Natalia Alonso
COMP 282 Fall 2013
December 8, 2013

Project 3 – Treasure Hunt

Database
The files for loading into the tables of the database are built using a custom program called AlonsoBuildTables. The function buildTables parses the waypointNeighbor.txt file into separate text files with the necessary information for the city table, the map table, and the treasure table. A Scanner takes in the waypointNeighbor.txt file. New files called cityFile, mapFile, and treasureFile create text files called city, map, and treasure respectively. PrintStream objects named city, map, and treasure are created to print to cityFile, mapFile, and treasureFile respectively. While the Scanner has data, a loop adds each value into an int variable for each waypoint, as well as any neighbors. If there is a city cost, the x and y coordinates, and the cost are added as a line of the text file. If there is a treasure, the x and y coordinates, and the treasure value are added as a line of the text file. If there are map coordinates, the x and y coordinates of the location, and the x and y coordinates of the map location are added as a line of the text file.

These text files, along with a text file for the provided player information are loaded onto the database via sftp. Once the files are on the table base, two sets of tables are created. iCity, iMap, iTreasure, and iPlayer are used as initial value to copy to the game tables; City, Map, Treasure, and Player are used to play the game and are reset at the beginning of each simulation. The game tables are reloaded by deleted the data from the tables and then copying the values from the initial value tables using the CreateTable function in Alonso282P3.

Each time there is a change to the value of the tables, they are updated via the game client. The updateMapTable function marks treasure map coordinates as (0,0) so the game client will not recognize them as a valid treasure location. The updatePlayer function calls updateWealth to update the wealth of the player and calls updatePlayerTable to update the place. The updatePlayerTable function replaces the previous place with the new x-y coordinates of the player. The updateTreasureTable functions updates the corresponding treasure value to 0 once it has been picked up by a player. The updateWealth function replaces the previous wealth value with the new amount.

Game Program
Once the map and waypoints are displayed, and the tables are reset for the new simulation, the players are created and initialized. The path for each player to its destination is found and loaded. The order that players take turns is selected at random using a random number chosen from 0-3 based on the current date and time. During each turn, if the player that is selected at random is in fact still playing, his next point on his path is visited and all of the information is updated. If there is a treasure, the player collects it. If there is a city, the player will pay for food if it has enough wealth. If there is a map, the player will pick up the map and a new path will be generated to the treasure location. Once the treasure location is reached, the player will resume its path from that point to its original goal location. If there are two players on one waypoint, the players have a contest and the winner gets 1/3 of the wealth of the loser. If the player has reached its final destination, its remaining strength is added to its wealth and it is marked as no longer playing. The loop to determine turns no longer tests for this “no longer playing” flag (a boolean value.)

The information about cities, maps, treasure, and player locations and wealth is gather via functions that prompt the SQL database. The getCost function takes in x-y coordinates and returns the cost at that point. The getDBPoint function takes in a player number and returns the current place/location for that player. The getTPoint function takes in x-y coordinates and returns the x-y coordinates of the treasure indicated by the map at that location. The getTreasure function takes in x-y coordinates and returns the value of the treasure at that location. The getWealth function takes in a player number and returns the wealth of that player.