In [ ]:
```
import sys

from pyspark import SparkContext
from pyspark.streaming import StreamingContext

# Function to map the point to the right quadrant
def get_quadrant(line):
    # Convert the input string into a pair of numbers
    try:
        (x, y) = [float(x) for x in line.split()]
    except:
        print "Invalid input"
        return ('Invalid points', 1)

    # Map the pair of numbers to the right quadrant
    if x > 0 and y > 0:
        quadrant = 'First quadrant'
    elif x < 0 and y > 0:
        quadrant = 'Second quadrant'
    elif x < 0 and y < 0:
        quadrant = 'Third quadrant'
    elif x > 0 and y < 0:
        quadrant = 'Fourth quadrant'
    elif x == 0 and y != 0:
        quadrant = 'Lies on Y axis'
    elif x != 0 and y == 0:
        quadrant = 'Lies on X axis'
    else:
        quadrant = 'Origin'

    # The pair represents the quadrant and the counter increment
    return (quadrant, 1)

if __name__ == "__main__":
    # Initialize a SparkContext with a name
    spc = SparkContext(appName="QuadrantCount")

    # Create a StreamingContext with a batch interval of 2 seconds
    stc = StreamingContext(spc, 2)

    # Checkpointing feature
    stc.checkpoint("checkpoint")

    # Creating a DStream to connect to hostname:port (like localhost:9999)
    lines = stc.socketTextStream("172.31.20.58", 9999)

    # Function that's used to update the state
    updateFunction = lambda new_values, running_count: sum(new_values) + (runn
ing_count or 0)

    # Update all the current counts of number of points in each quadrant
    running_counts = lines.map(get_quadrant).updateStateByKey(updateFunction)

    # Print the current state
    running_counts.pprint()
```

```
# Start the computation
stc.start()

# Wait for the computation to terminate
stc.awaitTermination()
```