacBook-Pro:~ matt$ cd Desktop/Project1Spongebob
Matts-MacBook-Pro:Project1Spongebob matt$ ls
CS 632P - Project 1 - Matt Brown.pdf     Songs
CS 632p – Project 1.pdf                  Spongebob.py
Images                                   venv
Requirements.txt
Matts-MacBook-Pro:Project1Spongebob matt$
```

- For Windows, enter in: `cd path\to\Project1Spongebob`(Example below):



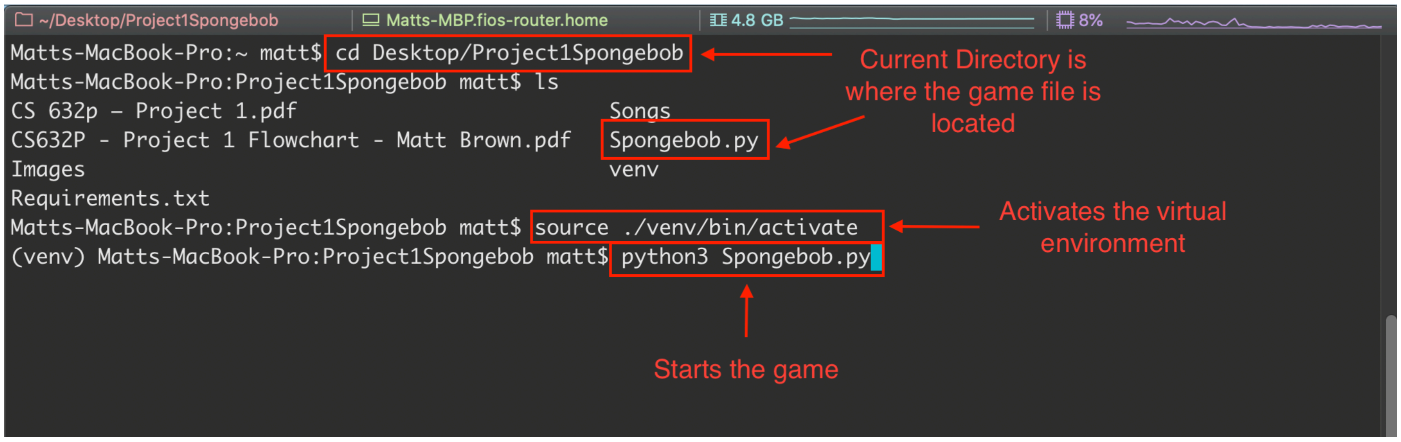
```
Microsoft Windows [Version 10.0.18362.418]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\Users\mbrown10>cd Desktop\Project1Spongebob

C:\Users\mbrown10\Desktop\Project1Spongebob>dir
 Volume in drive C is Windows
 Volume Serial Number is 8654-05C4

 Directory of C:\Users\mbrown10\Desktop\Project1Spongebob

10/29/2019  01:14 PM    <DIR>          .
10/29/2019  01:14 PM    <DIR>          ..
10/29/2019  01:14 PM    <DIR>          .idea
10/29/2019  01:14 PM    <DIR>          .vscode
10/02/2019  01:07 PM           309,268 CS 632P - Project 1 - Matt Brown.pdf
09/20/2019  11:17 AM            74,792 CS 632p – Project 1.pdf
10/29/2019  01:14 PM    <DIR>          Images
10/28/2019  09:24 PM               168 Requirements.txt
10/29/2019  01:14 PM    <DIR>          Songs
10/29/2019  12:59 PM            11,377 Spongebob.py
10/29/2019  01:15 PM    <DIR>          venv
               4 File(s)        395,605 bytes
               7 Dir(s)  116,726,599,680 bytes free

C:\Users\mbrown10\Desktop\Project1Spongebob>
```

4. Activate your virtual environment by typing in `source ./venv/bin/activate` on macOS or `source ./venv/Scripts/activate` on Windows:


~/Desktop/Project1Spongebob
Matts-MBP.fios-router.home
5.6 GB
9%
Indicates you are in the virtual environement

```
Matts-MacBook-Pro:~ matt$ cd Desktop/Project1Spongebob
Matts-MacBook-Pro:Project1Spongebob matt$ ls
CS 632p – Project 1.pdf                     Songs
CS632P - Project 1 Flowchart - Matt Brown.pdf  Spongebob.py
Images                                      venv
Requirements.txt
Matts-MacBook-Pro:Project1Spongebob matt$ source ./venv/bin/activate
(venv) Matts-MacBook-Pro:Project1Spongebob matt$
```

5. To start the game, type in `python3 Spongebob.py` and press enter:



```
Matts-MacBook-Pro:~ matt$ cd Desktop/Project1Spongebob
Matts-MacBook-Pro:Project1Spongebob matt$ ls
CS 632p - Project 1.pdf                        Songs
CS632P - Project 1 Flowchart - Matt Brown.pdf  Spongebob.py
Images                                         venv
Requirements.txt
Matts-MacBook-Pro:Project1Spongebob matt$ source ./venv/bin/activate
(venv) Matts-MacBook-Pro:Project1Spongebob matt$ python3 Spongebob.py
```

~/Desktop/Project1Spongebob
Matts-MBP.fios-router.home
4.8 GB
8%
Current Directory is where the game file is located
Activates the virtual environment
Starts the game

Playing the game:

- Once the game is started, the SpongeBob theme song will start to play. While the song is playing, the lyrics to the theme song are displayed:
  Screen shot

- You will then be presented with a greeting, and a menu:

```
There are so many adventures to go on, but SpongeBob is super busy today and can only choose from 3 of them! Select a number to choose:
Press 1 to go work at the Krusty Krab
Press 2 to go jellyfishing with Patrick
Press 3 to practice karate with Sandy
To quit, just enter the letter q
What adventure would you like to go on today?!
```

- If you enter 1 and choose to work at the Krusty Krab, you'll get to cook and assemble the legendary Krabby Patty! You'll meet Mr. Krabs, who will then ask you to make a Krabby Patty. First, you'll have to cook the patty. You'll be asked to flip the patty over a few times until it's done. Then, you'll need to assemble the Krabby Patty. The game will show you the ingredients needed to make a Krabby Patty. Enter in each ingredient, one-by-one, until you've correctly assembled the Krabby Patty. Don't worry about forgetting what you've entered for ingredients - the game will keep track of your list, and show your progress. Once you've successfully made a Krabby Patty, an image will appear. Close the image to return to the game.

- If you enter 2 and choose to go jellyfishing, you'll hang out with Patrick and get the chance to catch your very own jellyfish in jellyfish fields! Jellyfish don't get caught by just sitting around and waiting for them to fall into your net, so you'll have to run to catch them. When Patrick tells you to, run as fast as you can towards the jellyfish. Patrick, being the best buddy that he is, will let you know once you're close enough to try to catch the jellyfish. Once you catch it, an image will appear. Close the image to return to the game.

- If you enter 3 and choose to go practice karate, you and Sandy will brush up on your karate chopping skills! You and Sandy need to chop through the piece of wood, no matter how many chops it takes! Sandy will let you know when to start chopping, and when you're close to chopping the wood in half. Once you've chopped through the wood, an image will display. Close the image to return to the game. Are you up for the challenge? Hi-ya!

Game Features/Software Design:

Functions:

Below are the custom functions & algorithms that were made for the game:

```
def menu_sleep()
```

This is a simple function that's called during the menu presentation. After each choice presentation, wait for 3.5 seconds, then show the next choice.

```
def show_krabby_patty()
```

This function utilizes the Pillow package to show an image of a Krabby Patty once the user in successful in assembling it during the game.

```
def show_jellyfishing()
```

This function utilizes the Pillow package to show an image of SpongeBob and Patrick in jellyfish fields once the user "catches" the jellyfish in the game it during the game.

```
def show_karate()
```

This function utilizes the Pillow package to show an image of SpongeBob in action in his karate gear once the user completes "chopping" the wood in half during the game.

```
def play_again()
```

This function runs at the end of each adventure choice, and asks the user if they'd like to play again. A input prompt asks the user if they want to play again, and the user does, they press any key to continue. The game asks the user what adventure they'd like to go on, and they can either enter in another adventure choice (1, 2, 3) or 'q' to quit. If the user doesn't enter either of these, the game tells the user that the input is invalid.

```
def krusty_krab()
```

Adventure choice 1 is to go work at the Krusty Krab to cook and assemble a Krabby Patty. Once the user enters 1 as their adventure choice, this function is run. It starts by utilizing the Pygame package to play a sound bite, which is the from the classic episode where SpongeBob goes through training at the Krusty Krab. The song is loaded, and the `mixer.music.play(1)` plays the sound bite, but only once, as we passed `1` as a parameter. The code

```
while mixer_music.get_busy():
  sleep(1)
```

is needed to check if the music stream is playing, and returns `True` if the music stream is actively playing. Pygame is used again to load the Krusty Krab theme song, and this is played throughout the user's time in the Krusty Krab.

Mr. Krabs then tells SpongeBob that there are customers waiting for their Krabby Patties, and it's his job to cook and assemble a Krabby Patty. The code below creates a variable called `number_flips` which will be used as a counter for the number of times the user enters `f`. It's assigned a random number between 3 and 7, and while its value is < 10, the game will prompt an input from the user to "flip" the patty:

```
number_flips = randint(3, 7)
    while number_flips < 10:
        flip = input("Flip! ").lower()
        if flip == 'f':
            number_flips += 1
        elif flip == 'q':
            print("Thanks for playing! We hope you had fun!")
```

```python
            sys.exit()
        else:
            print("Wrong input! You need to press f to flip!")
```

For example, if the random number ends up being 5, the game will prompt the user to flip the patty 5 more times to fully cook it.

Once the patty is done cooking, the user needs to assemble the patty. Here, we are going to utilize lists and tuples for data structures. The list `shown_ingredients` is a list of Krabby Patty ingredients that are show to the user, 1 by 1 with the `for` loop below:

```python
for shown_ingredient in shown_ingredients:
  print(shown_ingredient)
  sleep(1.5)
```

The user needs to assemble the patty using the above ingredients in the correct order. The tuple `correct_order_ingredients` is the correct order in which a Krabby Patty is made. We went with a tuple here, since they are immutable, and the order of a Krabby Patty cannot be changed. Now, we know what you're thinking - "why is the cheese on top of the lettuce and not on top of the patty?". We agree that's it's maddening that this is the order in which a Krabby Patty is made (we had to confirm it with sources close to Mr. Krabs), but this is indeed the way it's made...

An empty list is created called `user_entered_ingredients`. This list will be appended with input from the user as they are assembling the Krabby Patty. We then have the `user_input` which will take the ingredient input from the user, a variable called `counter` to keep track of the index in the list, and a variable called `wrong_input`, which tracks how many times the user messed up their Krabby Patty and entered the wrong sequential ingredient. We then go into the algorithm that determines whether or not the user entered the correct ingredient:

```python
    while counter < 9:
        if user_input == correct_order_ingredients[counter]:
            counter += 1
            user_entered_ingredients.append(user_input)
            print(user_entered_ingredients)
            user_input = input("Correct! What's the next ingredient?! " +
"\n").lower()
        elif user_input == 'q':
            print("Thanks for playing! We hope you had fun!")
            sys.exit()
        else:
            wrong_input += 1
            user_input = input("Incorrect! Guess again! ")
```

The `while` loop conditional is that while `counter` is < 9, keep looping through. We set this to 9 because there are 9 ingredients in a Krabby Patty. If the `user_input` is the same as the counter's index spot in the `correct_order_ingredients` tuple, the `counter` is increased by 1, and the `user_entered_ingredients` list is appended with the input from `user_input`. The game will then print the list that the user is working on, so that they don't forget what they've entered so far, informs the user that they're correct, and asks for the next ingredient. If the user is incorrect, the `wrong_input` value increases by 1, and the program asks the user to guess again.

If, at any time, input is available to the user, and the user enters 'q', the program will end,and thank the user for playing.

Once the user has successfully "made" a Krabby Patty, the `show_krabby_patty()` function is called, and an image of a Krabby Patty appears on screen.

The game then takes the value of `wrong_input`, and displays a message depending on how many times the user incorrectly guessed the wrong ingredient.

```
def jellyfishing_patrick()
```

Adventure choice 2 is to go jellyfishing with Patrick in jellyfish fields. After the user enters 2, this function is run. A new song is started, which is the song you would hear all the time whenever SpongeBob and Patrick would hang out. The program has Patrick ask if the user is ready to go jellyfishing, and to enter 'go' to continue. If the user enters q, the program will end, and if the user enters anything other than 'go' Patrick will tell the user that it's an invalid input, and prompt the user again, asking if they are ready. Whenever there is an input, we are utilizing the `.lower()` function in order to avoid any potential case-sensitive input errors:

```
print("Hey Spongebob, are you ready to go jellyfishing?!")
    sleep(1)
    bring_net = input("Grab your net and enter 'go' to continue!
").lower()
    while bring_net != 'go':
        if bring_net == 'q':
            print("Thanks for playing! We hope you had fun!")
            sys.exit()
        else:
            bring_net = input("Invalid input! Enter 'go' when you're
ready! ").lower()
```

If the user enters 'go', the program continues, and Patrick explains how to catch a jellyfish to the user. The user will have to input 'r' to simulate running towards a jellyfish. Patrick will then let the user know that they are close to the jellyfish, and then tell the user to enter 'catch' to actually catch the jellyfish. `run_input` is a random `int` between 5 and 8, and represents the number of times the user needs to enter in 'r' to run. If the user enters 'r', `run_input` increases by 1, up until the conditional for the `while` loop it's in, which is 15. If the user enters 'q' at any time, the program exits. If the user enters anything other than 'r' or 'q', a message is displayed, letting the user know that the input is invalid and to try again. Once `run_input` hits 14, the game let's the user know that they're close by displaying "You almost got it SpongeBob". After the user enters in the next 'r', Patrick tells the user to enter "catch" to catch the jelly fish. If the user enters "catch", the `show_jellyfishing()` function is called, and the picture of SpongeBob and Patrick in jellyfish fields is displayed.

```
def sandy_karate()
```

Adventure choice 3 is to go practice karate with Sandy, and it works similarly to the `def jellyfishing_patrick()` function.

Libraries Used

Pygame:

The Pygame library is being used to play the music throughout the game. Pygame's `pygame.mixer.music` needs to first be initialized with `mixer.init()`. The music is then loaded, with `mixer.music.load('Path/to/song.mp3')`, with the path to the song file being passed through as a string. .mp3 files work best for this, and other sound files didn't have as much consistency as .mp3 files. The music will start to play with the `mixer.music.play()` method. A parameter can be passed depending on how many times you want to play the song, but sice we're only playing the theme song once, no parameter is needed. In order to check and make sure the music stream is playing, we run `mixer.music.get busy()`, and loop through the theme song/lyrics once to display them. We add `mixer.music.stop()` to stop the music where we want it to. When it's all put together, the code is as follows:

```
from pygame import *

mixer.init()

print("Welcome to the SpongeBob SquarePants Adventure Game! Let's get started!")
```

```python
mixer.music.load('Songs/Theme_Song.mp3')
mixer.music.play()
while mixer.music.get_busy():
    sleep(11)
    print("Who lives in a pineapple under the sea?")
    print("Spongebob Squarepants!")
    sleep(3.5)
    print("Absorbant, and yellow, and porous is he")
    print("Spongebob Squarepants!")
    sleep(3.7)
    print("If nautical nonsense be something you wish")
    print("Spongebob Squarepants!")
    sleep(3.8)
    print("Then drop on the deck and flop like a fish")
    print("Spongebob Squarepants!")
    sleep(4)
    print("Spongebob Squarepants!")
    sleep(2.2)
    print("Spongebob Squarepants!")
    sleep(2.2)
    print("Spongebob Squarepants!")
    sleep(2.2)
    print("Spongebob Squarepants!")
    sleep(10)
    mixer.music.stop()
```

Pillow:

The Pillow package's main feature is to display images when the user has completed one of their adventures. The following code is placed in a function, depending on the situation:

```python
import PIL form Image

with Image.open('Path/to/image.png') as img:
  img.show()
```

Main Body:

```python
adventure_choice = ' '

```

```python
while adventure_choice != 'q':

    adventure_choice = input("What adventure would you like to go on today?! ")

    if adventure_choice == str(1):
        krusty_krab()
        mixer.music.stop()
        if not play_again():
            sys.exit()
    elif adventure_choice == str(2):
        jellyfishing_patrick()
        mixer.music.stop()
        if not play_again():
            sys.exit()
    elif adventure_choice == str(3):
        sandy_karate()
        mixer.music.stop()
        if not play_again():
            sys.exit()
    elif adventure_choice == 'q':
        print("Thanks for playing, we hope you had fun!!")
        sys.exit()
    else:
        print("Invalid input!")
```

Testing:

Known Bugs:

- We have noticed 2 times out of several tests that the music played was very slowed down.
-

Considerations & Enhancements for Version 2.0:

- Add a 4th adventure that involves Squidward or Gary (SpongeBob's snail).
- Make a GUI for the characters to interact with each other.
- Allow the user to play as themself instead of SpongeBob (i.e. ask the user for their name, and have

the characters interact with the user).

Special Thanks:

Special thanks to the pygame and Pillow developers for making the libraries that made this game possible.

Thanks to all my friends and family that helped test this game by playing it.

A very special thanks to my wife, Dalila. Thank you for testing this game several times, and suffering through hearing the SpongeBob theme hundreds of times.