In [271]:
```
def isPointPositive(point, line):
    result = (point[0] * line[0]) + (point[1] * line[1]) + line[2]
    if result == 0:
        return 3
    elif result > 0:
        return True
    else:
        return False

def doRun(NUMPOINTS):
    centerpt = [random.uniform(-1,1), random.uniform(-1,1)]
    otherpt = [random.uniform(-1,1), random.uniform(-1,1)]

    points = []

    for a in range(NUMPOINTS):
        points.append([random.uniform(-1,1), random.uniform(-1,1)])

    linevect = [otherpt[0] - centerpt[0], otherpt[1] - centerpt[1], 0]
    origlinevect = [linevect[0], linevect[1], 0]

    for point in points:
        if point[1] > (otherpt[1]-centerpt[1])/(otherpt[0]-centerpt[0]
)*(point[0]-centerpt[0]):
            point.append(1)
        else:
            point.append(-1)

    linevect = [0,0,0]
    counter = 0
    for a in range(1000):
        counter += 1
        misclassified = []
        for point in points:
            score = isPointPositive(point, linevect)
            if score == 3:
                misclassified.append(point)
            elif score:
                if point[2] == -1:
                    misclassified.append(point)
            elif not score:
                if point[2] == 1:
                    misclassified.append(point)
            else:
                misclassified.append(point)

        #print("Iteration number " + str(counter))
        #print("Number of misclassified points: " + str(len(misclassif
```

```
ied)))

        if len(misclassified) == 0:
            break

        targetpoint = random.choice(misclassified)

        linevect[0] += targetpoint[2]*targetpoint[0]
        linevect[1] += targetpoint[2]*targetpoint[1]
        linevect[2] += targetpoint[2]

    #print(counter)

    wrongcount = 0
    for a in range(10000):
        point1 = [random.uniform(-1,1), random.uniform(-1,1)]
        if point1[1] > (otherpt[1]-centerpt[1])/(otherpt[0]-centerpt[0
])*(point1[0]-centerpt[0]):
            point1.append(1)
        else:
            point1.append(-1)
        score = isPointPositive(point1, linevect)
        if score == True and point1[2] == -1:
            wrongcount+=1
        elif score == False and point1[2] == 1:
            wrongcount+=1

    #print(wrongcount/10000.0)
    return [counter, wrongcount/10000.0]
```

In [272]:
```
avgCount = 0
avgWrongCount = 0

#Tests for 7 and 8
for a in range(1000):
    res = doRun(10)
    avgCount += (res[0]/1000.0)
    avgWrongCount += (res[1]/1000.0)

print("Average Count (N = 10): " + str(avgCount))
print("Average Wrong Count (N = 10): " + str(avgWrongCount))

avgCount = 0
avgWrongCount = 0

#Tests for 9 and 10
for a in range(1000):
    res = doRun(100)
    avgCount += (res[0]/1000.0)
    avgWrongCount += (res[1]/1000.0)

print("Average Count (N = 100): " + str(avgCount))
print("Average Wrong Count (N = 100): " + str(avgWrongCount))
```

```
Average Count (N = 10): 10.70299999999999
Average Wrong Count (N = 10): 0.10871729999999986
Average Count (N = 100): 104.01100000000008
Average Wrong Count (N = 100): 0.013603
```

Therefore the answers to 7-10 are:

1. B
2. C
3. B
4. B

In [ ]:
```
```