```
#include "treasure.h"
#include "map.h"
#include "searchers.h"
#include "utilities.h"

#ifdef DEBUG
   #include <iostream>
#endif

enum Directions {
   NORTH,
   EAST,
   SOUTH,
   WEST
};

SearcherList runSimulation(const Map& map, size_t startX, size_t startY, int nbSimulation) {
   SearcherList searcherList(nbSimulation, initSearcher());

   for (auto& searcher: searcherList) {
      runSearcher(map, startX, startY, searcher);
   }

   return searcherList;
}

void runSearcher(const Map& map, size_t startX, size_t startY, Searcher& searcher) {
   size_t currentX = startX;
   size_t currentY = startY;

   int maxSteps = (int)(getHeight(map) * getWidth(map));
   int steps = 0;

   #ifdef DEBUG
      Map displayMap = map;
   #endif
   for (; steps < maxSteps and getStatus(searcher) == UNDEFINED; ++steps) {

      switch ((Directions)getRandomInRange(3)) {
         case NORTH:
            ++currentY;
            break;
         case EAST:
            ++currentX;
            break;
         case SOUTH:
            --currentY;
            break;
         case WEST:
            --currentX;
            break;
      }

      switch (getMapValue(map, currentX, currentY)) {
         case MS_OUT:
            setStatus(searcher, LOST);
            break;
         case MS_WATER:
            setStatus(searcher, DROWNED);
            break;
         case MS_TREASURE:
            setStatus(searcher, RICH);
            break;
      }

      #ifdef DEBUG
         if(getStatus(searcher) != LOST and getMapValue(map, currentX, currentY) != MS_START)
            setMapValue(displayMap, currentX, currentY, MS_TREASURE);
      #endif
   }

   #ifdef DEBUG
      displayWorld(displayMap);
      std::cout << std::endl;
   #endif

   if (steps == maxSteps) {
```

```cpp
      setStatus(searcher, EXHAUSTED);
   }

   setSteps(searcher, steps);
}

bool getStatistics(const SearcherList& list, double& probability, double& avgSteps) {
   if (list.empty()) return false;

   int sum = 0, counter = 0;
   for (const Searcher& searcher : list) {
      if (getStatus(searcher) == RICH) {
         sum += getSteps(searcher);
         ++counter;
      }
   }
   probability = counter / (double)list.size();
   if(counter > 0)
      avgSteps = (double)sum / (double)counter;
   else
      avgSteps = 0;

   return true;
}
```