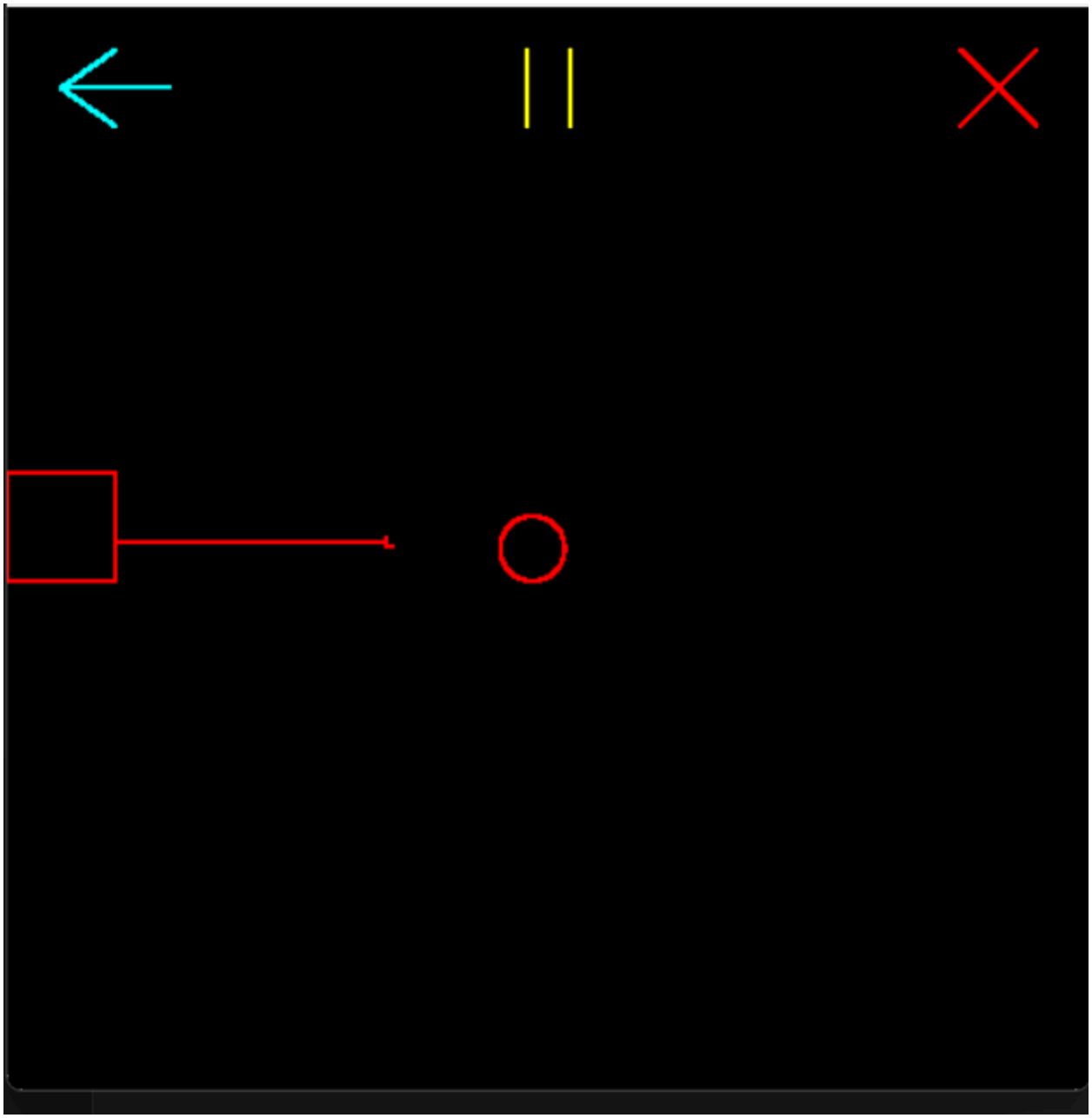
Balloon Shooter Game

Group-6

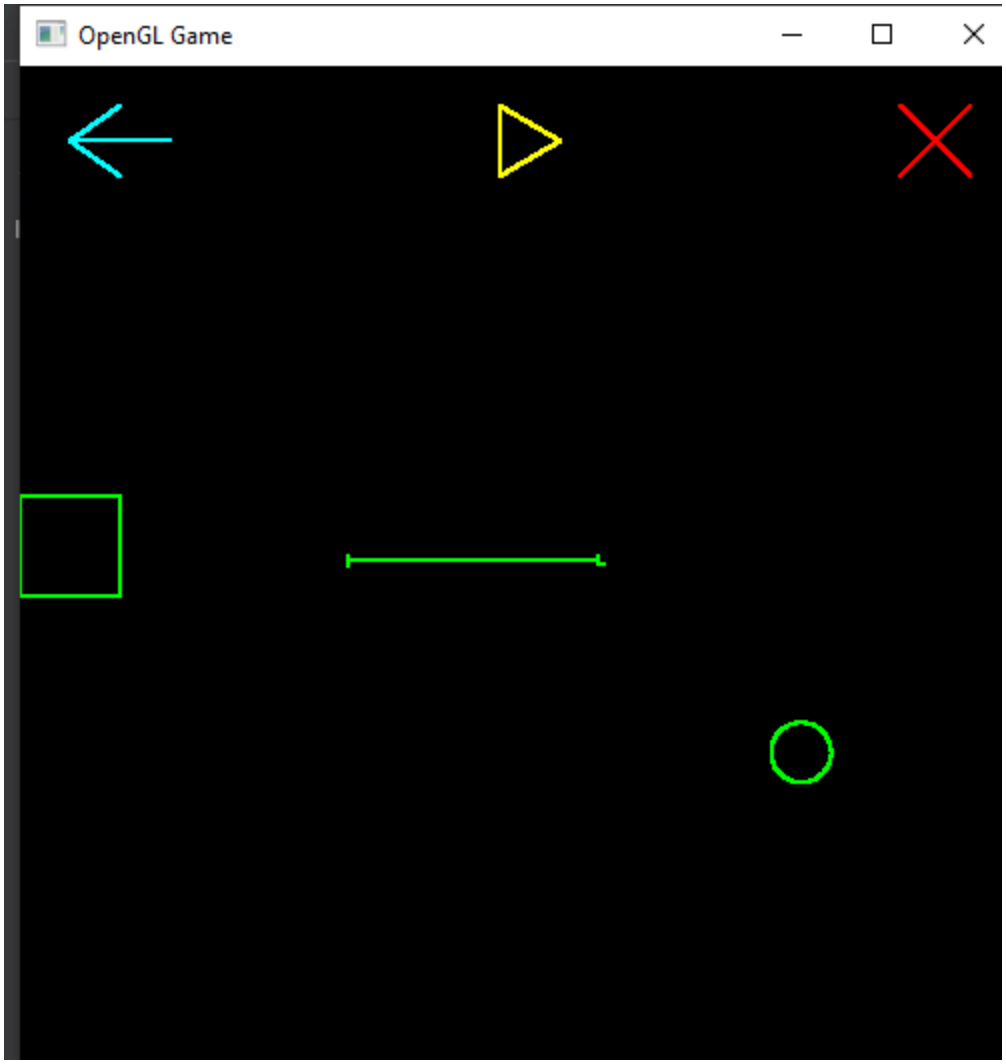
A) Game Description & Feature

Basically this is a balloon shooter game. You can shoot the balloon with the canon. You can move the canon. Whenever the arrow touches the balloon you will get a point. If you miss 3 balloons, the game will be over. You can restart the game with the arrow button. You can pause the game and you can end the game as well. The balloon's speed will increase every time you shoot them.

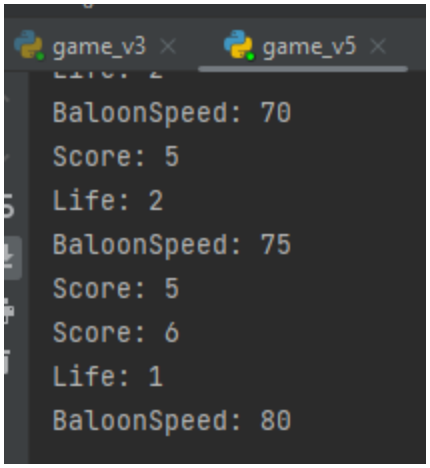
B) Screenshot of Gameplay



1) **The Blue coloured Left directed arrows allows you to restart the game, Score will be set to 0 and Life to 3 by default.Yellow coloured two lines will allow you to pause the game.**



2) **The game can be resumed again by clicking on the Yellow color play button.**



3) **After hitting the balloons, scores are being calculated, After Each score, Balloon speed increases by 5. If the Shooter misses the arrow, Life is deducted. Total of 3 life a player gets-**

Contributions:

23241114	**Nafisa Mehreen**	**Implementing balloon Pause Play button using midpoint circle and midpoint line algorithm, Animate function of the Balloon,Show-Screen Function.**
21301304	**Abtahi Noor**	**Implementing the Restart Button using midpoint line algorithm, Collision in the Animate function,Score Calculation, Game Over Button.**
21301024	**Abrar Rahman**	**Implementing the Canon and Arrow using the midpoint line algorithm,Arrow Movement Control.**

Source Code:

```
from OpenGL.GL import *
from OpenGL.GLUT import *
from OpenGL.GLU import *
from math import cos, sin, pi
import random
life=3

canon_x=0
canon_y=235
autoMoveSpeed = 10  # Speed of automatic movement
movingRight = True
x0, y0, x1, y1 = 0, 5000, 2500, 5350  # For Catcher

# New global variables for vertical movement
goUp = 0
goDown = 0

a0, b0, a1, b1 = 4700, 0, 5000, 0  # For Baloon
goLeft = 0
goRight = 0
BaloonSpeed = 50
BaloonColors = [(1.0, 0.0, 0.0), (0.0, 1.0, 0.0), (0.0, 0.0, 1.0)]
colorIndex = 0
pauseBoolean = False
onlyOnce = True
score = 0
gameOver = False



def circle(a0, b0, a1, b1):
        global colorIndex
        color = BaloonColors[colorIndex]
        glColor3f(*color)

        # Adjusting the position and dimensions of the circle
        center_x = (a0 + a1) / 2
        center_y = (b0 + b1) / 2
```

```
        radius = 2 * (a1 - a0) / 2

        # Midpoint circle drawing algorithm
        x = 0
        y = radius
        d = 1 - radius

        points = []

        while x <= y:
            points.append((x, y))

            if d < 0:
                d = d + 2 * x + 3
            else:
                d = d + 2 * (x - y) + 5
                y -= 1

            x += 1

        # Draw the points in all octants
        for point in points:
            x, y = point
            glBegin(GL_POINTS)
            glVertex2f(center_x + x, center_y + y)
            glVertex2f(center_x - x, center_y + y)
            glVertex2f(center_x + x, center_y - y)
            glVertex2f(center_x - x, center_y - y)
            glVertex2f(center_x + y, center_y + x)
            glVertex2f(center_x - y, center_y + x)
            glVertex2f(center_x + y, center_y - x)
            glVertex2f(center_x - y, center_y - x)
            glEnd()


def arrowLeft():
    glColor3f(0.0, 1.0, 1.0)
    midpoint(500, 9250, 1500, 9250)
    midpoint(500, 9250, 1000, 9600)
    midpoint(500, 9250, 1000, 8900)
```

```
def boxArrow():
    glColor3f(0.0, 0.0, 0.0)
    glBegin(GL_QUADS)
    glVertex2d(0, 8500)
    glColor3f(0.0, 0.0, 0.0)
    glVertex2d(2000, 8500)
    glColor3f(0.0, 0.0, 0.0)
    glVertex2d(2000, 10000)
    glColor3f(0.0, 0.0, 0.0)
    glVertex2d(0, 10000)
    glEnd()


def restartGame():
    global a0, b0, a1, b1,life, BaloonSpeed
    a0, b0, a1, b1 = 4700, 0, 5000, 0
    life=3
    BaloonSpeed = 50


def twoLines():
    glColor3f(1.0, 1.0, 0.0)
    midpoint(4800, 9600, 4800, 8900)
    midpoint(5200, 9600, 5200, 8900)


def twoLinesP():
    glColor3f(1.0, 1.0, 0.0)
    midpoint(4800, 9600, 4800, 8900)
    midpoint(4800, 8900, 5400, 9250)
    midpoint(4800, 9600, 5400, 9250)


def boxtwoLines():
    glColor3f(0.0, 0.0, 0.0)
    glBegin(GL_QUADS)
    glVertex2d(4000, 8500)
    glColor3f(0.0, 0.0, 0.0)
```

```
    glVertex2d(6000, 8500)
    glColor3f(0.0, 0.0, 0.0)
    glVertex2d(6000, 10000)
    glColor3f(0.0, 0.0, 0.0)
    glVertex2d(4000, 10000)
    glEnd()


def pauseGame():
    global pauseBoolean
    if pauseBoolean == False:
        pauseBoolean = True
    else:
        pauseBoolean = False


def cross():
    glColor3f(1.0, 0.0, 0.0)
    midpoint(8800, 9600, 9500, 8900)
    midpoint(8800, 8900, 9500, 9600)


def boxCross():
    glColor3f(0.0, 0.0, 0.0)
    glBegin(GL_QUADS)
    glVertex2d(8300, 8500)
    glColor3f(0.0, 0.0, 0.0)
    glVertex2d(10000, 8500)
    glColor3f(0.0, 0.0, 0.0)
    glVertex2d(10000, 10000)
    glColor3f(0.0, 0.0, 0.0)
    glVertex2d(8300, 10000)
    glEnd()


def draw_points(x0, y0):
    glPointSize(2)
    glBegin(GL_POINTS)
    glVertex2f(x0, y0)
    glEnd()
```

```python
def midpoint(x0, y0, x1, y1):
    zone = findZone(x0, y0, x1, y1)
    x0, y0 = zoneConvert0(zone, x0, y0)
    x1, y1 = zoneConvert0(zone, x1, y1)
    dx = x1 - x0
    dy = y1 - y0
    dinit = 2 * dy - dx
    dne = 2 * dy - 2 * dx
    de = 2 * dy
    for i in range(x0, x1):
        a, b = convert0_Original(zone, x0, y0)
        if dinit >= 0:
            dinit = dinit + dne
            draw_points(a, b)
            x0 += 1
            y0 += 1
        else:
            dinit = dinit + de
            draw_points(a, b)
            x0 += 1


def findZone(x0, y0, x1, y1):
    dx = x1 - x0
    dy = y1 - y0
    if abs(dx) > abs(dy):  # For Zone 0, 3, 4 & 7
        if dx > 0 and dy > 0:
            return 0
        elif dx < 0 and dy > 0:
            return 3
        elif dx < 0 and dy < 0:
            return 4
        else:
            return 7
    else:  # For zone 1, 2, 5 & 6
        if dx > 0 and dy > 0:
            return 1
        elif dx < 0 and dy > 0:
```

```
            return 2
        elif dx < 0 and dy < 0:
            return 5
        else:
            return 6


def zoneConvert0(zone, x0, y0):
    if zone == 0:
        return x0, y0
    elif zone == 1:
        return y0, x0
    elif zone == 2:
        return -y0, x0
    elif zone == 3:
        return -x0, y0
    elif zone == 4:
        return -x0, -y0
    elif zone == 5:
        return -y0, -x0
    elif zone == 6:
        return -y0, x0
    elif zone == 7:
        return x0, -y0


def convert0_Original(zone, x0, y0):
    if zone == 0:
        return x0, y0
    if zone == 1:
        return y0, x0
    if zone == 2:
        return -y0, -x0
    if zone == 3:
        return -x0, y0
    if zone == 4:
        return -x0, -y0
    if zone == 5:
        return -y0, -x0
    if zone == 6:
```

```
        return y0, -x0
    if zone == 7:
        return x0, -y0

def specialKeyListener(key, left, right):
    glutPostRedisplay()
    global canon_y, goLeft, goRight, goUp, goDown, x0, x1, autoMoveSpeed

    if not pauseBoolean:
        move_step = 20
        if key == GLUT_KEY_LEFT:
                autoMoveSpeed -= 50
        elif key == GLUT_KEY_RIGHT:
            autoMoveSpeed += 50
        elif key == GLUT_KEY_UP:
            goUp += 410
            canon_y += move_step
            # Move up
        elif key == GLUT_KEY_DOWN:
            goDown += 410
            canon_y -= move_step
            # Move down

        glutPostRedisplay()



def mouseListener(button, state, x, y):
    global ballx, bally, create_new, pauseBoolean, gameOver, score
    if button == GLUT_LEFT_BUTTON:
        if (state == GLUT_DOWN):
            if 200 <= x <= 300 and 0 <= y <= 75:
                pauseGame()
            if 0 <= x <= 100 and 0 <= y <= 75:
                gameOver = False
                score = 0
                restartGame()
            if 415 <= x <= 500 and 0 <= y <= 75:
                glutLeaveMainLoop()
```

```
def animate():
    global b0, b1, x0, x1, y0, y1, colorIndex, score,BaloonSpeed, a0,
a1,life
    global goLeft, goRight, movingRight, autoMoveSpeed, gameOver

    if not pauseBoolean and not gameOver:
        # Automatic movement of the catcher
        if movingRight:
            if x1 - goLeft + goRight < 10000:  # Assuming 10000 is the
right edge
                goRight += autoMoveSpeed
                goLeft = 0  # Reset goLeft when moving right
            else:
                movingRight = False  # Change direction
        else:
            if x0 - goLeft + goRight > 0:  # Assuming 0 is the left edge
                goLeft += autoMoveSpeed
                goRight = 0  # Reset goRight when moving left
            else:
                movingRight = True  # Change direction


        b0 += BaloonSpeed
        b1 += BaloonSpeed

        # Catcher's current position
        catcherLeft = x0 - goLeft + goRight
        catcherRight = x1 - goLeft + goRight
        catcherTop = y1 - goDown + goUp
        catcherBottom = y0 - goDown + goUp

        # Check for collision
        if catcherLeft < a1 < catcherRight and catcherBottom < b1 <
catcherTop:
            b0 = 0
            b1 = 0
            colorIndex = (colorIndex + 1) % len(BaloonColors)
            score += 1
            BaloonSpeed += 5
            print(f'Score: {score}')
```

```
            print(f'Life: {life}')
            print(f'BaloonSpeed: {BaloonSpeed}')
            a0 = random.randint(500, 9500)
            a1 = a0 + 300
        elif b1 > 7800:
            b0 = 0
            b1 = 0
            life -= 1
            print(f'Score: {score}')
            a0 = random.randint(500, 9500)
            a1 = a0 + 300
            if life <= 0:
                gameOver = True
                print("Game Over!")
                life=3

    glutPostRedisplay()

def iterate():
    glViewport(0, 0, 500, 500)
    glMatrixMode(GL_PROJECTION)
    glLoadIdentity()
    glOrtho(0.0, 10000, 0.0, 10000, 0.0, 1.0)
    glMatrixMode(GL_MODELVIEW)
    glLoadIdentity()


def showScreen():
    global x0, y0, x1, y1
    global a0, b0, a1, b1
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT)
    glLoadIdentity()
    iteratecanon()
    draw_canon()
    iterate()
    if not gameOver:
      animate()
      glColor3f(1.0, 0.0, 0.0)
      boxArrow()
      boxtwoLines()
```

```
      boxCross()
      arrowLeft()
      if pauseBoolean == False:
         twoLines()
      else:
        twoLinesP()

      #displayScore()
      #displayLife()
      cross()
      circle(a0, b0, a1, b1)
      catcher(x0, y0, x1, y1)
    else:
      # Render a game over screen
      renderText("Game Over", 4000, 5000)
      boxArrow()
      boxtwoLines()
      boxCross()
      arrowLeft()
      if pauseBoolean:
          twoLinesP()
      else:
          twoLines()
      cross()

    glutSwapBuffers()


def show_canon():
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT)
    glLoadIdentity()
    iteratecanon()

    draw_canon()

    glutSwapBuffers()

def draw_line(x1, y1, x2, y2, colour=None):
    glBegin(GL_LINES)
    if colour is not None:
```

```
        glColor3f(colour[0], colour[1], colour[2])
    glVertex2f(x1, y1)
    glVertex2f(x2, y2)
    glEnd()




def catcher(x0, y0, x1, y1):
        global gameOver, BaloonSpeed

        arrow_length = 50
        arrow_head_length = 50
        arrow_head_width = 5
        arrow_start_y = y0 - goDown + goUp

        # Draw the main body of the catcher (arrow)
        draw_midpoint_line(x0 - goLeft + goRight, arrow_start_y, x0 +
arrow_head_width - goLeft + goRight,arrow_start_y, (1, 1, 1))
        draw_midpoint_line(x0 + arrow_head_width - goLeft + goRight,
arrow_start_y, x0 + arrow_head_width - goLeft + goRight, arrow_start_y +
arrow_head_length, (1, 1, 1))
        draw_midpoint_line(x0 + arrow_head_width - goLeft + goRight,
arrow_start_y + arrow_head_length,x0 - goLeft + goRight, arrow_start_y +
arrow_length, (1, 1, 1))
        draw_midpoint_line(x0 - goLeft + goRight, arrow_start_y +
arrow_length, x0 - goLeft + goRight, arrow_start_y, (1, 1, 1))

        # Draw the head of the arrow at the right side of the catcher
        draw_midpoint_line(x1 - goLeft + goRight, arrow_start_y +
arrow_length / 2, x1 - goLeft + goRight, arrow_start_y + arrow_length / 2
+ arrow_head_width, (1, 1, 1))
        draw_midpoint_line(x1 - goLeft + goRight, arrow_start_y +
arrow_length / 2 + arrow_head_width, x1 - goLeft + goRight +
arrow_head_length,arrow_start_y + arrow_length / 2 + arrow_head_width /
2,(1, 1, 1))
```

```
        draw_midpoint_line(x0 + arrow_head_width / 2 - goLeft + goRight,
arrow_start_y + arrow_length + arrow_head_length,x0 - goLeft + goRight,
arrow_start_y + arrow_length, (1, 1, 1))
        draw_midpoint_line(x0 - goLeft + goRight, arrow_start_y +
arrow_length,x1 - goLeft + goRight, arrow_start_y + arrow_length, (1, 1,
1))
        draw_midpoint_line(x1 - goLeft + goRight, arrow_start_y +
arrow_length,x1 - arrow_head_width / 2 - goLeft + goRight,arrow_start_y +
arrow_length + arrow_head_length,(1, 1, 1))

def draw_canon():
    global canon_x, canon_y
    # Parameters for the box (cannon shape)
    width = 50  # Width of the rectangle
    height = 50  # Height of the rectangle

    # Draw the top horizontal line of the rectangle
    draw_midpoint_line(canon_x, canon_y, canon_x + width, canon_y, (1, 1,
1))

    # Draw the bottom horizontal line of the rectangle
    draw_midpoint_line(canon_x, canon_y + height, canon_x + width, canon_y
+ height, (1, 1, 1))

    # Draw the left vertical line of the rectangle
    draw_midpoint_line(canon_x, canon_y, canon_x, canon_y + height, (1, 1,
1))

    # Draw the right vertical line of the rectangle
    draw_midpoint_line(canon_x + width, canon_y, canon_x + width, canon_y
+ height, (1, 1, 1))


def draw_midpoint_line(x1, y1, x2, y2, colour=None):
    dx = x2 - x1
    dy = y2 - y1

    if abs(dx) > abs(dy):
        if x1 > x2:
            x1, y1, x2, y2 = x2, y2, x1, y1
```

```
        slope = dy / dx

        y = y1
        for x in range(int(x1), int(x2) + 1):
            draw_points(x, round(y))
            y += slope
    else:
        if y1 > y2:
            x1, y1, x2, y2 = x2, y2, x1, y1
        slope = dx / dy

        x = x1
        for y in range(int(y1), int(y2) + 1):
            draw_points(round(x), y)
            x += slope


def iteratecanon():
    glViewport(0, 0, 500, 500)
    glMatrixMode(GL_PROJECTION)
    glLoadIdentity()
    glOrtho(0.0, 500, 0.0, 500, 0.0, 1.0)
    glMatrixMode(GL_MODELVIEW)
    glLoadIdentity()

glutInit()
glutInitDisplayMode(GLUT_DEPTH | GLUT_DOUBLE | GLUT_RGB)
glutInitWindowSize(500, 500)
glutInitWindowPosition(0, 0)
wind = glutCreateWindow(b"OpenGL Game")  # window name
glutDisplayFunc(showScreen)
glutIdleFunc(showScreen)
glutSpecialFunc(specialKeyListener)
glutMouseFunc(mouseListener)
glutMainLoop()
```