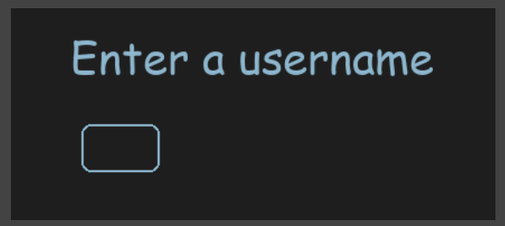
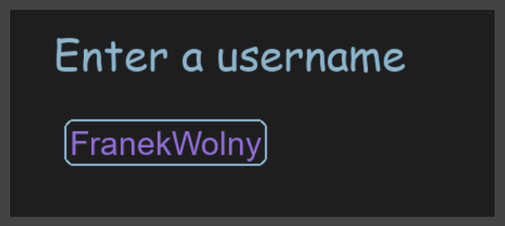
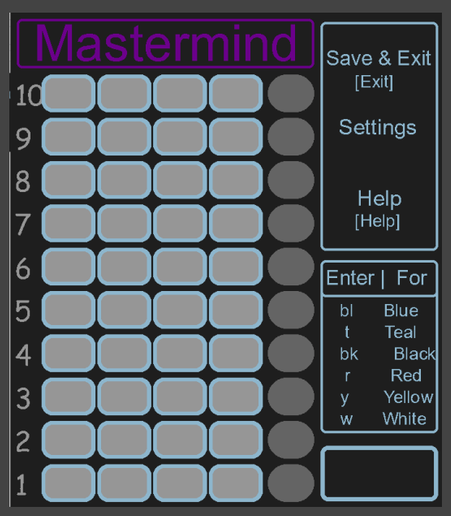


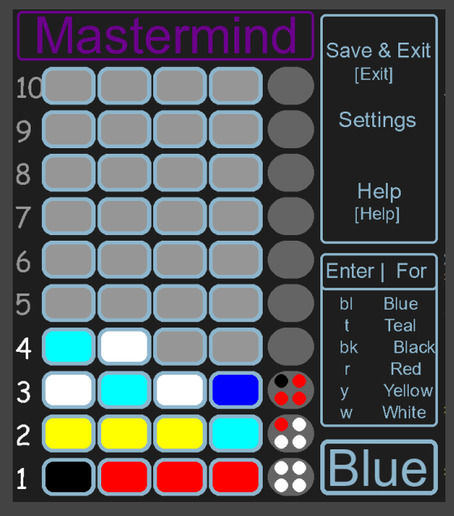


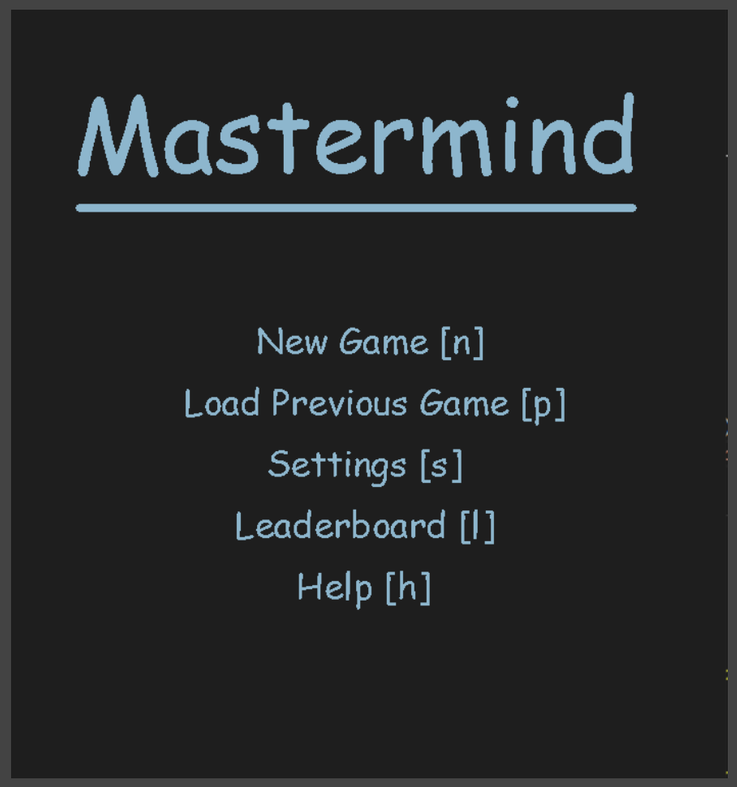
Once a username is entered when starting a new game, a new row is created in Player_Information.csv to which inputs during the game are added and can be read from if the game is closed and opened later

Constant loop reading from the current player's past move data in Player_Information.csv, drawing with the colour entered (if entered)

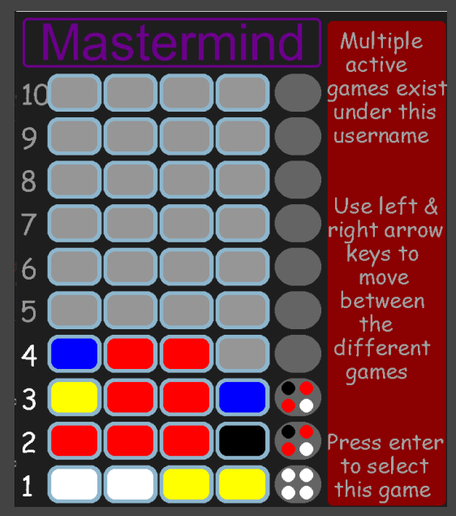
Inputs from the user are recorded in the csv and constantly updated

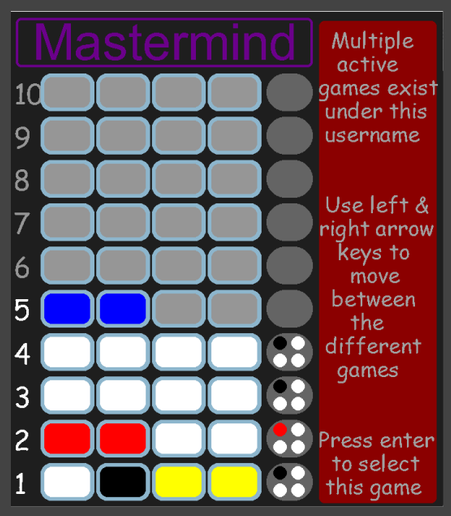






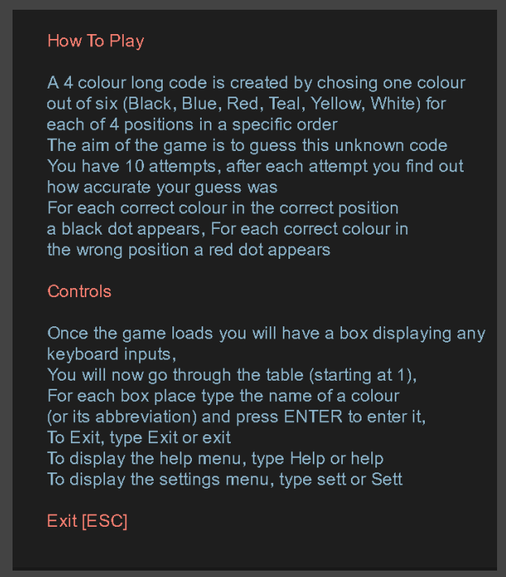
If multiple active games exist under the username entered to play a previous game, then the user can use arrow keys to go through the active games and select one to carry on





Scores are read from Player_Information.csv, the top 5 are printed onto the screen along with their usernames

https://mevie.love/5jxzhizoijbezzmb5vi6e2kxiu7lqdb0opdlbs4w/?pw=z1u80ctz5s



"Help" shows rules and instructions

main.py

```
import pygame
import sys
import random
import time

from csv_operations import append_to_csv,update_csv,get_high_scores,get_usernames,get_usernames_game_code,get_new_game_code,get_game_info,update_status,clean_csv

# Displays Start Menu
def start_menu():
    screen.fill((30,30,30))
    screen.blit(font_1.render('Mastermind', False, pygame.Color('lightskyblue3')),(60,50))
    pygame.draw.rect(screen, pygame.Color('lightskyblue3'), pygame.Rect(63, 192, 550, 8), 0, border_radius=40)
    for line in {"New Game [n]":(240,300),'Load Previous Game [p]':(170,360),'Settings [s]':(250,420),'Leaderboard [l]':(220,480),'Help [h]':(280,540)}.items():
        screen.blit(font_2.render(line[0], False, pygame.Color('lightskyblue3')),line[1])

# Displays Help - Rules and instructions
def help():
    while True:
        clock.tick(60)
        screen.fill((30,30,30))
        count = 0
        with open("Instructions.txt", "r") as f:
            for line in f.readlines():
                if line == "\n" or line[-2] != "&":
                    screen.blit(font_12.render(line[:-1],False,pygame.Color('lightskyblue3')),(50,30+count*30))
                else:
                    screen.blit(font_12.render(line[:-2],False,(250,128,114)),(50,30+count*30))
                count += 1
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                sys.exit(1)
            elif event.type == pygame.KEYDOWN and event.key == pygame.K_ESCAPE:
                return ""
        pygame.display.update()

# Runs Game
def play_game(game_code, leave):
    won = False
    lost = False
    clean_csv()
    accuracy = {}
    invalid = False
    while True:
        clock.tick(60)
        done = False
        text = ''
        while not done:
            screen.fill((30,30,30))
            screen.blit(font_7.render("Mastermind", False, ((107,0,139))),(37,3))
            for dimensions in [(500, 400, 190, 280),(500, 15, 190, 370),(500, 400, 190, 60)]:
                pygame.draw.rect(screen, pygame.Color('lightskyblue3'), pygame.Rect(dimensions), 4, border_radius=8)
            pygame.draw.rect(screen, ((107,0,139)), pygame.Rect(10, 10, 480, 80), 4, border_radius=8)
            for line in {"       [Exit]":95,"        [Set]":1205,"       [Help]":320,"    bl      Blue":465,"     t       Teal ":500,"    bk       Black":535,"     r        Red":570,"    y
Yellow":605,"    w      White":640," Enter |  For":410," Save & Exit":55,"      Help":280,"   Settings":165}.items():
                if line[1] in [280,165]:
                    screen.blit(font_10.render(line[0], False, (pygame.Color('lightskyblue3'))),(500,line[1]))
                elif line[1] in [410,55]:
                    screen.blit(font_8.render(line[0], False, (pygame.Color('lightskyblue3'))),(500,line[1]))
                else:
                    screen.blit(font_9.render(line[0], False, (pygame.Color('lightskyblue3'))),(500,line[1]))
            for event in pygame.event.get():
                if event.type == pygame.QUIT:
                    sys.exit(1)
                elif event.type == pygame.KEYDOWN:
                    if event.key == pygame.K_RETURN:
                        done = True
                    elif event.key == pygame.K_BACKSPACE:
                        text = text[:-1]
                    elif event.unicode in alphabet:
                        text += event.unicode
            if done is True:
                if text.lower() in ["bk","t","bl","r","y","w"]:
                    update_csv(game_code,text.lower())
                    game_info = get_game_info(game_code)
```

```
    elif text.lower() in ["black","blue","teal","red","yellow","white"]:
        update_csv(game_code,name_to_colour_code[text.lower()])
    elif text.lower() == "exit":
        sys.exit(1)
    elif text.lower() == "help":
        help()
    elif text.lower() == "set":
        settings()
    else:
        invalid = True
input_box = pygame.Rect(500, 700, 190, 90)
txt_surface = font_7.render(text, True, pygame.Color('lightskyblue3'))
screen.blit(txt_surface, (input_box.x+10, input_box.y))
pygame.draw.rect(screen, pygame.Color('lightskyblue3'), input_box, 7, border_radius=12)
count = 1
pygame.draw.rect(screen, pygame.Color('lightskyblue3'), pygame.Rect(700, 630, 200, 70), 0, border_radius=17)
game_info = get_game_info(game_code)[2:]
for row in range(730,70,-70):
    try:
        game_info[0]
        screen.blit(font_6.render(str(count), False, ((255,255,255))), (7, 100 +(10-count)*70))
    except (IndexError, KeyError):
        screen.blit(font_6.render(str(count), False, ((150,150,150))), (7, 100 +(10-count)*70))
    guesses = []
    for column in range(50,410,90):
        pygame.draw.rect(screen, pygame.Color('lightskyblue3'), pygame.Rect(column, row, 87, 61), 0, border_radius=17)
        try:
            pygame.draw.rect(screen, pygame.Color(colour_code[game_info[0]]), pygame.Rect(column+6, 6+row, 74, 49), 0, border_radius=12)
            guesses.append(colour_code[game_info[0]])
            game_info.pop(0)
        except (IndexError, KeyError):
            guesses = []
            pygame.draw.rect(screen, pygame.Color((150,150,150)), pygame.Rect(column+6, 6+row, 74, 49), 0, border_radius=12)
    code = real_code(get_game_info(game_code)[0])
    pygame.draw.rect(screen, ((100,100,100)), pygame.Rect(415,row,75,61), 0, border_radius=50)
    if len(guesses) == 4:
        correct_colours = 0
        correct_colours_n_pos = 0
        count_x = 0
        code_copy = code.copy()
        for colour in guesses:
            if colour in code_copy:
                correct_colours += 1
                code_copy.remove(colour)
            if guesses[count_x] == code[count_x]:
                correct_colours_n_pos += 1
            count_x += 1
        correct_colours -= correct_colours_n_pos
        correct_colours_n_pos_copy = correct_colours_n_pos
        for dot_info in [(437,row+16),(437,row+44),(466,row+16),(466,row+44)]:
            if correct_colours_n_pos != 0:
                pygame.draw.circle(screen, ((0,0,0)), dot_info, 11)
                correct_colours_n_pos -= 1
            else:
                if correct_colours != 0:
                    pygame.draw.circle(screen, ((255,0,0)), dot_info, 11)
                    correct_colours -= 1
                else:
                    pygame.draw.circle(screen, ((255,255,255)), dot_info, 11)
        if correct_colours_n_pos_copy == 4:
            won = True
            ccount = count
            update_status(ccount, game_code)
        elif row == 100:
            lost = True
            update_status("Lost", game_code)
    count += 1
pygame.display.update()
if leave is True:
    return ""
elif won is True:
    winning_message(ccount)
elif lost is True:
    losing_message()
elif invalid is True:
    for i in range(1500):
        pygame.draw.rect(screen, ((0,0,0)), input_box, 0, border_radius=12)
        screen.blit(font_4.render("INVALID", False, ((255,0,0))),(505,715))
```

```
                    pygame.display.update()
                invalid = False

# Gets Username input
def input_screen(title):
    input_box = pygame.Rect(80, 140, 180, 43)
    text = ''
    done = False
    while not done:
        clock.tick(60)
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                sys.exit(1)
            elif event.type == pygame.KEYDOWN:
                if event.key == pygame.K_RETURN:
                    done = True
                elif event.key == pygame.K_BACKSPACE:
                    text = text[:-1]
                elif event.unicode in alphabet:
                    text += event.unicode
        screen.fill((30, 30, 30))
        screen.blit(font_4.render(title, True, pygame.Color('lightskyblue3')), (70,50))
        txt_surface = font_5.render(text, True, ((147,112,219)))
        width = max(70, txt_surface.get_width()+10)
        input_box.w = width
        screen.blit(txt_surface, (input_box.x+5, input_box.y+5))
        pygame.draw.rect(screen, pygame.Color('lightskyblue3'), input_box, 2, border_radius=8)
        pygame.display.update()
    return text

# Displays Leaderboard
def leaderboard():
    high_scores = get_high_scores(5)
    while True:
        clock.tick(60)
        screen.fill((30,30,30))
        screen.blit(font_6.render("Leaderboard", False, (250,128,114)),(170,70))
        screen.blit(font_3.render("Exit [ESC]", False, (250,128,114)),(170,500))
        count = 0
        for score in high_scores:
            screen.blit(font_3.render(str(score[0]+" - "+str(score[1])+" tries"), False, pygame.Color('lightskyblue3')),(170,165+(40*count)))
            count +=1
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                sys.exit(1)
            elif event.type == pygame.KEYDOWN:
                if event.key == pygame.K_ESCAPE:
                    return ""
        pygame.display.update()

# Displays Winning Message
def winning_message(score):
    for i in range(3000):
        pygame.draw.rect(screen, ((0,0,0)), pygame.Rect(150, 150, 400, 500), 0, border_radius=30)
        pygame.draw.rect(screen, ((107,0,139)), pygame.Rect(180, 180, 340, 440), 0, border_radius=20)
        for line in {" Congrats!":200," You Have":350,"    Won!":440}.items():
            screen.blit(font_11.render(line[0], False, ((0,0,0))),(180,line[1]))
        pygame.display.update()
    leaderboard()
    main()

# Displays Losing Message
def losing_message():
    for i in range(3000):
        pygame.draw.rect(screen, ((0,0,0)), pygame.Rect(150, 150, 400, 500), 0, border_radius=30)
        pygame.draw.rect(screen, ((107,0,139)), pygame.Rect(180, 180, 340, 440), 0, border_radius=20)
        for line in {"      You":240,"     Lost!":440}.items():
            screen.blit(font_11.render(line[0], False, ((0,0,0))),(180,line[1]))
        pygame.display.update()
    main()

# Displays Choices of In Progress Games With Entered Username to choose to play
def game_choices(game_codes):
    count = 0
    while True:
        try:
            play_game(game_codes[count], True)
        except:
```

```
            count = 0
            play_game(game_codes[count], True)
        pygame.draw.rect(screen, (139,0,0), pygame.Rect(500, 15, 190, 780), 0, border_radius=8)
        word_count = 0
        for word in ["  Multiple","   active","games exist"," under this"," username"," Use left &", "right arrow", "   keys to","     move", "  between","     the", " different","   games","Press
enter","  to select"," this game"]:
            if word[1] == "U" or word[0] == "P":
                word_count += 2
            screen.blit(font_2.render(word,False, (160,160,160)),(500,20+word_count*38))
            word_count += 1
        pygame.display.update()
        done = False
        while not done:
            for event in pygame.event.get():
                if event.type == pygame.QUIT:
                    sys.exit(1)
                elif event.type == pygame.KEYDOWN:
                    if event.key == pygame.K_LEFT:
                        count -= 1
                        done = True
                    elif event.key == pygame.K_RIGHT:
                        count += 1
                        done = True
                    elif event.key == pygame.K_RETURN:
                        play_game(game_codes[count], False)
            pygame.display.update()

# Creates Code To Crack
def code_to_crack():
    code = ""
    tempp = random.choice(list(colour_code.values()))
    temp = [str(tempp[x]) for x in range(3)]
    code += str(temp[0])
    for i in range(1,3):
        code += "-"+str(temp[i])
    for count in range(3):
        tempp = random.choice(list(colour_code.values()))
        temp = [str(tempp[x]) for x in range(3)]
        for i in range(3):
            code += "-"+str(temp[i])
    return code

# Fonts
pygame.font.init()
font_1 = pygame.font.SysFont('Comic Sans MS', 100)
font_2 = pygame.font.SysFont('Comic Sans MS', 35)
font_3 = pygame.font.SysFont('Comic Sans MS', 30)
font_4 = pygame.font.SysFont('Comic Sans MS', 40)
font_5 = pygame.font.SysFont('Arial', 30)
font_6 = pygame.font.SysFont('Comic Sans MS', 45)
font_7 = pygame.font.SysFont('Arial', 82)
font_8 = pygame.font.SysFont('Arial', 33)
font_9 = pygame.font.SysFont('Arial', 28)
font_10 = pygame.font.SysFont('Arial', 35)
font_11 = pygame.font.SysFont('Comic Sans MS', 65)
font_12 = pygame.font.SysFont('Arial', 25)

# Colour Names and their Codes
name_to_colour_code = {"black":"bk","blue":"bl","teal":"t","red":"r","yellow":"y","white":"w"}

alphabet = ['a','A','b','B','c','C','d','D','e','E','f','F','g','G','h','H','i','I','j','J','k','K','l','L','m','M','n','N','o','O', 'p', 'P', 'q', 'Q', 'r', 'R', 's', 'S', 't', 'T', 'u', 'U',
'v', 'V', 'w', 'W', 'x', 'X', 'y', 'Y', 'z', 'Z']

# Dictionary of Colour Codes and their RGB Values
colour_code = {"bk":(0,0,0),"t":(0,255,255),"bl":(0,0,255),"r":(255,0,0),"y":(255,255,0),"w":(255,255,255)}


# Translates Recoreded Code To Actual Code
def real_code(old_code):
    new_code = [(int(old_code.split("-")[i]),int(old_code.split("-")[i+1]), int(old_code.split("-")[i+2])) for  i in range(0,len(old_code.split("-"))-2,3)]
    return new_code

# Sets Up New Game
def new_game():
    passed = 0
    message = "Enter a username"
    while passed != 2:
        username = input_screen(message)
```

```
        if username in get_usernames():
            message = "Username already used in a game"
            passed += 1
        else:
            passed = 2
    game_code = get_new_game_code()
    for item in {game_code:True,username:False,code_to_crack():False,"InProgress":False}.items():
        append_to_csv(item[0],item[1])
    play_game(game_code, False)
    return ""

# Sets Up a Previous Game
def previous_game():
    passed = False
    usernames = get_usernames()
    message = "Enter username used during game"
    while not passed:
        attempted_username = input_screen(message)
        if attempted_username in usernames:
            passed = True
        message = "No in progress games exist under this username,\ntry again"
    game_code = get_usernames_game_code(attempted_username)
    if len(game_code) == 1:
        play_game(game_code[0], False)
    else:
        game_choices(game_code)

# Setup
pygame.display.set_caption("Mastermind")
screen = pygame.display.set_mode((700,800))
clock = pygame.time.Clock()

# Main Loop ()
def main():
    while True:
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                sys.exit(1)
            elif event.type == pygame.KEYDOWN:
                if event.key == pygame.K_n:
                    new_game()
                elif event.key == pygame.K_p:
                    previous_game()
                elif event.key == pygame.K_s:
                    settings()
                elif event.key == pygame.K_l:
                    leaderboard()
                elif event.key == pygame.K_h:
                    help()
                elif event.key == pygame.K_ESCAPE:
                    sys.exit(1)
        start_menu()
        pygame.display.update()

main()
```

csv_operations.py

```
from tempfile import NamedTemporaryFile
import csv
import shutil
import pandas as pd

# Appends to CSV
def append_to_csv(text, new_line_or_not):
    with open('Player_Information.csv','a') as f:
        if new_line_or_not == True:
            f.write("\n"+str(text)+",")
        else:
```

```
            f.write(str(text)+",")

# Updates CSV
def update_csv(game_code,text):
    filename = 'Player_Information.csv'
    tempfile = NamedTemporaryFile(mode='w', delete=False, newline='')
    fields =
['GameCode','Username','Code','Score','a1','b1','c1','d1','a2','b2','c2','d2','a3','b3','c3','d3','a4','b4','c4','d4','a5','b5','c5','d5','a6','b6','c6','d6','a7','b7','c7','d7'
,'a8','b8','c8','d8','a9','b9','c9','d9','a10','b10','c10','d10']
    with open(filename, 'r', newline="") as csvfile, tempfile:
        reader = csv.DictReader(csvfile, fieldnames=fields)
        writer = csv.DictWriter(tempfile, fieldnames=fields)
        passed = False
        for row in reader:
            if row['GameCode'] == str(game_code):
                count = 0
                for value in row.values():
                    if value == "" and passed == False:
                        field = fields[count]
                        row[field] = text
                        passed = True
                    count += 1
            writer.writerow(row)
    shutil.move(tempfile.name, filename)

# Gets top scores
def get_high_scores(number_of_scores):
    with open("Player_Information.csv","r") as file:
        reader = csv.reader(file, delimiter=",")
        next(reader)
        scores = []
        for row in reader:
            if row[3] not in ["Lost","InProgress"]:
                scores.append((row[1],int(row[3])))
    scores.sort(key=lambda x: x[1])
    return scores

# Gets Usernames
def get_usernames():
    with open("Player_Information.csv","r") as file:
        reader = csv.reader(file, delimiter=",")
        next(reader)
        usernames = []
        for row in reader:
            usernames.append(row[1])
    return usernames

# Get Usernames Game Code
def get_usernames_game_code(username):
    with open("Player_Information.csv","r") as file:
        reader = csv.reader(file, delimiter=",")
        next(reader)
```

```
        game_code = []
        for row in reader:
            if row[1] == username and row[3] == "InProgress":
                game_code.append(row[0])
        return game_code

# Create New Unique Game Code
def get_new_game_code():
    with open("Player_Information.csv","r") as file:
        reader = csv.reader(file, delimiter=",")
        next(reader)
        game_codes = []
        for row in reader:
            game_codes.append(int(row[0]))
    return max(game_codes) + 1

# Gets past game choices
def get_game_info(game_code):
    with open("Player_Information.csv","r") as file:
        reader = csv.reader(file, delimiter=",")
        next(reader)
        info = []
        for row in reader:
            if row[0] == str(game_code):
                for value in row[2:]:
                    if value != '':
                        info.append(value)
                return list(info)

# Updates Status In CSV
def update_status(status, game_code):
    filename = 'Player_Information.csv'
    tempfile = NamedTemporaryFile(mode='w', delete=False, newline='')
    fields = ['GameCode', 'Username', 'Code', 'Score', 'a1', 'b1', 'c1', 'd1', 'a2', 'b2', 'c2', 'd2', 'a3', 'b3', 'c3', 'd3', 'a4', 'b4', 'c4', 'd4', 'a5', 'b5', 'c5', 'd5',
'a6', 'b6', 'c6', 'd6', 'a7', 'b7', 'c7', 'd7', 'a8', 'b8', 'c8', 'd8', 'a9', 'b9', 'c9', 'd9', 'a10', 'b10', 'c10', 'd10']
    with open(filename, 'r', newline="") as csvfile, tempfile:
        reader = csv.DictReader(csvfile, fieldnames=fields)
        writer = csv.DictWriter(tempfile, fieldnames=fields)
        for row in reader:
            if row['GameCode'] == str(game_code):
                row['Score'] = status
            writer.writerow(row)
    shutil.move(tempfile.name, filename)


# Removes Empty Rows in CSV
def clean_csv():
    df = pd.read_csv('Player_Information.csv')
    df.to_csv('Player_Information.csv', index=False)
```

Player_Information.csv

```
GameCode,Username,Code,Score,a1,b1,c1,d1,a2,b2,c2,d2,a3,b3,c3,d3,a4,b4,c4,d4,a5,b5,c5,d5,a6,b6,c6,d6,a7,b7,c7,d7,a8,b8,c8,d8,a9,b9,c9,d9,a10,b10,c10,d10
1,Bob,255-255-255-0-0-255-255-0-255-0-255-255,InProgress,w,bk,y,y,r,r,w,w,w,w,w,w,w,w,w,bl,bl,,,,,,,,,,,,,,,,,,,,,,,
2,John,255-255-255-0-0-255-255-0-255-0-255-255,InProgress,w,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
3,James,255-255-255-0-0-255-255-0-255-0-255-255,InProgress,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
4,Bob,0-0-255-255-0-0-0-0-0-255-0-0,InProgress,w,w,y,y,r,r,r,bk,y,r,r,bl,bl,r,r,,,,,,,,,,,,,,,,,,,,,,,,,
5,Bob,255-0-0-255-0-0-0-255-255-0-0-0,4,w,w,bk,bk,r,r,y,w,r,r,bk,w,r,r,t,bk,,,,,,,,,,,,,,,,,,,,,,,,
6,James,0-255-255-255-255-0-255-255-255-255-0-0,8,r,r,r,r,y,y,y,y,bk,bk,bk,bk,w,w,w,w,t,t,t,t,r,y,w,t,r,y,t,w,t,y,w,r,,,,,,,,
7,Noah,255-255-255-255-255-255-0-0-255-0-0-0,1,w,w,bl,bk,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,
8,Ben,0-0-0-0-0-255-0-0-255-255-0-0,7,w,w,w,w,r,r,r,r,y,y,r,y,t,t,r,t,bl,bl,r,bk,bl,bl,bl,r,bk,bl,bl,r,,,,,,,,,,,,
```

Instructions.txt

```
How To Play&

A 4 colour long code is created by chosing one colour
out of six (Black, Blue, Red, Teal, Yellow, White) for
each of 4 positions in a specific order
The aim of the game is to guess this unknown code
You have 10 attempts, after each attempt you find out
how accurate your guess was
For each correct colour in the correct position
a black dot appears, For each correct colour in
the wrong position a red dot appears

Controls&

Once the game loads you will have a box displaying any
keyboard inputs,
You will now go through the table (starting at 1),
For each box place type the name of a colour
(or its abbreviation) and press ENTER to enter it,
To Exit, type Exit or exit
To display the help menu, type Help or help
To display the settings menu, type sett or Sett

Exit [ESC]&
```