TRIBHUVAN UNIVERSITY
INSTITUTE OF SCIENCE AND TECHNOLOGY

Central Department of Computer Science and Information Technology
Kirtipur, Kathmandu
2022

Lab Report: II

“Implementation of polygon, turn test, and convexity”

Submitted By:
Sudhan Kandel
Semester: 2nd
Roll no: 2

Submitted To:
Asst.Prof. Jagdish Bhatta

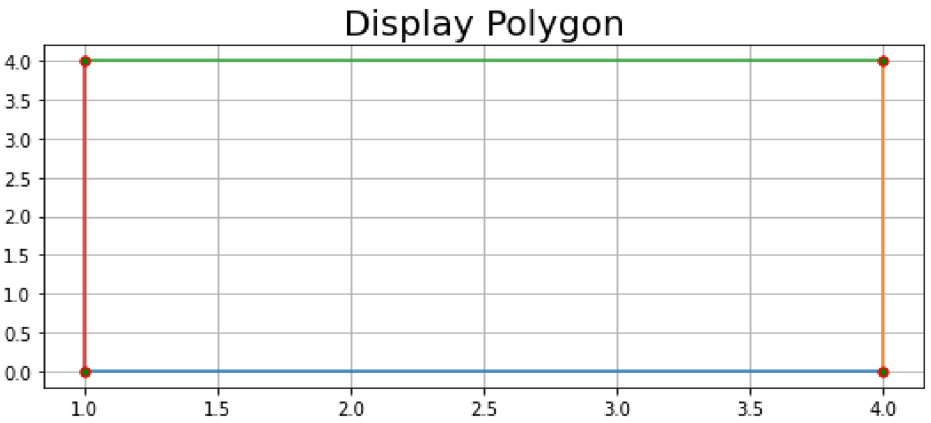
____

1. Write programs for Implementation of Polygon.

```
import matplotlib.pyplot as plt
class Node:
    def __init__(self,data):
        self.data = data;
        self.previous = None;
        self.next = None;

class DoublyLinkedList:
    def __init__(self):
        self.head = None;
        self.tail = None;
    def addNode(self, data):
        newNode = Node(data);
        if(self.head == None):
            self.head = self.tail = newNode;
            self.head.previous = None;
            self.tail.next = None;
        else:
            self.tail.next = newNode;
            newNode.previous = self.tail;
            self.tail = newNode;
            self.tail.next = None;
class Polygon(DoublyLinkedList):
    def __init__(self):
        super(Polygon, self).__init__()
        number_of_vertices=int(input("Please Enter the number of vertices"))
        for i in range(number_of_vertices):
            x,y=input("Please Enter X and Y cordinate").split(',')
            x=int(x)
            y=int(y)
            self.addNode([x,y])
    def dislpay(self):
        print("\n................................")
        print("Name: Sudhan Kandel","\nRoll No: 2","\nSection: A")
        print("\n................................")
        current=self.head
        if(self.head==None):
            print("List is empty")
            return
        while(current!=None):
            print("Vertices of the polygon is: ")
            print(current.data)
            current=current.next
    def visualization(self):
        self.dislpay()
        cur = self.head;
        while cur:
            a=cur.data
            cur=cur.next
            if cur==None:
                cur1=self.head
                while cur1:
                    b=cur1.data
                    break
            else:
                b=cur.data
            plt.plot([a[0],b[0]],[a[1],b[1]], linestyle="-", marker="o", markersize=5, markeredgecolor="red",
markerfacecolor="green")
            plt.title("Display Polygon",fontdict={'fontsize':20})
pol=Polygon()
pol.visualization()
```

OUTPUT:

```
Please Enter the number of vertices4
Please Enter X and Y cordinate1,0
Please Enter X and Y cordinate4,0
Please Enter X and Y cordinate4,4
Please Enter X and Y cordinate1,4

.................................
Name: Sudhan Kandel
Roll No: 2
Section: A

.................................
Vetrices of the polygon is:
[1, 0]
Vetrices of the polygon is:
[4, 0]
Vetrices of the polygon is:
[4, 4]
Vetrices of the polygon is:
[1, 4]
```

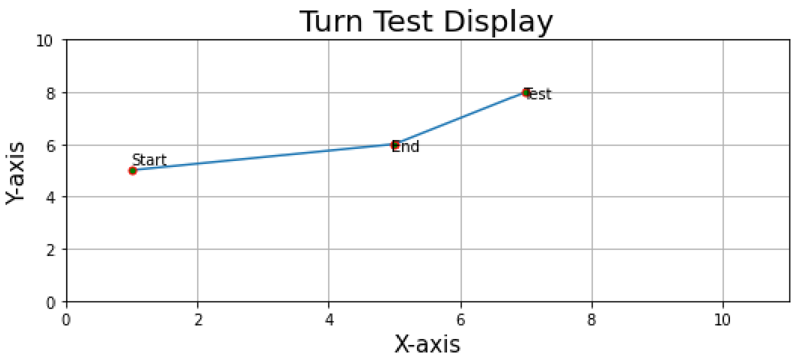

Display Polygon
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
1.0
1.5
2.0
2.5
3.0
3.5
4.0

2. Write a program for Implementation of Turn Test (Left, Right and Collinear)

```
import matplotlib.pyplot as plt
class Turntest:
    def __init__(self):
        self.points=[]
        for i in ['starting','ending','testing']:
            x,y=input("Please Enter "+i+" point's X-Cordinate and Y-cordinate....").split(",")
            x=int(x)
            y=int(y)
            self.points.append([x,y])
    def checkturn(self):
        area=(self.points[1][0]-self.points[0][0])*(self.points[2][1]-self.points[0][1])-(self.points[2][0]-
self.points[0][0])*(self.points[1][1]-self.points[0][1])
        if area>0:
            return "Left Turn"
        elif area<0:
            return "Right Turn"
        else:
            return "Colinear"
    def displayinfo(self):
        print("Starting point:", "\nX-Cordinate: ",self.points[0][0],"\nY-Cordinate: ",self.points[0][1])
        print("................................")
        print("Ending point:", "\nX-Cordinate: ",self.points[1][0],"\nY-Cordinate: ",self.points[1][1])
        print("................................")
        print("Testing point:", "\nX-Cordinate: ",self.points[2][0],"\nY-Cordinate: ",self.points[2][1])
        print("............Checking Result...................")
        print("the given test point is: ",self.checkturn())

    def visualization(self):
        plt.rcParams["figure.figsize"] = [7.50, 3.50]
        plt.rcParams["figure.autolayout"] = True
        start=self.points[0]
        end=self.points[1]
        testing=self.points[2]
        x_values = [start[0], end[0],testing[0],end[0]+4]
        y_values = [start[1], end[1],testing[1],end[1]*1.25]
        plt.grid()
        plt.xlim(0,max(x_values)+2)
        plt.ylim(0,max(y_values)+2)
        plt.plot(x_values[:-1], y_values[:-1], linestyle="-", marker="o", markersize=5, markeredgecolor="red", markerfacecolor="green")
        plt.text(start[0]-0.015, start[1]+0.25, "Start")
        plt.text(end[0]-0.050, end[1]-0.25, "End")
        plt.text(testing[0]-0.050, testing[1]-0.25, "Test")
        plt.title("Turn Test Display",fontdict={'fontsize':20})
        plt.xlabel("X-axis",fontdict={'fontsize':15})
        plt.ylabel("Y-axis",fontdict={'fontsize':15})
        plt.savefig('line.png')
        plt.show()
turn=Turntest()
turn.displayinfo()
turn.visualization()
```

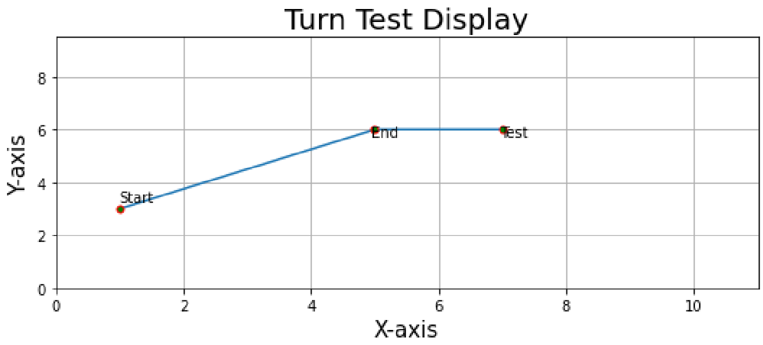
OUTPUT:

```
Please Enter starting point's X-Cordinate and Y-cordinate....1,5
Please Enter ending point's X-Cordinate and Y-cordinate....5,6
Please Enter testing point's X-Cordinate and Y-cordinate....7,8
Starting point:
X-Cordinate:  1
Y-Cordinate:  5
.................................
Ending point:
X-Cordinate:  5
Y-Cordinate:  6
.................................
Testing point:
X-Cordinate:  7
Y-Cordinate:  8
.............Checking Result...................
the given test point is:  Left Turn
```

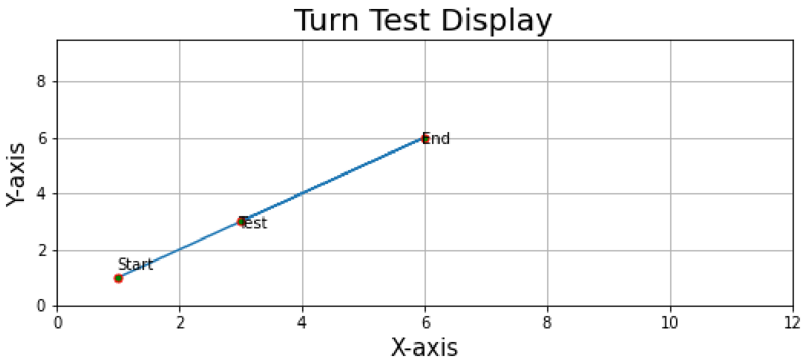


```
Please Enter starting point's X-Cordinate and Y-cordinate....1,3
Please Enter ending point's X-Cordinate and Y-cordinate....5,6
Please Enter testing point's X-Cordinate and Y-cordinate....7,6
Name: Sudhan Kandel
Roll No: 2
Section: A
Starting point:
```

```
X-Cordinate:  1
Y-Cordinate:  3
.................................
Ending point:
X-Cordinate:  5
Y-Cordinate:  6
.................................
Testing point:
X-Cordinate:  7
Y-Cordinate:  6
.............Checking Result...................
the given test point is:  Right Turn
```



```
Please Enter starting point's X-Cordinate and Y-cordinate....1,1
Please Enter ending point's X-Cordinate and Y-cordinate....6,6
Please Enter testing point's X-Cordinate and Y-cordinate....3,3
Name: Sudhan Kandel
Roll No: 2
Section: A
Starting point:
X-Cordinate:  1
Y-Cordinate:  1
.................................
Ending point:
X-Cordinate:  6
Y-Cordinate:  6
.................................
Testing point:
X-Cordinate:  3
Y-Cordinate:  3
.............Checking Result...................
the given test point is:  Colinear
```



3. **Write a program for Checking whether polygon created in Q1. is convex or not.**

```
import matplotlib.pyplot as plt
class Node:
    def __init__(self,data):
        self.data = data;
        self.previous = None;
        self.next = None;

class DoublyLinkedList:
    def __init__(self):
        self.head = None;
        self.tail = None;
    def addNode(self, data):
        newNode = Node(data);
        if(self.head == None):
            self.head = self.tail = newNode;
            self.head.previous = None;
            self.tail.next = None;
        else:
            self.tail.next = newNode;
            newNode.previous = self.tail;
            self.tail = newNode;
            self.tail.next = None;
class Polygon(DoublyLinkedList):
    def __init__(self):
        super(Polygon, self).__init__()
        number_of_vertices=int(input("Please Enter the number of vertices"))

        for i in range(number_of_vertices):
            x,y=input("Please Enter X and Y cordinate").split(',')
            x=int(x)
            y=int(y)
            self.addNode([x,y])
    def turntest(self,points):
        area=(points[1][0]-points[0][0])*(points[2][1]-points[0][1])-(points[2][0]-points[0][0])*(points[1][1]-points[0][1])
        if area>0:
            return "Left"
        elif area<0:
            return "Right"
        else:
```

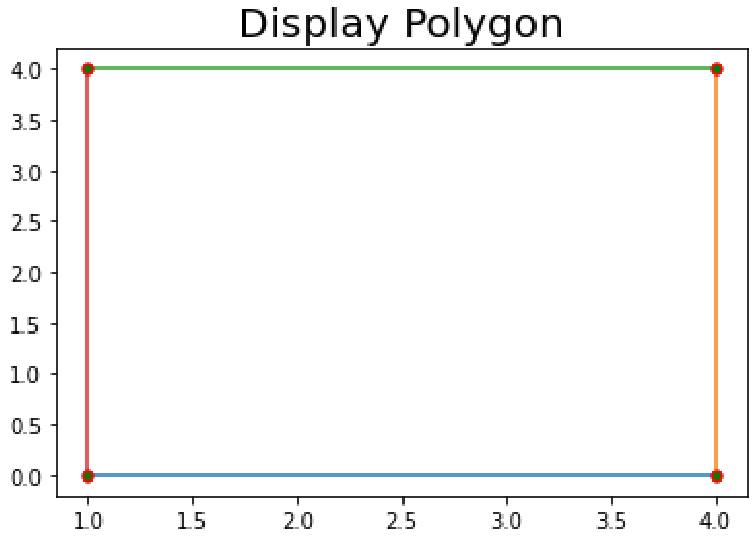
```
        return "Colinear"
def checkconvex(self):
    turn=[]
    cur=self.head
    while cur:
        if cur==None:
            cur1=self.head
            while cur1:
                a=cur1.data
                cur1=cur1.next
                b=cur1.data
                cur1=cur1.next
                c=cur1.data
                break
        else:
            a=cur.data
            cur=cur.next
            b=cur.data
            cur=cur.next
            if cur==None:
                cur2=self.head
                while cur2:
                    c=cur2.data
                    break
            else:
                c=cur.data
                cur=cur.previous
            turn.append(self.turntest([a,b,c]))
    return turn

def dislpay(self):
    print("\n................................")
    print("Name: Sudhan Kandel","\nRoll No: 2","\nSection: A")
    print("\n................................")
    current=self.head
    if(self.head==None):
        print("List is empty")
        return
    while(current!=None):
        print("Vetrices of the polygons is: ")
```

```
            print(current.data)
            current=current.next
        print("............Checking Result...................")

        print("Turn Test Result is: ",self.checkconvex())
        List=self.checkconvex()
        result = all(element == List[0] for element in List)
        if (result):
            print("Given Polygon is Convex")
        else:
            print("Given Polygon is not Convex")
    def visualization(self):
        self.dislpay()
        cur = self.head;
        while cur:
            a=cur.data
            cur=cur.next
            if cur==None:
                cur1=self.head
                while cur1:
                    b=cur1.data
                    break
            else:
                b=cur.data
            plt.plot([a[0],b[0]],[a[1],b[1]], linestyle="-", marker="o", markersize=5, markeredgecolor="red",
markerfacecolor="green")
            plt.title("Display Polygon",fontdict={'fontsize':20})
pol=Polygon()
pol.visualization()
```

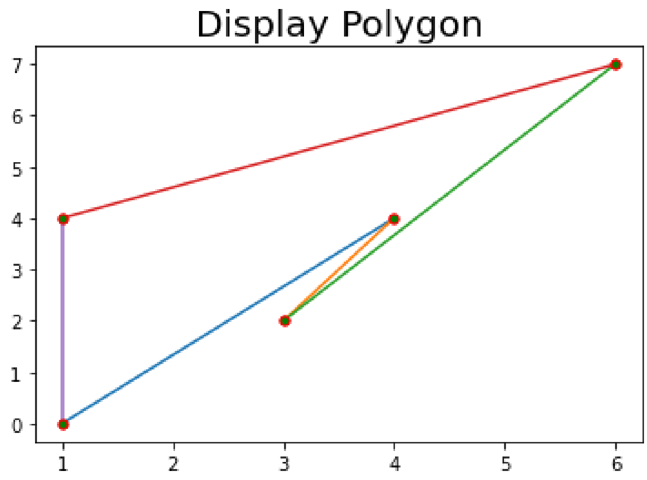
OUTPUT:

```
Please Enter the number of vertices4
Please Enter X and Y cordinate1,0
Please Enter X and Y cordinate4,0
Please Enter X and Y cordinate4,4
Please Enter X and Y cordinate1,4

.................................
Name: Sudhan Kandel
Roll No: 2
Section: A

.................................
Vetrices of the polygons is:
```

```
[1, 0]
Vetrices of the polygons is:
[4, 0]
Vetrices of the polygons is:
[4, 4]
Vetrices of the polygons is:
[1, 4]
.............Checking Result...................
Turn Test Result is:  ['Left', 'Left', 'Left']
Given Polygon is Convex
```



```
Please Enter the number of vertices5
Please Enter X and Y cordinate1,0
Please Enter X and Y cordinate4,4
Please Enter X and Y cordinate3,2
Please Enter X and Y cordinate6,7
Please Enter X and Y cordinate1,4

................................
Name: Sudhan Kandel
Roll No: 2
Section: A

................................
Vetrices of the polygons is:
[1, 0]
Vetrices of the polygons is:
[4, 4]
Vetrices of the polygons is:
[3, 2]
Vetrices of the polygons is:
[6, 7]
Vetrices of the polygons is:
[1, 4]
.............Checking Result...................
Turn Test Result is:  ['Right', 'Left', 'Left', 'Left']
Given Polygon is not Convex
```



Code link:
https://github.com/sudhankandel/CGlab/blob/main/SUDHAN_KANDEL_2_Lab2.ipynb