CS 417 Assignment 2
Sakib Jalal (sfj19), Nathan Miniovich (nm540), Tanya Balaraju (tb463), Aditya Geria (aag177)

TEST CASES

1) Normal (expected) input

This test case and the one following it represent normal cases: the client receives valid input; PlaceServer looks up the city name provided by the client and returns the corresponding coordinates to the client; the client sends the latitude and longitude to AirportServer; and AirportServer looks up the coordinates and returns the 5 nearest airports to the client.

Client Input:

```
      java Client -ha python.cs.rutgers.edu -hp python.cs.rutgers.edu –pa 23124 -pp 23124
Newark NJ
```

Client Output:

```
      looking up //python.cs.rutgers.edu:23124/Places
      looking up //python.cs.rutgers.edu:23124/Airports
      Searching up: Newark, NJ
      East Newark borough, NJ: 40.749901, -74.161105
      code=EWR, name=Newark Liberty International, state=NJ, distance: 3.476799000792525
      miles
      code=TEB, name=Teterboro, state=NJ, distance: 9.027352005750439 miles
      code=CDW, name=Fairfield, state=NJ, distance: 10.362043928873005 miles
      code=LGA, name=New York/La Guardia, state=NY, distance: 13.726012837751917 miles
      code=MMU, name=Morristown, state=NJ, distance: 13.972017970847714 miles
```

PlaceServer Output:

```
      Looking up: Newark, NJ... Success
```

AirportServer Output:

```
      Looking up: 40.538204, -74.378585... Success!
```

2) Normal (expected) input with two-word city name

Double quotes around the city name ensure that a two-word city name is valid input.

Client Input:

```
      java Client -ha python.cs.rutgers.edu -hp python.cs.rutgers.edu -pa 23125 -pp 23125
      "San Francisco" CA
```

Client Output:

```
      looking up //python.cs.rutgers.edu:23125/Places
      looking up //python.cs.rutgers.edu:23125/Airports
      Searching up: San Francisco, CA
      San Francisco city, CA: 37.759881, -122.437392
      code=NGZ, name=Alameda NAS, state=CA, distance: 7.62641698570278 miles
      code=SFO, name=San Francisco, state=CA, distance: 12.232636364232368 miles
      code=OAK, name=Oakland, state=CA, distance: 13.124376759503594 miles
      code=HWD, name=Hayward, state=CA, distance: 19.931425641197567 miles
```

```
code=SQL, name=San Carlos, state=CA, distance: 21.042848627766418 miles
```

PlaceServer Output:

```
Looking up: San Francisco, CA... Success
```

AirportServer Output:

```
Looking up: 37.759881, -122.437392... Success!
```

3) Servers running on separate machines

- *Export CLASSPATH and run “rmiregistry 23123 &” and “java –Djava.security.policy=policy PlaceServer 23123” on python.cs.rutgers.edu*
- *Export CLASSPATH and run “rmiregistry 23124 &” and “java –Djava.security.policy=policy AirportServer 23124” on perl.cs.rutgers.edu*

Client Input:

```
java Client -hp python.cs.rutgers.edu –pp 23123 -ha perl.cs.rutgers.edu –hp 23124
Princeton NJ
```

It should run as expected and give proper output.

4) Too many arguments to client

This error is caught and handled by the client, before any messages are sent to PlaceServer, and the appropriate error message is returned.

Client Input:

```
java Client -ha python.cs.rutgers.edu -hp python.cs.rutgers.edu –pa 23124 –pp 23124
Too Many CA
```

Client Output:

```
Extra arguments found
```

5) Too few arguments to client

This error, like the previous one, is caught and handled by the client, if a cityArg or stateArg is not found.

Client Input:

```
java Client -ha python.cs.rutgers.edu -hp python.cs.rutgers.edu –pa 23124 –pp 23124
Piscataway
```

Client Output:

```
City or state missing in args
```

6) Port number less than 1024

This error is caught in the client as well, as a preliminary check.

Client Input:

```
java Client -ha python.cs.rutgers.edu -hp python.cs.rutgers.edu –pa 23124 –pp 100
Newark NJ
```

Client Output:

```
Port number is less than 1024, exiting...
```

7) Invalid port number

This error is also caught in the client, when an integer port number cannot be found in the arguments.

Client Input:

```
java Client -ha python.cs.rutgers.edu -hp python.cs.rutgers.edu –pa 23124 –pp
NotANumber Newark NJ
```

Client Output:

```
Invalid port number
```

8) Nonexistent place name

This error is caught in PlaceServer, which returns the appropriate error messages.

Client Input:

```
java Client -ha python.cs.rutgers.edu -hp python.cs.rutgers.edu –pa 23124 –pp 23124
Nowhere NJ
```

Client Output:

```
looking up //python.cs.rutgers.edu:23124/Places
looking up //python.cs.rutgers.edu:23124/Airports
Searching up: Nowhere, NJ

Place "Nowhere, NJ" not found!
```

PlaceServer Output:

```
Looking up: Nowhere, NJ... Failure
```

9) AirportServer is offline during lookup

Although the client connects to PlaceServer, and PlaceServer successfully finds the coordinates of the specified city, AirportServer is offline and cannot return the 5 nearest airports.

Client Input:

```
java Client -ha python.cs.rutgers.edu -hp python.cs.rutgers.edu -pa 23125 -pp 23125
Edison NJ
```

Client Output:

```
looking up //python.cs.rutgers.edu:23125/Places
looking up //python.cs.rutgers.edu:23125/Airports
Searching up: Edison, NJ
Edison CDP, NJ: 40.538204, -74.378585
Client exception: java.rmi.ConnectException: Connection refused to host:
128.6.13.233; nested exception is:
      java.net.ConnectException: Connection refused
```

PlaceServer Output:

```
Looking up: Edison, NJ... Success
```

10) PlaceServer is offline during lookup

The client attempts to connect to PlaceServer, but PlaceServer is offline and unable to return coordinates.

Client Input:

```
java Client -ha python.cs.rutgers.edu -hp python.cs.rutgers.edu -pa 23125 -pp 23125
“New York” NY
```

Client Output:

```
looking up //python.cs.rutgers.edu:23125/Places
looking up //python.cs.rutgers.edu:23125/Airports
Searching up: New York, NY
Client exception: java.rmi.ConnectException: Connection refused to host:
128.6.13.233; nested exception is:
      java.net.ConnectException: Connection refused
```

11) Both servers offline

The client attempts to connect to PlaceServer first, but PlaceServer cannot return any coordinates because it is offline.

Client Input:

```
java Client -ha python.cs.rutgers.edu -hp python.cs.rutgers.edu -pa 23125 -pp 23125
"San Francisco" CA
```

Client Output:

```
looking up //python.cs.rutgers.edu:23125/Places
looking up //python.cs.rutgers.edu:23125/Airports
Searching up: San Francisco, CA
Client exception: java.rmi.ConnectException: Connection refused to host:
128.6.13.233; nested exception is:
      java.net.ConnectException: Connection refused
```

12) Client dies during lookup

The client goes offline before the 5 nearest airports can be returned to it.
The results are unable to be sent back to the client, but both PlaceServer and AirportServer run normally.

Client Input:

```
java Client -ha python.cs.rutgers.edu -hp python.cs.rutgers.edu -pa 23125 -pp 23125
Orlando FL
```

Client Output:

```
looking up //python.cs.rutgers.edu:23125/Places
looking up //python.cs.rutgers.edu:23125/Airports
Searching up: Orlando, FL
```

PlaceServer Output:

```
Looking up: Orlando, FL... Success
```

AirportServer Output:

```
Looking up: 28.533513, -81.375789... Success!
```