APCSP Program Code

```python

# Images and media used in game:
# [1] Galaxy Background Image - https://www.pinterest.com.mx/pin/343610646540535196/?nic_v1=1aTdT4hS%2BVmBzrzbvEZJspx9ySJq2DVjssULmih%2BMD%2B3%2FhH%2BLGz47w5tmvRGfSh6wj
# [2] Fireball Sprites - https://www.seekpng.com/ipng/u2q8a9u2i1r5u2r5_meteor-design/ (Edited the colors of the sprite)
# [3] Bullet Sound Effect - https://www.zapsplat.com/music/retro-sci-fi-laser-beam-version-2/
# [4] Spaceship Sprite - https://www.seekpng.com/ipng/u2q8q8q8q8y3y3e6_preview-sprite-spaceship-png/ (Edited the colors of the sprite)

# Libraries:
import random
import time
import pygame
pygame.init()

# Game Window:
gameWindowWidth = 500
gameWindowHeight = 500
gameWindow = pygame.display.set_mode((gameWindowWidth, gameWindowHeight))
background = pygame.image.load("galaxy.jpg")
pygame.display.set_caption("Game Window")

# Classes:
class textClass():
    def __init__(self, txt, color, size):
        self.text = txt
        self.textColor = color
        self.textSize = size
        self.textFont = "freesansbold.ttf"
    def textDisplay(self):
        font = pygame.font.Font(self.textFont, self.textSize)
        displayText = font.render(self.text, True, self.textColor)
        return displayText

class spaceShip():
    def __init__(self, x, y):
        self.spaceShipx = x
        self.spaceShipy = y
        self.spaceShipWidth = 64
        self.spaceShipHeight = 64
        self.spaceShipSpeed = 10
        self.spaceShipSprite = pygame.image.load("spaceship.png")
    def spaceshipLeft(self):
        self.spaceShipx -= self.spaceShipSpeed
    def spaceshipRight(self):
        self.spaceShipx += self.spaceShipSpeed
    def spaceshipUp(self):
        self.spaceShipy -= self.spaceShipSpeed
    def spaceshipDown(self):
        self.spaceShipy += self.spaceShipSpeed
    def drawSpaceShip(self):
        gameWindow.blit(self.spaceShipSprite, (self.spaceShipx, self.spaceShipy))

class bulletClass():
    def __init__(self, x, y):
        self.bulletx = x
        self.bullety = y
        self.bulletWidth = 3
        self.bulletHeight = 7
        self.bulletSpeed = 8
        self.bulletColor = (255, 0, 0)
    def bulletCollision(self, list):
        list.pop(list.index(self))
    def moveBullet(self):
        self.bullety -= self.bulletSpeed
    def drawBullet(self):
        pygame.draw.rect(gameWindow, self.bulletColor, (self.bulletx, self.bullety, self.bulletWidth, self.bulletHeight))

class fireballClass():
    def __init__(self, x, y):
        self.fireballx = x
        self.firebally = y
        self.fireballWidth = 29
        self.fireballHeight = 65
        self.fireballSpeed = 8
        self.fireballRandNum = random.randint(0, 1)
        self.fireballPlus = False
        self.fireballMinus = False
        self.fireballSprite = pygame.image.load("fireball.png")
    def fireballCollision(self, list):
        list.pop(list.index(self))
    def moveFireball(self):
        self.firebally += self.fireballSpeed
    def drawFireball(self):
        gameWindow.blit(self.fireballSprite, (self.fireballx, self.firebally))

# Score Tracking:
def scoreListMethod(list, num):
    list.append(str(num) + " ")

def scoreFileWriteMethod(list):
    scoreFileWrite = open("playerScore.txt", "a")
    scoreFileWrite.writelines(list)

def listSplitMethod(list):
    return list[0].split()

def listConvertNum(list):
    for x in range(0, len(list)):
        list[x] = int(list[x])

def scoringMethod():
    scoreFileRead = open("playerScore.txt", "r")
    scoreFileText = scoreFileRead.read()
    scoreFileList = []
    scoreFileList.append(scoreFileText)
    scoreFileList2 = listSplitMethod(scoreFileList)
    listConvertNum(scoreFileList2)
    highScore = max(scoreFileList2)
    if len(scoreFileList2) == 0:
        highScore = 0
    return highScore

# Music and Sound Effects:
def playBulletSoundEffect():
    pygame.mixer.music.load("bulletsoundeffect.wav")
    pygame.mixer.music.play()

# Game Over:
red = (255, 0, 0)
gameOver = textClass("Game Over!", red, 50)
gameOverTextDisplay = gameOver.textDisplay()
gameOverRect = gameOverTextDisplay.get_rect()
gameOverRect.center = ((gameWindowWidth//2), (gameWindowHeight//2))
def gameOverMethod():
    gameWindow.blit(gameOverTextDisplay, gameOverRect)
    pygame.display.update()
    time.sleep(2)
    introLoop()

# New Record:
newRecord = textClass("New Record!", red, 50)
newRecordTextDisplay = newRecord.textDisplay()
newRecordRect = newRecordTextDisplay.get_rect()
newRecordRect.center = ((gameWindowWidth//2), (gameWindowHeight))
def newRecordMethod():
    gameWindow.blit(newRecordTextDisplay, gameOverRect)
    pygame.display.update()
    time.sleep(2)
    introLoop()

# Collisions:
def bulletCollision(x1, y1, width1, height1, x2, y2, width2, height2):
    if x2 <= x1 <= x2 + width2 and y2 <= y1 <= y2 + height2:
        return True
    elif x2 <= x1 + width1 <= x2 + width2 and y2 <= y1 <= y2 + height2:
        return True
    else:
        return False

def spaceshipCollision(x1, y1, width1, height1, x2, y2, width2, height2):
    if x2 <= x1 <= x2 + width2 and y2 <= y1 <= y2 + height2:
        return True
    elif x2 <= x1 + width1 <= x2 + width2 and y2 <= y1 <= y2 + height2:
        return True
    elif x2 <= x1 + width1 <= x2 + width2 and y2 <= y1 + height1 <= y2 + height2:
        return True
    elif x2 <= x1 + width1 <= x2 + width2 and y2 <= y1 + height1 <= y2 + height2:
        return True
    elif x1 <= x2 <= x1 + width1 and x1 <= x2 + width2 <= x1 + width1:
        if y1 <= y2 + height2 <= y1 + height1 and y1 <= y2 + height2 <= y1 + height1:
            return True
    else:
        return False

# Key Presses:
def keyPresses(obj):
    keyPressed = pygame.key.get_pressed()
    if keyPressed[pygame.K_LEFT] == True and obj.spaceShipx >= 0:
        obj.spaceshipLeft()
    elif keyPressed[pygame.K_RIGHT] == True and obj.spaceShipx <= gameWindowWidth - obj.spaceShipWidth:
        obj.spaceshipRight()
    elif keyPressed[pygame.K_UP] == True and obj.spaceShipy >= 0:
        obj.spaceshipUp()
    elif keyPressed[pygame.K_DOWN] == True and obj.spaceShipy <= gameWindowHeight - obj.spaceShipHeight:
        obj.spaceshipDown()

def moveBullets(list, obj):
    keyPressed = pygame.key.get_pressed()
    if keyPressed[pygame.K_SPACE] == True:
        playBulletSoundEffect()
        if len(list) < 25:
            list.append(bulletClass(round(obj.spaceShipx + obj.spaceShipWidth // 2), obj.spaceShipy))

# IntroLoop Method:
def introPlayedMethod():
    played = open("playerScore.txt", "r")
    playedData = played.read()
    playedList = []
    playedList.append(playedData)
    playedList2 = listSplitMethod(playedList)
    return len(playedList2)


# Game:
def gameLoop():
    # Objects:
    spaceship = spaceShip(215, 400)
    bulletList = []
    fireballList = []
    score = 0
    scoreList = []
    gameRunning = True
    while gameRunning == True:
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                gameRunning = False
        playerHighScore = scoringMethod()
        # Bullets:
        for bullet in bulletList:
            if bullet.bullety < gameWindowHeight and bullet.bullety > 0:
                bullet.moveBullet()
            else:
                bulletList.pop(bulletList.index(bullet))

        # Fireballs:
        for fireball in fireballList:
            if fireball.firebally < gameWindowHeight and fireball.firebally > 0:
                fireball.moveFireball()
            else:
                fireballList.pop(fireballList.index(fireball))
            if fireball.fireballRandNum == 1:
                fireball.fireballSprite = pygame.image.load('fireball2.png')
                fireball.fireballPlus = False
            else:
                fireball.fireballSprite = pygame.image.load('fireball.png')
                fireball.fireballPlus = True
        if len(fireballList) < 4:
            fireballList.append(fireballClass(random.randrange((spaceship.spaceShipWidth // 2), (gameWindowWidth - (spaceship.spaceShipWidth))), 1))

        moveBullets(bulletList, spaceship)
        keyPresses(spaceship)

        # Collisions and Crashing:
        for bullet in bulletList:
            for fireball in fireballList:
                collision = bulletCollision(bullet.bulletx, bullet.bullety, bullet.bulletWidth, bullet.bulletHeight, fireball.fireballx, fireball.firebally, fireball.fireballWidth, fireball.fireballHeight)
                if collision == True:
                    fireball.fireballCollision(fireballList)
                    bullet.bulletCollision(bulletList)
                    if fireball.fireballPlus == True:
                        score += 1
                    else:
                        score -= 1

        for fireball in fireballList:
            crash = spaceshipCollision(spaceship.spaceShipx, spaceship.spaceShipy, spaceship.spaceShipWidth, spaceship.spaceShipHeight, fireball.fireballx, fireball.firebally, fireball.fireballWidth, fireball.fireballHeight)
            if crash == True:
                scoreListMethod(scoreList, score)
                scoreFileWriteMethod(scoreList)
                if score > playerHighScore:
                    newRecordMethod()
                else:
                    gameOverMethod()

        # FPS:
        gameClock = pygame.time.Clock()
        gameClock.tick(30)

        # Score Text:
        scoreText = textClass("Score: " + str(score), red, 15)
        scoreDisplayText = scoreText.textDisplay()
        highScoreText = textClass("High Score: " + str(playerHighScore), red, 15)
        highScoreDisplayText = highScoreText.textDisplay()

        gameWindow.blit(background, (0, 0))
        spaceship.drawSpaceShip()
        for bullet in bulletList:
            bullet.drawBullet()
        for fireball in fireballList:
            fireball.drawFireball()
        gameWindow.blit(scoreDisplayText, (430, 20))
        gameWindow.blit(highScoreDisplayText, (392, 35))
        pygame.display.update()
    pygame.quit()


def introLoop():
    # Main Loop
    gameRunning = True
    while gameRunning == True:
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                gameRunning = False

        # Avoid crashing into fireballs.
        red = (255, 0, 0)
        rule1Text = textClass("Avoid crashing into fireballs.", red, 25)
        rule1DisplayText = rule1Text.textDisplay()

        # Arrow keys to move. Space to shoot.
        rule2Text = textClass("Arrow keys to move. Space to shoot.", red, 25)
        rule2DisplayText = rule2Text.textDisplay()

        # Orange fireballs increase your score.
        orange = (255, 162, 0)
        rule3Text = textClass("Orange fireballs increase your score.", orange, 25)
        rule3DisplayText = rule3Text.textDisplay()

        # Blue fireballs decrease your score.
        blue = (0, 81, 255)
        rule4Text = textClass("Blue fireballs decrease your score.", blue, 25)
        rule4DisplayText = rule4Text.textDisplay()

        # This game has been played ___ times.
        playedLen = introPlayedMethod()
        playedText = textClass("This game has been played " + str(playedLen) + " time(s).", red, 25)
        playedDisplayText = playedText.textDisplay()

        # Press Q to continue.
        pressQText = textClass("Press Q to continue to game.", red, 25)
        pressQDisplayText = pressQText.textDisplay()

        # Key Presses
        keyPressed = pygame.key.get_pressed()
        if keyPressed[pygame.K_q] == True:
            gameLoop()

        black = (0, 0, 0)
        gameWindow.fill(black)
        gameWindow.blit(rule1DisplayText, (75, 50))
        gameWindow.blit(rule2DisplayText, (30, 85))
        gameWindow.blit(rule3DisplayText, (22, 120))
        gameWindow.blit(rule4DisplayText, (37, 155))
        gameWindow.blit(playedDisplayText, (9, 190))
        gameWindow.blit(pressQDisplayText, (73, 225))
        pygame.display.update()
    pygame.quit()
introLoop()
```

PDF document made with CodePrint using Prism