include

include

include

include

include

include

using namespace std;

vector makeRandVec(int count){ vector retVal(count); int val = rand()%(2*count); for (size_t i = 0; i < count; i++) { retVal[i] = val++; } return retVal; }

bool findRightAnswer(const vector& Bob, const vector& Alice, const int len, const int messageSize){ int ref; int j = 0; for (size_t i = 0; i < messageSize; i++) { ref = Bob[rand()%len]; //cout << ref << endl; if (std::find(Alice.begin(), Alice.end(), ref) != Alice.end()) { j = j+1; } } if(j > (((double) messageSize)/10)){ return true; } return false; }

bool alwaysRightAnswer(const vector& Bob, const vector& Alice, const int len){ int BobIndex = 0; int AliceIndex = 0; int matches = 0; while (BobIndex < len && AliceIndex < len) { if(Bob[BobIndex] > Alice[AliceIndex]){ AliceIndex = AliceIndex+1; }else if(Bob[BobIndex] == Alice[AliceIndex]){ BobIndex = BobIndex+1; AliceIndex = AliceIndex+1; matches = matches + 1; }else{ BobIndex = BobIndex+1; } }

if(len/10 <= matches) return true; return false;

}

int testAlgorithm(const int len, const int logs){ vector Bob = makeRandVec(len); vector Alice = makeRandVec(len); bool ourAlg = findRightAnswer(Bob,Alice,len, logs); bool ans = alwaysRightAnswer(Bob,Alice,len); if(ourAlg != ans && !ans){ return 2; }else if(ourAlg == ans){ return 1; }else{ return 0; } }

int main(int argc, char const *argv[]) { srand(time(NULL));

double size; cout << "Input list size:" << endl; cin >> size; cout << "Input number of tests:" << endl; double count = 100000; cin >> count; int logsize = (int) log2(size);

double truePos = 0; double falsePos = 0; int elem; for (size_t i = 0; i < count; i++) { elem = testAlgorithm(size, logsize); if(elem == 2){ falsePos = falsePos + 1; }else if(elem == 1){ truePos = truePos + 1; } } cout << “False Positives:” << falsePos << endl; cout << “True Pos/Fal:” << truePos << endl; cout << “Tests:” << count << endl; cout << “Percentage:” << (int) ((truePos/count)*100+0.5) << endl;

return 0; }