Name: Md. Danial Islam
Id: 20101534
Section : 03
Lab: 02

File Name: argtaker.py

```
import argparse
parser = argparse.ArgumentParser()
parser.add_argument("--stdId", "-id", help="Student Id to show", type=str)
args = parser.parse_args()
```

File Name: windowsize.py

```
# window size, and window position on screen
windowX,windowY = 500,600
windowPosX,windowPosY = 700,300
windowName = "CSE423 - Lab 2 - Md. Danial Islam"
```

File Name: helper.py

```
from argtaker import args
from windowsize import *
from OpenGL.GL import *
from OpenGL.GLUT import *
from OpenGL.GLU import *
import random
def get_random_color():
    return random.random(), random.random(), random.random()
def getStudentId():
    if(args.stdId):
        return args.stdId
    else:
        return input("Please enter your studentId: ")
def draw(x, y):
    glPointSize(5)
    glBegin(GL_POINTS)
    glVertex2f(abs(x), abs(y))
    glEnd()
def midpoint(x0, y0, x1, y1):
    r,g,b = get_random_color()
    glColor3f(r,g,b)
    if x0 > x1:
        x0,y0,x1,y1 = x1,y1,x0,y0
```

```
    zone = find_zone(x0, y0, x1, y1)
    x0, y0 = zone_sort(x0, y0, zone)
    x1, y1 = zone_sort(x1, y1, zone)
    dy = y1 - y0
    dx = x1 - x0
    d = (2 * dy) - dx
    dE = 2 * dy
    dNE = 2 * (dy - dx)
    x = x0
    y = y0
    x_org, y_org = zone_sort(x, y, zone,False)
    draw(x_org, y_org)
    while x <= x1:
        if x == x1 and y == y1:
            break
        if d <= 0:
            x += 1
            d += dE
        else:
            x += 1
            y += 1
            d += dNE
        x_org, y_org = zone_sort(x, y, zone,False)
        draw(x_org, y_org)

def find_zone(x0, y0, x1, y1):
    dy = y1 - y0
    dx = x1 - x0
    if abs(dx) > abs(dy):
        if dx > 0 and dy > 0:
            return 0
        elif dx < 0 < dy:
            return 3
        elif dx < 0 and dy < 0:
            return 4
        else:
            return 7
    else:
        if dx > 0 and dy > 0:
            return 1
        elif dx < 0 < dy:
```

```
            return 2
        elif dx < 0 and dy < 0:
            return 5
        else:
            return 6
any_to_zone_zero = [("x","y"), ("y","x"),( "y","-x"), ("-x","y"), ("-x","-y"),
("-y","-x"), ("-y","x"), ("x","-y")]
zero_to_any_zone = [("x","y"), ("y","x"), ("-y","x"), ("-x","y"), ("-x","-y"),
("-y","-x"), ("y","-x"), ("x","y")]
def zone_sort(x, y, zone,flag=True):
    if flag:
        for i in range(8):
            if i == zone:
                xz = any_to_zone_zero[i][0]
                yz = any_to_zone_zero[i][1]
                if xz == "-x":
                    x = x * (-1)
                elif xz == "y":
                    x,y = y,x
                    return x, y
                if yz == "-y":
                    y = y * (-1)
                elif yz == "x":
                    x,y = y,x
                return x, y
    else:
        for i in range(8):
            if i == zone:
                xz = zero_to_any_zone[i][0]
                yz = zero_to_any_zone[i][1]
                if xz == "-x":
                    x = x * (-1)
                elif xz == "y":
                    x,y = y,x
                    return x, y
                if yz == "-y":
                    y = y * (-1)
                elif yz == "x":
                    x,y = y,x
                return x, y
```

File Name: number.py

```
from helper import midpoint
def draw_numbers(num, x):
    numbers = [zero, one, two, three, four, five, six, seven, eight, nine]
    x = numbers[num](x, 300)
    return x + 20

def zero(x, y):
    midpoint(x, y + 20, x, y + 40)
    midpoint(x + 20, y + 40, x, y + 40)
    midpoint(x + 20, y + 20, x + 20, y + 40)
    midpoint(x + 20, y, x + 20, y + 20)
    midpoint(x, y, x + 20, y)
    midpoint(x, y, x, y + 20)
    return x + 15
def one(x, y):
    midpoint(x + 20, y + 20, x + 20, y + 40)
    midpoint(x + 20, y, x + 20, y + 20)
    return x + 15
def two(x, y):
    midpoint(x + 20, y + 40, x, y + 40)
    midpoint(x + 20, y + 20, x + 20, y + 40)
    midpoint(x, y, x + 20, y)
    midpoint(x, y, x, y + 20)
    midpoint(x, y + 20, x + 20, y + 20)
    return x + 15
def three(x, y):
    midpoint(x + 20, y + 40, x, y + 40)
    midpoint(x + 20, y + 20, x + 20, y + 40)
    midpoint(x + 20, y, x + 20, y + 20)
    midpoint(x, y, x + 20, y)
    midpoint(x, y + 20, x + 20, y + 20)
    return x + 15
def four(x, y):
    midpoint(x, y + 20, x, y + 40)
    midpoint(x + 20, y + 20, x + 20, y + 40)
    midpoint(x + 20, y, x + 20, y + 20)
    midpoint(x, y + 20, x + 20, y + 20)
    return x + 15
def five(x, y):
    midpoint(x, y + 20, x, y + 40)
    midpoint(x + 20, y + 40, x, y + 40)
```

```
    midpoint(x + 20, y, x + 20, y + 20)
    midpoint(x, y, x + 20, y)
    midpoint(x, y + 20, x + 20, y + 20)
    return x + 15
def six(x, y):
    midpoint(x, y + 20, x, y + 40)
    midpoint(x + 20, y + 40, x, y + 40)
    midpoint(x + 20, y, x + 20, y + 20)
    midpoint(x, y, x + 20, y)
    midpoint(x, y, x, y + 20)
    midpoint(x, y + 20, x + 20, y + 20)
    return x + 15
def seven(x, y):
    midpoint(x + 20, y + 40, x, y + 40)
    midpoint(x + 20, y + 20, x + 20, y + 40)
    midpoint(x + 20, y, x + 20, y + 20)
    return x + 15
def eight(x, y):
    midpoint(x, y + 20, x, y + 40)
    midpoint(x + 20, y + 40, x, y + 40)
    midpoint(x + 20, y + 20, x + 20, y + 40)
    midpoint(x + 20, y, x + 20, y + 20)
    midpoint(x, y, x + 20, y)
    midpoint(x, y, x, y + 20)
    midpoint(x, y + 20, x + 20, y + 20)
    return x + 15
def nine(x, y):
    midpoint(x, y + 20, x, y + 40)
    midpoint(x + 20, y + 40, x, y + 40)
    midpoint(x + 20, y + 20, x + 20, y + 40)
    midpoint(x + 20, y, x + 20, y + 20)
    midpoint(x, y, x + 20, y)
    midpoint(x, y + 20, x + 20, y + 20)
    return x + 15
```

File Name: main.py

```
from helper import *
from number import draw_numbers
def iterate():
    glViewport(0, 0, windowX, windowY)
    glMatrixMode(GL_PROJECTION)
```

```
    glLoadIdentity()
    glOrtho(0.0, windowX, 0.0, windowY, 0.0, 1.0)
    glMatrixMode (GL_MODELVIEW)
    glLoadIdentity()
def showScreen():
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT)
    glLoadIdentity()
    iterate()
def showScreen(n):
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT)
    glLoadIdentity()
    iterate()
    x = windowX//2-40

    for num in n:
        x = draw_numbers(int(num), x)
    glutSwapBuffers()

    glutSwapBuffers()
n = getStudentId()
windowName = f"Student Id: {n}.Let's draw: {n[-2:]}"

glutInit()
glutInitDisplayMode(GLUT_RGBA)

# windowsize, window position and window title
glutInitWindowSize(windowX, windowY)
glutInitWindowPosition(windowPosX, windowPosY)

wind= glutCreateWindow(bytes(windowName, "utf-8"))

# display function
glutDisplayFunc(lambda: showScreen(n[-2:]))

glutMainLoop()
```

Screenshots of the running program

Task 1: