I-Training Phase

#1| statsList

statsList, which returns a list of pairs, where each pair describes a piece of furniture follows the following hierarchy to get the desired output:

1- statsList1, which is a statsList's helper function receives as an argument the output of the function statsListUnsorted (another statsList's helper function), which returns the statsList but unsorted, statsList1 sorts the unsorted statsList provided by statsListUnsorted.
2- statsList1 uses the function superCoolAhmedSort to sort the list of triples.
3- superCoolAhmedSort is a sorting function that is tailored to the list of triples we are dealing with.
4- superCoolAhmedSort uses two helper functions, maxL and delete1, functions tailored to provide the functionality of deleting and finding the maximum but tailored to deal with the list of triples.
5- statsListUnsorted, which is one of the statList function helpers provides an unsorted statList, it does so by using the function statsList1stHelper, the training list, and a statsList with the 5 furniture pieces without any information.
6- statsList1stHelper provides statsList2ndHelper, which is statsList1stHelper's helper with each room separately.
7- statsList2ndHelper takes as an argument a room and the current statsList, it updates the statsList based on that room by using the function generate.

#2| generate

generate, updates the current statsList given based on given room, it does so by following the following hierarchy:

1- generate uses the function generateHelperRight (generate's helper function) to update the statsList with all the needed triple entries involving the position right, it then passes the updated statsList to generateHelperBelow (another generate's helper function) to update the statsList with all the needed triples entries involving the position below, generateHelperBelow provides an updated statsList with the desired functionality.
2- generateHelperRight (generate's helper function) receives the room and a statsList, it updates the statsList based on the room, by using the function findFurnitureUpdate
3- generateHelperBelow (generate's helper function) receives the transpose of a room and a statsList, it updates the statsList based on the room, by using the function findFurnitureUpdate

#3| findFurnitureUpdate

FindFurnitureUpdate adds a triple to the list of triples in the statsList, it does so by following the following hierarchy:

1- FindFurnitureUpdate uses findFurnitureUpdateRightHelper (findFurnitureUpdate helper) to add triples involving the direction right, and findFurnitureUpdateBelowHelper (another findFurnitureUpdate helper) to add a triple involving the direction below.
2- findFurnitureUpdateRightHelper (findFurnitureUpdate helper) adds the triple to its relevant list of triples in the list of lists in the statsList
3- findFurnitureUpdateBelowHelper (findFurnitureUpdate helper) adds the triple to its relevant list of triples in the list of lists in the statsList
4- addOrIncrement, which is a helper function used by both findFurnitureUpdateRightHelper and findFurnitureUpdateBelowHelper decides whether the triple exists in the list of triples and thus increments its occurrence, or if does not exist, and thus adds it.

I-Generation Phase

#1| furnishRoom

furnishRoom returns a furnished room, based on the received dimension and the starting object by following the following hierarchy:

1- furnishRoom uses the helper function createRoom to construct a room with the required dimensions, the room contains 1 furniture piece only which is the starting object provided as an argument to the function furnishRoom, furnishRoom uses its second helper furnishRoomHelper to add furniture pieces to the room.
2- createRoom (furnishRoom helper) creates a room with the required dimension and the top left corner furniture piece inserted.
3- furnishRoomHelper (furnishRoom helper) receives the indices of the next element to add in the room, it uses three helper functions (getFromLeft, getFromUp, getPossibleNeighbour, buildRoom), getFromLeft gets the interesting list of triples from the element at the left of the element being inserted, , getFromUp gets the interesting list of triples from the element at the top of the element being inserted, getPossibleNeighbour uses the output of getFromLeft and getFromUp to provide a suitable element to be added at the current spot, finally buildRoom modifies the room to insert the new item.
4- getFromLeft (furnishRoomHelper helper) uses getFurnStat to get the list of triples, needed to provide its boss function (furnishRoomHelper) with the list needed to insert a suitable element.
5- getFromRight (furnishRoomHelper helper) uses getFurnStat to get the list of triples, needed to provide its boss function (furnishRoomHelper) with the list needed to insert a suitable element.
6- buildRoom (furnishRoomHelper helper) modifies the current room to include a newly inserted furniture piece.

#2| getFurnStat

getFurnStat uses the statList to get the list of lists of triples for the required furniture piece by following the following hierarchy:

1- getFurnStat calls its helper function (getFurnStatHelper) by providing it with required object and the output of the function of statsList function.
2- getFurnStatHelper (getFurnStat helper) searches the statsList function output for the required furniture piece to get the relevant list of lists of triples.

// sometimes the function getFurnStat gets depressed form being surrounded by such complex functions and the fact that it is a rather simple function.

// poor getFurnStat feels left out ☹

#3| getPossibleNeighbour

getPossibleNeighbour returns a random furniture piece from the two lists of triples provided by following the following hierarchy:

1- getPossibleNeighbour uses the helper function getPossibleNeighbourHelper to get a list of all the furniture pieces that could be added with each ones frequency included by repeating the furniture piece based on its frequency, and then its outputs a random piece out of the list by using the function randomZeroToX.
2- getPossibleNeighbourHelper (getPossibleNeighbour helper) which receives two lists representing a list of triples uses the helper function expand to include all the furniture pieces in the 2 lists with their relative frequency.
3- expand (getPossibleNeighbourHelper helper) receives an item and a counter it returns a list with the item repeated based on the counter provided.