High Performance Computing
CS-4373
Final Project

Team 2 Project Report

Members:

Austin Lively

Luke Runnels

David Fantin

The University of Tulsa

Date Submitted:
May 5th, 2022

Finding the Determinant of a Matrix

Algorithm Description:

Our implementation of the determinant is based off of LU Decomposition. The premise behind this algorithm is reducing our matrix down to a product a lower-triangular matrix and an upper-triangular matrix.

$$\begin{bmatrix} l_{11} & 0 & \cdots & 0 \\ l_{21} & l_{22} & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ l_{n1} & l_{n2} & \cdots & l_{nn} \end{bmatrix} \times \begin{bmatrix} u_{11} & u_{12} & \cdots & u_{1n} \\ 0 & u_{22} & \cdots & u_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & u_{nn} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nn} \end{bmatrix}$$

Rigidly, if U is initially a copy of A, the elements of L and U are represented as

$$l_{ij} = \frac{u_{ji}}{u_{ii}} \tag{1}$$

$$u_{jk} = u_{jk} - l_{ij} \times u_{ik} \tag{2}$$

Fortunately, there is an approach that we used for calculating the determinant based off of LU decomposition without having to create two separate matrices. This approach is outlined in the following pseudocode:

```
for i = 0 to N:
    for j = i+1 to N:
        ratio = A[j][i] / A[i][i];
        for k = 0 to N:
            A[j][k] -= ratio*A[i][k]
```

```
det = 0
for i = 0 to N:
    det *= A[i][i]
```

Libraries used:

The omp.h (OpenMP) library was used in this implementation. The function omp_get_wtime() was used to time the program.

Output from tests: [Note: See the Appendix for all outputs]

Sample output (Size%3A 1000):

```
1 threads
File: /home/lar9482/Final_Project_1/m1000x1000.bin
Size: 1000
Det: +inf
Log Det: 743.122063
Time: 0.8552089259028435

2 threads
File: /home/lar9482/Final_Project_1/m1000x1000.bin
Size: 1000
```

```
Det: +inf
Log Det: 743.122063
Time: 0.4532068748958409

4 threads
File: /home/lar9482/Final_Project_1/m1000x1000.bin
Size: 1000
Det: +inf
Log Det: 743.122063
Time: 0.2316409051418304

8 threads
File: /home/lar9482/Final_Project_1/m1000x1000.bin
Size: 1000
Det: +inf
Log Det: 743.122063
Time: 0.1220189523883164

16 threads
File: /home/lar9482/Final_Project_1/m1000x1000.bin
Size: 1000
Det: +inf
Log Det: 743.122063
Time: 0.0712027358822525

32 threads
File: /home/lar9482/Final_Project_1/m1000x1000.bin
Size: 1000
Det: +inf
Log Det: 743.122063
Time: 1.1239543850533664
```

Results from testing:

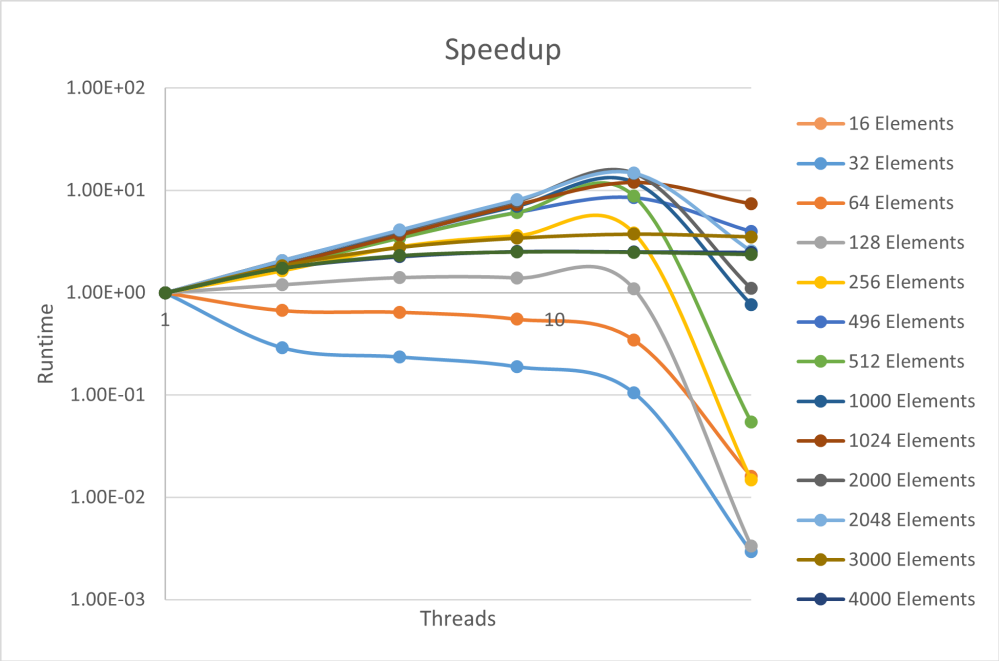

Figure 1: Graph of the speedup that occurs with multiple cores. [LU Decomposition]

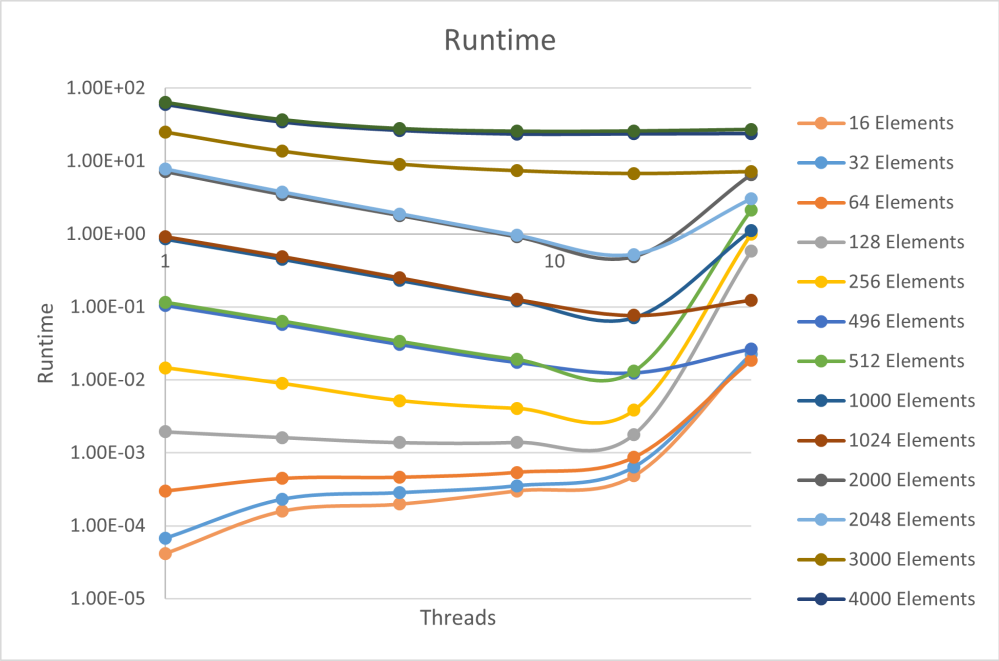

Figure 2: Graph of the runtime for various thread counts. [LU Decomposition]

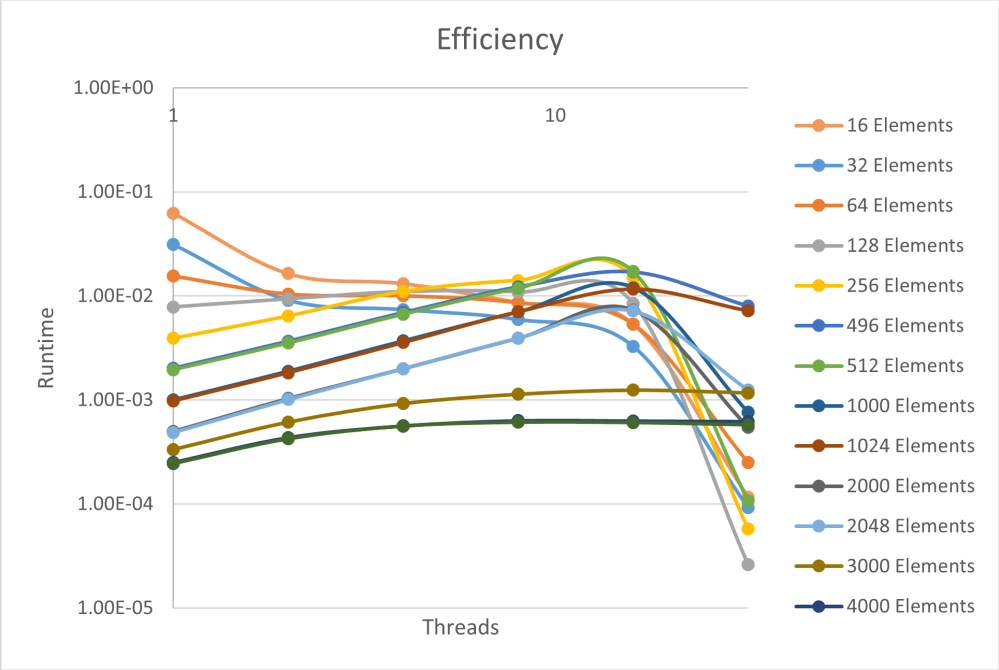

Figure 3: Graph of the efficiency for various thread counts. [LU Decomposition]

Analysis of results:

To analysis our results, we graphed the runtime, speed, efficiency after running our program on 1, 2, 4, 8, and 16 threads.

Scalability:

We can observe volatile trends in Figure 3. Therefore, it is safe to assume that our implementation is weakly scalable.

Determinant Solver - Source Code:

```c
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include <omp.h>

//Change this variable to input different files.
//NOTE: Overflow is an issue for SIZE>=512. Solve this later cuz I'm tired right now.
#define SIZE 32

int count_digits();
void perform_LU_Decomposition(double *matrix[SIZE], int thread_count);
void read_matrix(double *matrix[SIZE]);
double calculate_det(double *matrix[SIZE]);
double calculate_log_det(double *matrix[SIZE]);

int main(int argc, char *argv[]) {

  int thread_count = strtol(argv[1], NULL, 10);
  printf("%d threads\n", thread_count);
  double *matrix[SIZE];
  int i, j;
  for (i = 0; i < SIZE; i++) {
    matrix[i] = (double*) malloc(SIZE * sizeof(double));
  }

  read_matrix(matrix);

  double start_time = omp_get_wtime();
  perform_LU_Decomposition(matrix, thread_count);
  double log_det = calculate_log_det(matrix);
  if (SIZE < 1000) {
    double det = calculate_det(matrix);
    printf("Det: %lf\n", det);
  }
  else {
    if (log_det < 0) {
      printf("Det: -");
    }
    else {
      printf("Det: +");
    }
    printf("inf\n");
  }

  printf("Log Det: %lf", log_det);
  printf("Time: %1.16f\n\n", omp_get_wtime() - start_time);
  return 0;
}


void perform_LU_Decomposition(double *matrix[SIZE], int thread_count) {
  int i, j, k;
  double ratio;

  for (int i = 0; i < SIZE; i++) {

    //Unfortunately, due to crucial data dependency in the second loop, we can't parallelize
     the first loop.
    #pragma omp parallel for num_threads(thread_count) private(j, k, ratio)
    for (j = i+1; j < SIZE; j++) {
      ratio = (double) matrix[j][i] / (double) matrix[i][i];
      for (k = 0; k < SIZE; k++) {
        matrix[j][k] = (double) matrix[j][k] - ratio*matrix[i][k];
      }
    }
  }
```

```
}

double calculate_det(double *matrix[SIZE]) {
  int i;
  double det = 1;
  #pragma omp parallel for num_threads(thread_count) private(i) reduction(*: det)
  for (i = 0; i < SIZE; i++) {
    det *= (matrix[i][i]);
  }

  return det;
}

double calculate_log_det(double *matrix[SIZE]) {
  int i;
  double det = 0;
  #pragma omp parallel for num_threads(thread_count) private(i) reduction(+: det)

  for (i = 0; i < SIZE; i++) {
    det += log10(fabs(matrix[i][i]));
  }

  return det;
}
int count_digits() {
  int count = 0;
  int size = SIZE;
  while (size != 0) {
    size /= 10;
    count++;
  }
  return count;
}

void read_matrix(double *array[SIZE]) {
  char f_name[50];
    int i,j;
    //Create filename
    // int num_digits = count_digits();
    // if (num_digits == 2) {
    //  sprintf(f_name,"m00%dx00%d.bin", SIZE, SIZE);
    // }
    // else if (num_digits == 3) {
    //  sprintf(f_name,"m0%dx0%d.bin", SIZE, SIZE);
    // }
    // else if (num_digits == 4) {
    //  sprintf(f_name,"m%dx%d.bin", SIZE, SIZE);
    // }


    if (num_digits == 2) {
      sprintf(f_name,"/home/lar9482/Final_Project_1/m00%dx00%d.bin", SIZE, SIZE);
    }
    else if (num_digits == 3) {
      sprintf(f_name,"/home/lar9482/Final_Project_1/m0%dx0%d.bin", SIZE, SIZE);
    }
    else if (num_digits == 4) {
      sprintf(f_name,"/home/lar9482/Final_Project_1/m%dx%d.bin", SIZE, SIZE);
    }


    printf("File: %s\n", f_name);
    printf("Size: %d\n", SIZE);
    //Open file
    FILE *datafile=fopen(f_name,"rb");
    //Read elelements
    for (i=0; i< SIZE; i++) {
      for (j=0; j< SIZE; j++) {
```

```
134             fread(&array[i][j],sizeof(double),1,datafile);
135                 //printf("a[%d][%d]=%f\n",i,j,array[i][j]);
136           }
137       }
138
139       //printf("Matrix has been read.\n");
140   }
```

Traveling Salesperson Problem

As a team, we attempted two separate algorithms to solve this problem:

1. Stochastic Nearest Neighbor

Algorithm Description:

Our implementation to solve the traveling sales person problem is based off of a stochastic variation of the nearest neighbor heuristic. The idea behind this approach was to use the nearest neighbor heuristic as normal, with a degree of randomness added in to explore away from the heuristic. Ideally, this would lead to exploring paths away from the nearest neighbor solution, which would result in better outcomes. To accomplish this, a parameter was used to add a degree of randomness (called ODDS). This "ODDS" determines the likely-hood of any individual step, resorting to randomness rather than the heuristic. This implementation is essentially as follows:

```
Starting city = random start;
while there are unvisited cities
    if (random chance (ODDS))
        Randomly choose an unvisited city
    else
        Choose the unvisited city closest to the last
```

Parallelization for this algorithm was done using a combination of MPI and OpenMP. MPI was used to split the work across several machines, and OpenMP was used to split the work among the cores of a single machine. In this implementation, the threads on each machine all run the same function (shown above) as many times as possible within the time limit, while remembering the best path they had seen so far.

Libraries used:

The omp.h (OpenMP) and mpi.h (MPI) libraries were used in this implementation. time.h was used for timing the algorithm.

Output from tests: [Note: See the Appendix for all outputs]

Sample output (job.34134.out):

```
[Process 6] Searched 147125 paths and found best path 43643
[Process 7] Searched 145528 paths and found best path 43588
[Process 1] Searched 146327 paths and found best path 43279
[Process 4] Searched 146170 paths and found best path 43191
[Process 5] Searched 145258 paths and found best path 43080
[Process 2] Searched 144154 paths and found best path 43495
[Process 0] Searched 146879 paths and found best path 43117
[Process 3] Searched 144559 paths and found best path 43369
[Process 5] Entering send mode...
Reducton Result: [0] 43080 [1] 5
[Process 0] Entering recv mode...
BestPathCost: 43080
227 493 603 844 994 39 816 103 85 294
```

Results from testing:

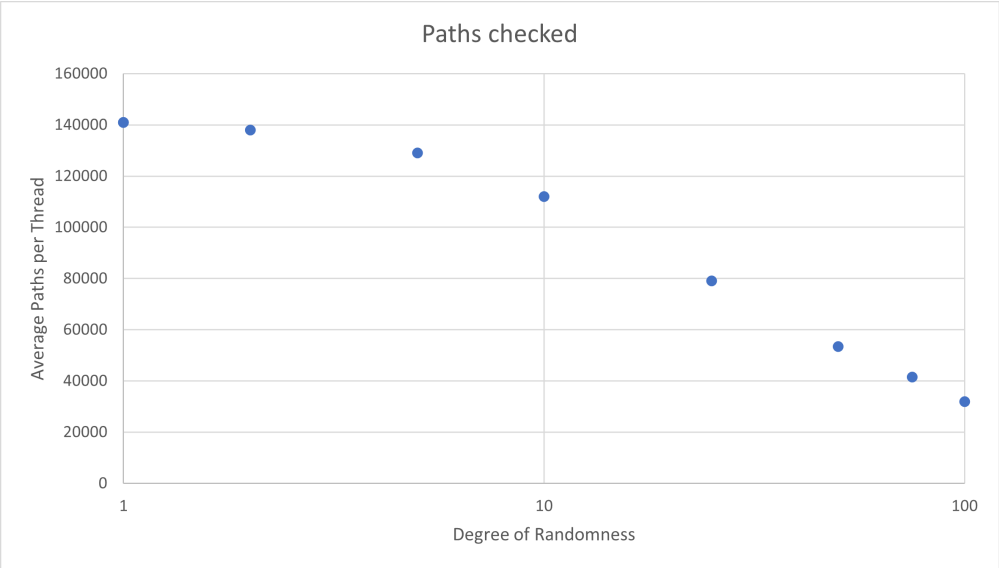

Figure 4: Paths checked per thread based on degree of randomness.

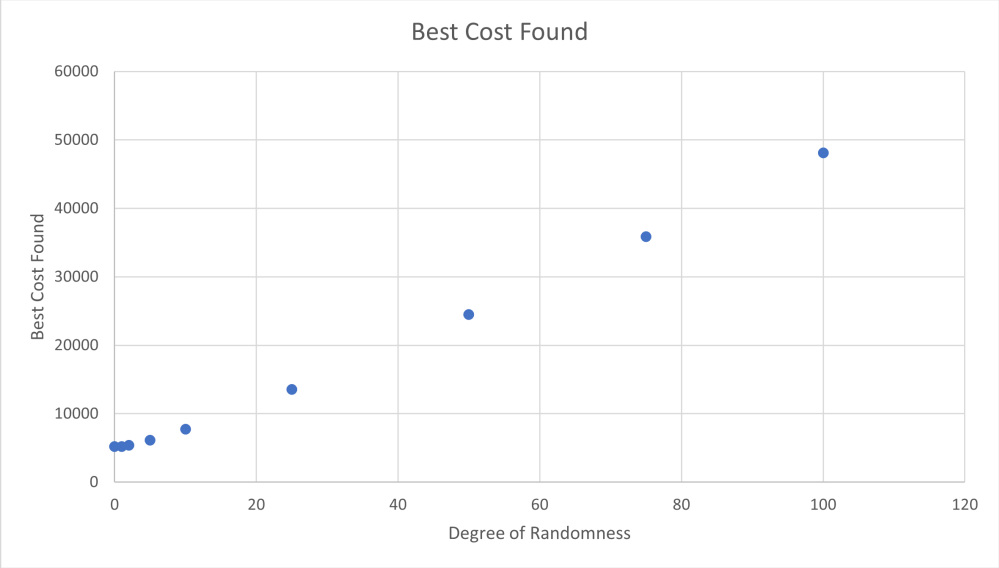

Figure 5: Best cost paths by degree of randomness.

Analysis of results:

This implementation revealed several interesting things about the problem, including the effects of randomness upon both the quality of answer and the quantity of paths checked by the process.

The results showed that the degree of randomness greatly affected the number of paths that the process could explore. The 'rand()' function in our C implementation was slow compared to the heuristic check, so running a higher degree of randomness (leaning heavily on the 'rand()' function rather than the heuristic) decreased the number of paths the process could check within the time constraints.

Figure 5 uses the average among the individual processes on each machine to measure this value. The next thing we found was that the amount of randomness affected the quality of path found during the runtime. This actually shows that a large amount of randomness was harmful to the answer quality the process found, at anything above ODDS = 5%. At and below ODDS = 5%, the answers were very close to the non-random value. To explore this, we ran an experiment with large randomness but more time, to make up for the time issue that 'rand()' applies to the process. We did this by setting the ODDS to 90%, then increasing the time proportional to the gap in the number of paths checked. This resulted in multiplying the time constraint by 4 to attempt to 'even the odds' for the rand() function. By doing this, we were able to get the Average paths checked per process to 145,000, but the best cost path for the process was still at 43,080, more than 8 times the low randomness answer. In conclusion, the Nearest Neighbor heuristic for the Traveling Salesman problem does not benefit from an aspect of randomness, but actually suffers to find a good path under the added randomness.

Scalability:

This implementation seems weekly scalable because there seems to be no indication that the efficiency would drastically increase with more threads.

Stochastic Nearest Neighbor - Source Code:

```
/* Traveling Salesman Problem
 * - Implements basic Heuristics as a search method, with added randomness
 * - Multiple Nodes with multiple threads
 *    - Nodes handle different starting cities
 *    - Threads handle different random paths from the same starting city
 * - Returns the shortest path found in 60ish seconds
 */

#include <stdio.h>
#include <time.h>
#include <stdlib.h>
#include <limits.h>
#include <mpi.h>
#include <omp.h>

#define ODDS 50
#define FILENAME "DistanceMatrix1000_v2.csv"

int main(int argc, char* argv[]){
  // MPI Initializing
  MPI_Init(NULL, NULL);
    int commSz;
    MPI_Comm_size(MPI_COMM_WORLD, &commSz);
  int rank;
  MPI_Comm_rank(MPI_COMM_WORLD, &rank);
    time_t start;
    srand((unsigned) time(&start));
    // Start Timer
    int (*distMat)[1000] = malloc(sizeof(int[1000][1000]));
    if(distMat==NULL){
        printf("Memory Allocation Issue: Matrix");
        exit(1);
    }
  printf("Rank: %d\n", rank);
    if(rank==0){
      // Read Input
      FILE* filePointer;
      filePointer = fopen(FILENAME,"r");
      for(int i = 0; i<1000;i++){
          for(int j=0;j<1000;j++){
              fscanf(filePointer, "%d,", &distMat[i][j]);
        //printf("Reading %d, %d = %d\r", i, j, distMat[i][j]);
          }
          fscanf(filePointer, "\n");
      }
  }
    // Search Loop
  MPI_Bcast(&(distMat[0][0]),1000*1000, MPI_INT, 0, MPI_COMM_WORLD);
  // Randomly pick a starting city
  int startCity = rand() % 1000;
  // distMat[A][B] = distance from A->B
  int* bestPath;
  int bestCost = INT_MAX;
  printf("Starting Value: %d ; Starting City: %d\n", bestCost, startCity);
    #pragma omp parallel private(rank, startCity)
    //Reduction of what exactly?? That could actually be kinda painful to figure out
  {
    time_t tim;
    int *currPath = calloc(1000, sizeof(int));
    currPath[0] = startCity;
    int currCost = 0;
    time(&tim);
    while(tim-start < 60){
      int *visited = calloc(1000, sizeof(int)); //Visited array
      int prev = startCity;
      int next;
```

```
      for(int=1;i<1000;i++){
        //Each Node on the path
        if(rand()%100<ODDS){
          // IF RANDOM
          do{
            next = rand()%1000;
          } while(visited[next]);
        } else {
          for(int j = 0;j<1000;j++){
            if(visited[j]) continue;
            printf("distMat[%d][%d] < distMat[%d][%d]\n", prev, j, prev, next);
            if(distMat[prev][j] < distMat[prev][next]){ // Nearest Neighbor
              next = j;
            }
          }
          // ELSE HEURISTIC
        }
        // Count up stats, update visited
        //printf("CurrCost: %d ; distMat[%d][%d]=%d\n", currCost, prev, next, distMat[prev][
    next]);
        fflush(stdout);
        currCost += distMat[prev][next];
        currPath[i] = next;
        visited[next] = 1;
        prev = next;
      }
      //Tack on final step
      currCost += distMat[prev][startCity];
      // Check path against our past paths
      #pragma omp critical
      {
        if(currCost<bestCost){
          bestCost = currCost;
          bestPath = currPath;
        }
      }
      time(&tim); //Time check before looking at another path
    }
  }//PARRALLEL
  //Argmin from MPI: MINLOC then SEND/RECV path to root
  printf("%d\n", bestCost);
  //MINLOC
  // IF me == loc, send path to root
  // IF me == root, get path from loc
  // Report output
  MPI_Finalize();
  return 0;
}
```

2. Ant Colony Optimization

Algorithm Description:

Our alternative implementation to solve the traveling sales person problem is based off of ant colony optimization. The idea behind this algorithm was to run μ agents. Each agent has access to a randomly generated pheromone table for referencing the current city r and the next potential city u using $\tau[r][u]$. Then, the μth agent would pick the best unvisited city that based on highest probability to transitioning. Rigidly, the agent attempts to a pick a next city u based on the following equation.

$$\arg\max_{u \in J^{\mu}} [\tau[r][u] \times \eta[r][u]^{\beta}] \tag{3}$$

or the μth agent selects a city with the probability.

$$P_\mu(r,u) = \begin{cases} \frac{\tau(r,u)\eta(r,u)^\beta}{\Sigma_{s\in J^\mu}\tau(r,s)\eta(r,s)^\beta} & \text{if } u \in J^\mu \\ 0 & \text{if } u \notin J^\mu \end{cases} \tag{4}$$

where J^μ is the set of all unvisited cities by the μth agent and β is a positive constant. Each agent learns information through local and global pheromone updates. Local updates, represented in the following equation, are calculated immediately after the best city has been selected by the μth agent.

$$\tau(r,u) = (1-\rho)\cdot\tau(r,u) + \frac{\rho}{N\cdot L_{GB}} \tag{5}$$

where ρ is a evaporation parameter, $0 < \rho < 1$, N is the number of cities, and L_{GB} is the length of the best global tour.

Global updates are calculated after every agent has selected a tour. The pheromone updates are detailed in the following equations.

$$\tau(r,u) = (1-\rho)\cdot\tau(r,u) + \rho\cdot\Delta\tau_\mu(r,u) \tag{6}$$

$$\Delta\tau_\mu(r,u) = \begin{cases} \frac{Q}{L_{GB}} & \text{if } (r,u) \in \text{global best tour} \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

(where Q is a non-positive constant and L_{GB} is the length of the best global tour)

This problem was modelled in the following pseudocode. The implementation has every agent generate a series of solutions and update the pheromone table based on the best observed tour. The basic outline of ant colony solution is

```
        while iterations < iterations_max:
            for i in num_agents:
                generateSolutions()
            globalPheromoneUpdate()
        repeat

```

Generating solutions involved the following steps.

```
        let local_tour[n] and local_length = 0
        while agent still has cities to visit:
            calculate the best probability using equation (4)
            add the best city to local_tour[n].
            update pheromone table with current_city, best_city, and equation (5)
            compare and/or update best global tour using local_tour[n]
        repeat

```

Global updates to the pheromone table were made in the following steps.

```
        for each (r, u) pairing:
            check if (r, u) is in best global tour using equation (7).
            if (r, u) is in global tour:
                update t[r][u] using equation (6)
            else:
                move to next pairing


```

Libraries used:

The omp.h (OpenMP) library was used in this implementation. The function omp_get_wtime() was used to time the program.

Output from tests: [Note: The outputs are long, so see the Appendix for all outputs]

Results from testing:

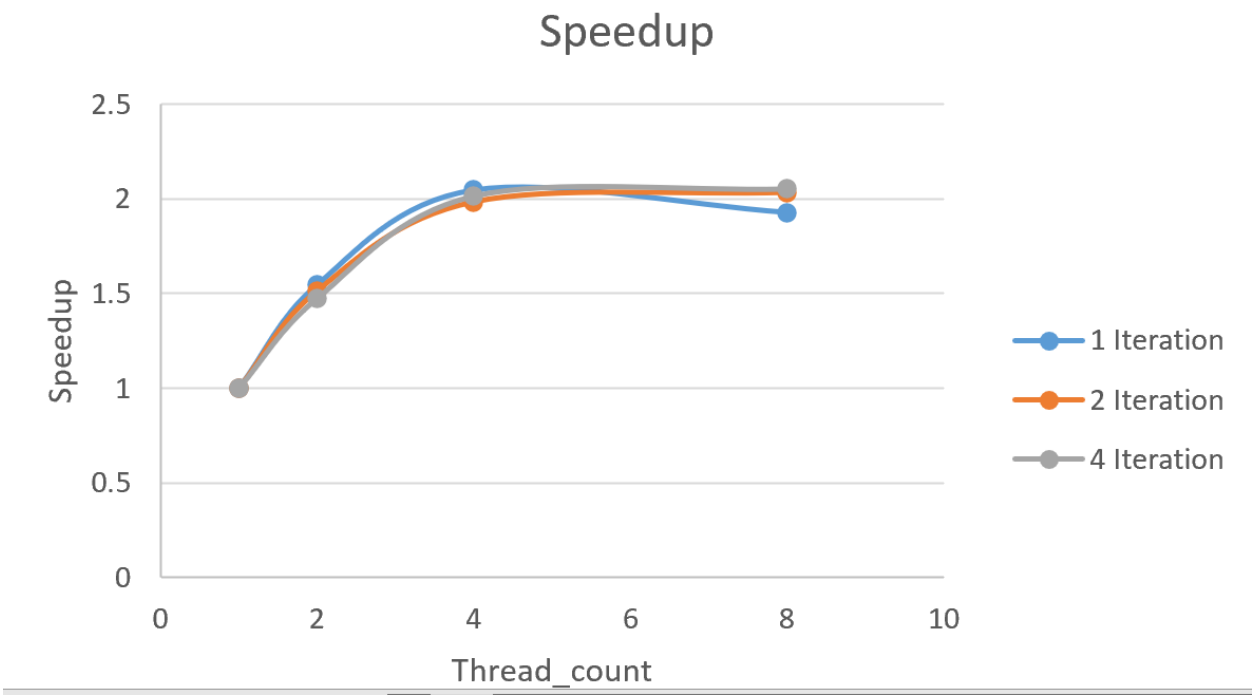

Figure 6: Graph of the speedup that occurs with multiple cores. [Ant Colony]

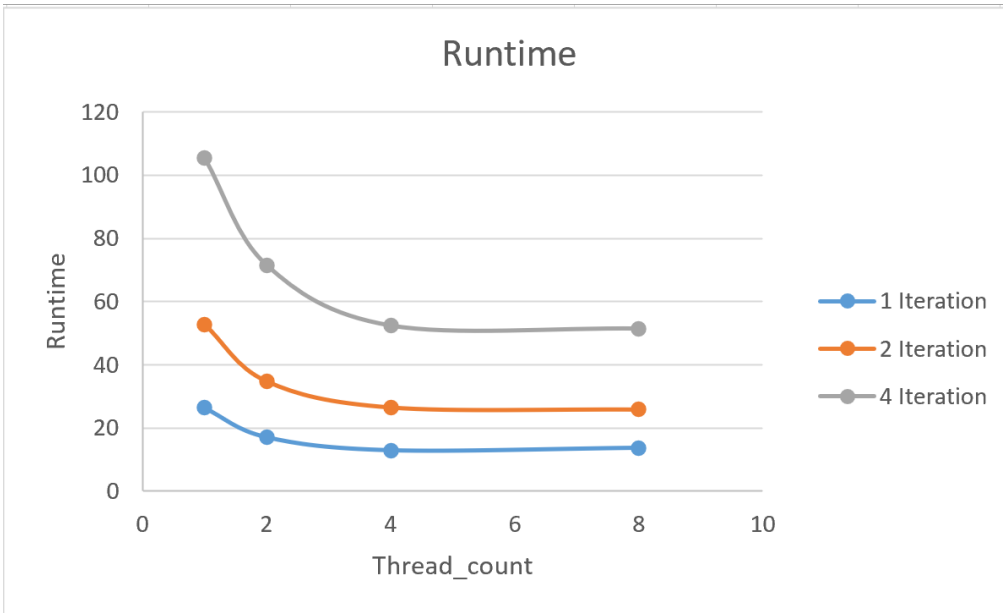

Figure 7: Graph of the runtime for various thread counts. [Ant Colony]

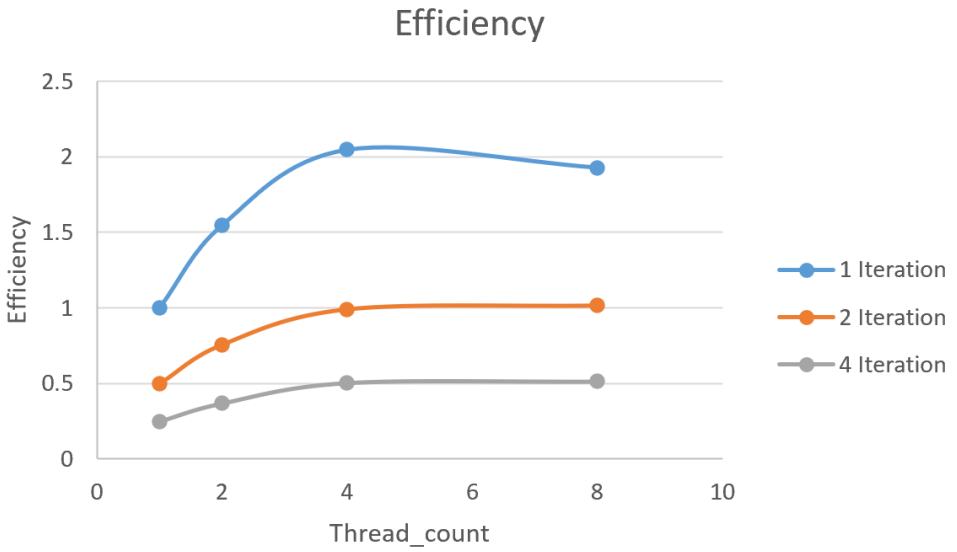

Figure 8: Graph of the efficiency for various thread counts. [Ant Colony]

Analysis of results and scalability:

Based on the concave trends of the efficiency, there is no evidence that this program is strongly-scalable. Therefore, there is weak scalability in this program.

Ant Colony - Source Code:

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <math.h>
#include <limits.h>
#include <omp.h>

#define SIZE 1000
void read_matrix(double graph[SIZE][SIZE]);
void initialize_visited_cities(int* list_cities);
void initialize_transitions(double* transitions[SIZE]);
double heuristic_function(double graph[SIZE][SIZE], int curr_city, int next_city, double
    beta);

void generate_solutions(double graph[SIZE][SIZE], double* transitions[SIZE], int
    starting_city);
void global_update(double* transitions[SIZE], int current_city, int next_city);
void local_update(double* transitions[SIZE], int current_city, int next_city);
void update_transitions(double* transitions[SIZE]);
int pairing_global_tour(int current_city, int next_city);

int num_agents;
int thread_count;
double phi;
double beta;
double Q;

int *best_tour;
int best_length;

int main(int argc, char** argv) {
  //printf("Testing test\n");
  thread_count = strtol(argv[1], NULL, 10);
  num_agents = 1;
  phi = 0.5;
  beta = 2;
  Q = 5.00;

  int iterations_max = 1;


  double graph[SIZE][SIZE];
  read_matrix(graph);
  int i, j;

  double* transitions[SIZE];
  for (i = 0; i < SIZE; i++) {
    transitions[i] = (double*) malloc(SIZE*sizeof(double));
  }

    best_tour = malloc((SIZE+1)*sizeof(int));
  best_length = INT_MAX;

  initialize_transitions(transitions);
  double start = omp_get_wtime();

  int iteration = 0;
  while (iteration < iterations_max) {
    for (i = 0; i < num_agents; i++) {
      int starting_city;
      for (starting_city = 0; starting_city < SIZE; starting_city++) {
        generate_solutions(graph, transitions, starting_city);
      }
    }

    update_transitions(transitions);
```

```
65      iteration++;
66    }
67
68    printf("Length: %d\n", best_length);
69    for (i = 0; i < SIZE+1; i++) {
70      if (i % 20 == 0) {
71        printf("\n");
72      }
73      printf("%d ->", best_tour[i]);
74    }
75    printf("\nTime: %1.16f\n\n", omp_get_wtime() - start);
76
77    return 0;
78  }
79
80  void generate_solutions(double graph[SIZE][SIZE], double* transitions[SIZE], int
        starting_city) {
81    int current_city = starting_city;
82    int remaining_cities = SIZE-1;
83
84    //If visited[i] == -1, it indicates that the ith has been visited.
85    int* visited_cities = malloc(SIZE*sizeof(int));
86    initialize_visited_cities(visited_cities);
87    visited_cities[starting_city] = -1;
88
89
90    int *agent_tour = malloc((SIZE+1)*sizeof(int));
91    agent_tour[0] = starting_city;
92    int length = 0;
93    int k = 1;
94    //printf("agent_tour_start %d\n", starting_city);
95    while (remaining_cities > 0) {
96
97      int i;
98
99      int best_city = -5;
100     double best_prob = -5.00;
101     double total_prob = 0.00;
102
103     //Calculate the total probability for all legal cities in visited_cities
104     #pragma omp parallel for num_threads(thread_count) private(i) reduction(+: total_prob)
105     for (i = 0; i < SIZE; i++) {
106       if (visited_cities[i] != -1 && i != current_city) {
107         total_prob += (double)(transitions[current_city][i]*heuristic_function(graph,
        current_city, i, beta));
108       }
109     }
110
111     //Calculate the local probability.
112
113     #pragma omp parallel for num_threads(thread_count) private(i) reduction(max: best_prob)
114     for (i = 0; i < SIZE; i++) {
115       if (visited_cities[i] != -1 && i != current_city) {
116         double local_prob = (double)(transitions[current_city][i]*heuristic_function(graph,
        current_city, i, beta)) / total_prob;
117         if (local_prob > best_prob) {
118           best_city = i;
119         }
120         best_prob = fmax(local_prob, best_prob);
121       }
122     }
123
124     visited_cities[best_city] = -1;
125     length += graph[current_city][best_city];
126     //local update for transitions.
127     local_update(transitions, current_city, best_city);
128
129     current_city = best_city;
```

```
130     agent_tour[k] = best_city;
131
132     k++;
133     remaining_cities--;
134   }
135
136   length += graph[current_city][starting_city];
137   agent_tour[SIZE] = starting_city;
138
139   // for (k = 0; k < SIZE+1; k++) {
140   //  printf("%d ", agent_tour[k]);
141   // }
142   // printf("\n%d\n", length);
143
144   if (length < best_length) {
145     best_length = length;
146
147     #pragma omp parallel for num_threads(thread_count) private(k)
148     for (k = 0; k < SIZE+1;k++) {
149       best_tour[k]=agent_tour[k];
150     }
151   }
152
153   free(agent_tour);
154   free(visited_cities);
155 }
156
157 void local_update(double* transitions[SIZE], int current_city, int next_city) {
158   //printf("Updating pheromone from %d to %d\n", current_city, next_city);
159   transitions[current_city][next_city] = (1-phi)*transitions[current_city][next_city] + phi
        /(SIZE*best_length);
160 }
161
162
163 void update_transitions(double* transitions[SIZE]) {
164   int i, j;
165
166   #pragma omp parallel for num_threads(thread_count) private(i, j)
167   for (i = 0; i < SIZE; i++) {
168     for (j = 0; j < SIZE; j++) {
169       double change_transition = 0.00;
170       int test = pairing_global_tour(i, j);
171
172       if (test == 1) {
173         change_transition = (double) (Q) / (double) (best_length);
174       }
175       //printf("Change happening %d", test);
176       //printf("Change in transition[%d][%d]: %1f\n ", i, j, change_transition);
177       transitions[i][j] = ((1 - phi)*transitions[i][j]) + (phi*change_transition);
178     }
179   }
180 }
181
182 int pairing_global_tour(int current_city, int next_city) {
183   int result = 0;
184   for (int i = 0; i <= (SIZE-1); i++) {
185     if (best_tour[i] == current_city && best_tour[i+1] == next_city) {
186       result = 1;
187       break;
188     }
189   }
190   return result;
191 }
192
193
194
195 void read_matrix(double graph[SIZE][SIZE]) {
196   char f_name[55];
```

```
197
198     //Create filename
199
200     //sprintf(f_name,"/home/lar9482/Final_Project_2/DistanceMatrix%d_v2.csv", SIZE);
201     sprintf(f_name,"DistanceMatrix%d_v2.csv", SIZE);
202     //printf("Reading array file %s of size %dx%d\n",f_name,SIZE,SIZE);
203     //Open file
204     FILE *datafile=fopen(f_name,"rb");
205
206
207     //Read elements
208     if (!datafile) {
209       printf("Can't open file\n");
210     }
211     else {
212       //printf("Reading a file now\n");
213       int i,j = 0;
214       char buffer[SIZE*SIZE];
215       while (fgets(buffer, SIZE*SIZE, datafile)) {
216
217         j = 0;
218         char *value  = strtok(buffer, ", ");
219         while (value) {
220           double num = strtod(value, NULL);
221           graph[i][j] = num;
222           //printf("%1f\n", num);
223           value = strtok(NULL, ", ");
224           j++;
225         }
226         i++;
227
228         free(value);
229       }
230     }
231
232     //printf("Matrix has been read.\n");
233     return;
234 }
235
236 void initialize_visited_cities(int* list_cities) {
237   int i;
238   for (i = 0; i < SIZE; i++) {
239     list_cities[i] = i;
240   }
241   return;
242 }
243
244 void initialize_transitions(double* transitions[SIZE]) {
245   int i, j;
246   //srand(5);
247   for (i = 0; i < SIZE; i++) {
248     for (j = 0; j < SIZE; j++) {
249       //printf("%1f", (double) ((double)rand() / (double)RAND_MAX));
250       transitions[i][j] = (double) ((double)rand() / (double)RAND_MAX);
251     }
252   }
253   //printf("finished transitions\n");
254 }
255
256 double heuristic_function(double graph[SIZE][SIZE], int curr_city, int next_city, double
      beta) {
257   return (double) pow((double) ( ((double) 1.00) / ((double) graph[curr_city][next_city]) ),
       beta);
258 }
```

Team contributions

Luke Runnels

Luke implemented the OpenMP programs for LU decomposition for and ant colony optimization for the determinant problem and Traveling Salesperson Problem respectively. He helped research different techniques to solve both problems. He also helped some of the write the report, and edit the presentation.

Austin Lively

Austin implemented the stochastic nearest neighbor algorithm for the Traveling Salesperson Problem, and managed to use MPI and OpenMP in tandem for the implementation. He also made the presentation and helped write some of the report.

David Fantin

David worked on taking Luke's LU decomposition program for the determinant problem and implementing both MPI and OpenMP with it (however, that code was unused). He also helped put together most of the report.

Appendix

Determinant Problem - all outputs

Size%3A 1000

```
1 threads
File: /home/lar9482/Final_Project_1/m1000x1000.bin
Size: 1000
Det: +inf
Log Det: 743.122063
Time: 0.8552089259028435

2 threads
File: /home/lar9482/Final_Project_1/m1000x1000.bin
Size: 1000
Det: +inf
Log Det: 743.122063
Time: 0.4532068748958409

4 threads
File: /home/lar9482/Final_Project_1/m1000x1000.bin
Size: 1000
Det: +inf
Log Det: 743.122063
Time: 0.2316409051418304

8 threads
File: /home/lar9482/Final_Project_1/m1000x1000.bin
Size: 1000
Det: +inf
Log Det: 743.122063
Time: 0.1220189523883164

16 threads
File: /home/lar9482/Final_Project_1/m1000x1000.bin
Size: 1000
Det: +inf
Log Det: 743.122063
Time: 0.0712027358822525

32 threads
File: /home/lar9482/Final_Project_1/m1000x1000.bin
Size: 1000
Det: +inf
Log Det: 743.122063
Time: 1.1239543850533664
```

Size%3A 1024

```
1 threads
File: /home/lar9482/Final_Project_1/m1024x1024.bin
Size: 1024
Det: +inf
Log Det: 766.734112
Time: 0.9158545862883329

2 threads
File: /home/lar9482/Final_Project_1/m1024x1024.bin
Size: 1024
Det: +inf
Log Det: 766.734112
Time: 0.4892211970873177

4 threads
File: /home/lar9482/Final_Project_1/m1024x1024.bin
Size: 1024
```

```
Det: +inf
Log Det: 766.734112
Time: 0.2505814270116389

8 threads
File: /home/lar9482/Final_Project_1/m1024x1024.bin
Size: 1024
Det: +inf
Log Det: 766.734112
Time: 0.1267781811766326

16 threads
File: /home/lar9482/Final_Project_1/m1024x1024.bin
Size: 1024
Det: +inf
Log Det: 766.734112
Time: 0.0765595729462802

32 threads
File: /home/lar9482/Final_Project_1/m1024x1024.bin
Size: 1024
Det: +inf
Log Det: 766.734112
Time: 0.1239477191120386
```

Size%3A 128

```
1 threads
File: /home/lar9482/Final_Project_1/m0128x0128.bin
Size: 128
Det: -372588862239759235716422166852654858240.000000
Log Det: 38.571230
Time: 0.0019422979094088

2 threads
File: /home/lar9482/Final_Project_1/m0128x0128.bin
Size: 128
Det: -372588862239759235716422166852654858240.000000
Log Det: 38.571230
Time: 0.0016236300580204

4 threads
File: /home/lar9482/Final_Project_1/m0128x0128.bin
Size: 128
Det: -372588862239759160158558440938331439104.000000
Log Det: 38.571230
Time: 0.0013820840977132

8 threads
File: /home/lar9482/Final_Project_1/m0128x0128.bin
Size: 128
Det: -372588862239759084600694715024008019968.000000
Log Det: 38.571230
Time: 0.0013924879021943

16 threads
File: /home/lar9482/Final_Project_1/m0128x0128.bin
Size: 128
Det: -372588862239759311274285892766978277376.000000
Log Det: 38.571230
Time: 0.0017858017235994

32 threads
File: /home/lar9482/Final_Project_1/m0128x0128.bin
Size: 128
Det: -372588862239759084600694715024008019968.000000
Log Det: 38.571230
Time: 0.5805477849207819
```

Size%3A 16

```
1 threads
File: /home/lar9482/Final_Project_1/m0016x0016.bin
Size: 16
Det: 0.000335
Log Det: -3.475405
Time: 0.0000417362898588

2 threads
File: /home/lar9482/Final_Project_1/m0016x0016.bin
Size: 16
Det: 0.000335
Log Det: -3.475405
Time: 0.0001590480096638

4 threads
File: /home/lar9482/Final_Project_1/m0016x0016.bin
Size: 16
Det: 0.000335
Log Det: -3.475405
Time: 0.0001982566900551

8 threads
File: /home/lar9482/Final_Project_1/m0016x0016.bin
Size: 16
Det: 0.000335
Log Det: -3.475405
Time: 0.0003012767992914

16 threads
File: /home/lar9482/Final_Project_1/m0016x0016.bin
Size: 16
Det: 0.000335
Log Det: -3.475405
Time: 0.0004848632961512

32 threads
File: /home/lar9482/Final_Project_1/m0016x0016.bin
Size: 16
Det: 0.000335
Log Det: -3.475405
Time: 0.0224974402226508
```

Size%3A 2000

```
1 threads
File: /home/lar9482/Final_Project_1/m2000x2000.bin
Size: 2000
Det: +inf
Log Det: 1786.384632
Time: 7.1684080380946398

2 threads
File: /home/lar9482/Final_Project_1/m2000x2000.bin
Size: 2000
Det: +inf
Log Det: 1786.384632
Time: 3.4646525667048991

4 threads
File: /home/lar9482/Final_Project_1/m2000x2000.bin
Size: 2000
Det: +inf
Log Det: 1786.384632
Time: 1.7934232107363641

8 threads
File: /home/lar9482/Final_Project_1/m2000x2000.bin
```

```
Size: 2000
Det: +inf
Log Det: 1786.384632
Time: 0.9142539231106639

16 threads
File: /home/lar9482/Final_Project_1/m2000x2000.bin
Size: 2000
Det: +inf
Log Det: 1786.384632
Time: 0.4889733823947608

32 threads
File: /home/lar9482/Final_Project_1/m2000x2000.bin
Size: 2000
Det: +inf
Log Det: 1786.384632
Time: 6.5275183026678860
```

Size%3A 2048

```
1 threads
File: /home/lar9482/Final_Project_1/m2048x2048.bin
Size: 2048
Det: +inf
Log Det: 1838.927339
Time: 7.7259128112345934

2 threads
File: /home/lar9482/Final_Project_1/m2048x2048.bin
Size: 2048
Det: +inf
Log Det: 1838.927339
Time: 3.7380560617893934

4 threads
File: /home/lar9482/Final_Project_1/m2048x2048.bin
Size: 2048
Det: +inf
Log Det: 1838.927339
Time: 1.8843039511702955

8 threads
File: /home/lar9482/Final_Project_1/m2048x2048.bin
Size: 2048
Det: +inf
Log Det: 1838.927339
Time: 0.9566240622662008

16 threads
File: /home/lar9482/Final_Project_1/m2048x2048.bin
Size: 2048
Det: +inf
Log Det: 1838.927339
Time: 0.5248457016423345

32 threads
File: /home/lar9482/Final_Project_1/m2048x2048.bin
Size: 2048
Det: +inf
Log Det: 1838.927339
Time: 3.0181718836538494
```

Size%3A 256

```
1 threads
File: /home/lar9482/Final_Project_1/m0256x0256.bin
Size: 256
```

```
Det:
    -125432077551451954328580267560162592140737090034509817260784699619488480376041124208664776119921855060
Log Det: 115.098409
Time: 0.0146755273453891

2 threads
File: /home/lar9482/Final_Project_1/m0256x0256.bin
Size: 256
Det:
    -125432077551451932456073019730043348415714818858296163729090390687364116118335060109134784126162622830
Log Det: 115.098409
Time: 0.0089475261047482

4 threads
File: /home/lar9482/Final_Project_1/m0256x0256.bin
Size: 256
Det:
    -125432077551452063691116506710758810765848445915578084919256244280110301664571444706314736088718016240
Log Det: 115.098409
Time: 0.0052077649161220

8 threads
File: /home/lar9482/Final_Project_1/m0256x0256.bin
Size: 256
Det:
    -125432077551451976201087515390281835865759361210723470792479008551612844633747188308194768113681087300
Log Det: 115.098409
Time: 0.0040686563588679

16 threads
File: /home/lar9482/Final_Project_1/m0256x0256.bin
Size: 256
Det:
    -125432077551452041818609258880639567040826174739364431387561935347985937406865380606784744094958784000
Log Det: 115.098409
Time: 0.0038318368606269

32 threads
File: /home/lar9482/Final_Project_1/m0256x0256.bin
Size: 256
Det:
    -125432077551452019946102011050520323315803903563150777855867626415861573149159316507254752101199551770
Log Det: 115.098409
Time: 0.9918027860112488
```

Size%3A 3000

```
1 threads
File: /home/lar9482/Final_Project_1/m3000x3000.bin
Size: 3000
Det: +inf
Log Det: 2945.780634
Time: 25.0743106701411307

2 threads
File: /home/lar9482/Final_Project_1/m3000x3000.bin
Size: 3000
Det: +inf
Log Det: 2945.780634
Time: 13.6584573322907090

4 threads
```

```
File: /home/lar9482/Final_Project_1/m3000x3000.bin
Size: 3000
Det: +inf
Log Det: 2945.780634
Time: 9.0624369839206338

8 threads
File: /home/lar9482/Final_Project_1/m3000x3000.bin
Size: 3000
Det: +inf
Log Det: 2945.780634
Time: 7.3657856858335435

16 threads
File: /home/lar9482/Final_Project_1/m3000x3000.bin
Size: 3000
Det: +inf
Log Det: 2945.780634
Time: 6.7294900962151587

32 threads
File: /home/lar9482/Final_Project_1/m3000x3000.bin
Size: 3000
Det: +inf
Log Det: 2945.780634
Time: 7.1467988681979477
```

Size%3A 32

```
1 threads
File: /home/lar9482/Final_Project_1/m0032x0032.bin
Size: 32
Det: -0.263326
Log Det: -0.579507
Time: 0.0000671721063554

2 threads
File: /home/lar9482/Final_Project_1/m0032x0032.bin
Size: 32
Det: -0.263326
Log Det: -0.579507
Time: 0.0002309870906174

4 threads
File: /home/lar9482/Final_Project_1/m0032x0032.bin
Size: 32
Det: -0.263326
Log Det: -0.579507
Time: 0.0002843323163688

8 threads
File: /home/lar9482/Final_Project_1/m0032x0032.bin
Size: 32
Det: -0.263326
Log Det: -0.579507
Time: 0.0003536217845976

16 threads
File: /home/lar9482/Final_Project_1/m0032x0032.bin
Size: 32
Det: -0.263326
Log Det: -0.579507
Time: 0.0006395578384399

32 threads
File: /home/lar9482/Final_Project_1/m0032x0032.bin
Size: 32
Det: -0.263326
```

```
Log Det: -0.579507
Time: 0.0226851953193545
```

Size%3A 4000

```
1 threads
File: /home/lar9482/Final_Project_1/m4000x4000.bin
Size: 4000
Det: +inf
Log Det: 4176.335020
Time: 59.3883682219311595

2 threads
File: /home/lar9482/Final_Project_1/m4000x4000.bin
Size: 4000
Det: +inf
Log Det: 4176.335020
Time: 34.2185549330897629

4 threads
File: /home/lar9482/Final_Project_1/m4000x4000.bin
Size: 4000
Det: +inf
Log Det: 4176.335020
Time: 26.3845605687238276

8 threads
File: /home/lar9482/Final_Project_1/m4000x4000.bin
Size: 4000
Det: +inf
Log Det: 4176.335020
Time: 23.5257420139387250

16 threads
File: /home/lar9482/Final_Project_1/m4000x4000.bin
Size: 4000
Det: +inf
Log Det: 4176.335020
Time: 23.6734951878897846

32 threads
File: /home/lar9482/Final_Project_1/m4000x4000.bin
Size: 4000
Det: +inf
Log Det: 4176.335020
Time: 23.9503137781284750
```

Size%3A 4096

```
1 threads
File: /home/lar9482/Final_Project_1/m4096x4096.bin
Size: 4096
Det: +inf
Log Det: 4297.334290
Time: 63.9182369438931346

2 threads
File: /home/lar9482/Final_Project_1/m4096x4096.bin
Size: 4096
Det: +inf
Log Det: 4297.334290
Time: 36.8999505229294300

4 threads
File: /home/lar9482/Final_Project_1/m4096x4096.bin
Size: 4096
Det: +inf
Log Det: 4297.334290
Time: 27.8421836658380926
```

```
8 threads
File: /home/lar9482/Final_Project_1/m4096x4096.bin
Size: 4096
Det: +inf
Log Det: 4297.334290
Time: 25.5887725781649351

16 threads
File: /home/lar9482/Final_Project_1/m4096x4096.bin
Size: 4096
Det: +inf
Log Det: 4297.334290
Time: 25.7578369048424065

32 threads
File: /home/lar9482/Final_Project_1/m4096x4096.bin
Size: 4096
Det: +inf
Log Det: 4297.334290
Time: 27.0827914997935295
```

Size%3A 496

```
1 threads
File: /home/lar9482/Final_Project_1/m0496x0496.bin
Size: 496
Det:
    594806771494169358889992703411261467461087713607977414468993934268449142931461453281424761179694968365

Log Det: 292.774376
Time: 0.1050216848962009

2 threads
File: /home/lar9482/Final_Project_1/m0496x0496.bin
Size: 496
Det:
    594806771494169536156221913046884302389421252366798917543187132877140722722990550354581256322199367804

Log Det: 292.774376
Time: 0.0576045550405979

4 threads
File: /home/lar9482/Final_Project_1/m0496x0496.bin
Size: 496
Det:
    594806771494169270256878098593450049996920944228566662931897334964103353035696904744846513608442768646

Log Det: 292.774376
Time: 0.0305056408978999

8 threads
File: /home/lar9482/Final_Project_1/m0496x0496.bin
Size: 496
Det:
    594806771494168649825075864868770127747753558572691402172221139833682823765345064988798780609677370610

Log Det: 292.774376
Time: 0.0172814643010497

16 threads
File: /home/lar9482/Final_Project_1/m0496x0496.bin
Size: 496
Det:
    594806771494168649825075864868770127747753558572691402172221139833682823765345064988798780609677370610

Log Det: 292.774376
Time: 0.0124078299850225
```

```

32 threads
File: /home/lar9482/Final_Project_1/m0496x0496.bin
Size: 496
Det:
    594806771494169979321794937135941389710255099263852675228670129398869672201813293037472494178460366401

Log Det: 292.774376
Time: 0.0265024350956082
```

Size%3A 512

```
1 threads
File: /home/lar9482/Final_Project_1/m0512x0512.bin
Size: 512
Det:
    -190892165734141680938631963126632963645675931583056933590367238992277544972860047548200213748846473820

Log Det: 304.280788
Time: 0.1156383547931910

2 threads
File: /home/lar9482/Final_Project_1/m0512x0512.bin
Size: 512
Det:
    -190892165734141729665202020126628365412587870958998487971501526967315179312392333612943667580976986150

Log Det: 304.280788
Time: 0.0639014630578458

4 threads
File: /home/lar9482/Final_Project_1/m0512x0512.bin
Size: 512
Det:
    -190892165734141851481627162626616869829867719398852373924337246904909265161223048774802302161303266970

Log Det: 304.280788
Time: 0.0337760969996452

8 threads
File: /home/lar9482/Final_Project_1/m0512x0512.bin
Size: 512
Det:
    -190892165734141754028487048626626066296043840646969265162068670954833996482158476645315394497042242310

Log Det: 304.280788
Time: 0.0190623099915683

16 threads
File: /home/lar9482/Final_Project_1/m0512x0512.bin
Size: 512
Det:
    -190892165734141754028487048626626066296043840646969265162068670954833996482158476645315394497042242310

Log Det: 304.280788
Time: 0.0131535837426782

32 threads
File: /home/lar9482/Final_Project_1/m0512x0512.bin
Size: 512
Det:
    -190892165734141632212061906126637561878763992207115379209232951017239910633327761483456759916715961490

Log Det: 304.280788
Time: 2.1167190782725811
```

Size%3A 64

```
1 threads
```

```
File: /home/lar9482/Final_Project_1/m0064x0064.bin
Size: 64
Det: 37889891.273514
Log Det: 7.578523
Time: 0.0002983249723911

2 threads
File: /home/lar9482/Final_Project_1/m0064x0064.bin
Size: 64
Det: 37889891.273514
Log Det: 7.578523
Time: 0.0004452797584236

4 threads
File: /home/lar9482/Final_Project_1/m0064x0064.bin
Size: 64
Det: 37889891.273514
Log Det: 7.578523
Time: 0.0004620826803148

8 threads
File: /home/lar9482/Final_Project_1/m0064x0064.bin
Size: 64
Det: 37889891.273514
Log Det: 7.578523
Time: 0.0005399738438427

16 threads
File: /home/lar9482/Final_Project_1/m0064x0064.bin
Size: 64
Det: 37889891.273514
Log Det: 7.578523
Time: 0.0008668368682265

32 threads
File: /home/lar9482/Final_Project_1/m0064x0064.bin
Size: 64
Det: 37889891.273514
Log Det: 7.578523
Time: 0.0186105351895094
```

Traveling Salesperson Problem - all outputs

1. Stochastic Nearest Neighbor

job.34117.out

```
Rank: 0
Rank: 1
Rank: 2
Rank: 3
Rank: 6
Rank: 5
Rank: 7
Rank: 4
[Process 6] Searched 31752 paths and found best path 48491
[Process 2] Searched 32269 paths and found best path 48532
[Process 0] Searched 32908 paths and found best path 48481
[Process 1] Searched 32111 paths and found best path 48122
[Process 3] Searched 32473 paths and found best path 48753
[Process 7] Searched 32803 paths and found best path 48324
[Process 5] Searched 32772 paths and found best path 48621
[Process 4] Searched 33356 paths and found best path 48548
[Process 1] Entering send mode...
Reducton Result: [0] 48122 [1] 1
```

```
[Process 0] Entering recv mode...
BestPathCost: 48122
56 400 598 354 892 68 388 792 327 572
```

job.34118.out

```
[Process 3] Searched 40927 paths and found best path 36064
[Process 0] Searched 41363 paths and found best path 35868
[Process 2] Searched 41491 paths and found best path 36163
[Process 1] Searched 41971 paths and found best path 36347
[Process 6] Searched 41024 paths and found best path 36445
[Process 7] Searched 41267 paths and found best path 36611
[Process 5] Searched 41731 paths and found best path 36430
[Process 4] Searched 41673 paths and found best path 36137
Reducton Result: [0] 35868 [1] 0
Best Path Found Cost: 35868
142 779 421 474 799 898 147 984 927 98
```

job.34119.out

```
[Process 1] Searched 53491 paths and found best path 24815
[Process 0] Searched 53567 paths and found best path 24553
[Process 4] Searched 53548 paths and found best path 24668
[Process 6] Searched 54271 paths and found best path 24522
[Process 7] Searched 53692 paths and found best path 24483
[Process 3] Searched 53520 paths and found best path 24665
[Process 2] Searched 53852 paths and found best path 24904
[Process 5] Searched 53467 paths and found best path 24960
Reducton Result: [0] 24483 [1] 7
[Process 0] Entering recv mode...
[Process 7] Entering send mode...
BestPathCost: 24483
427 2 998 926 59 320 903 111 42 985
```

job.34121.out

```
[Process 1] Searched 79418 paths and found best path 13897
[Process 6] Searched 79711 paths and found best path 13574
[Process 7] Searched 78993 paths and found best path 13667
[Process 0] Searched 79643 paths and found best path 13871
[Process 2] Searched 79703 paths and found best path 13932
[Process 3] Searched 79169 paths and found best path 13684
[Process 5] Searched 79609 paths and found best path 13829
[Process 4] Searched 79935 paths and found best path 13750
Reducton Result: [0] 13574 [1] 6
[Process 0] Entering recv mode...
[Process 6] Entering send mode...
BestPathCost: 13574
694 33 73 420 69 750 383 29 16 123
```

job.34122.out

```
[Process 0] Searched 113655 paths and found best path 8006
[Process 6] Searched 113360 paths and found best path 7927
[Process 7] Searched 113016 paths and found best path 7886
[Process 1] Searched 111123 paths and found best path 7930
[Process 2] Searched 113405 paths and found best path 7731
[Process 3] Searched 112385 paths and found best path 7962
[Process 5] Searched 112865 paths and found best path 7835
[Process 4] Searched 112703 paths and found best path 7862
Reducton Result: [0] 7731 [1] 2
[Process 0] Entering recv mode...
BestPathCost: 7731
2 171 21 195 43 101 24 64 28 87
[Process 2] Entering send mode...
```

job.34123.out

```
[Process 0] Searched 129235 paths and found best path 6218
[Process 1] Searched 129463 paths and found best path 6203
[Process 6] Searched 129579 paths and found best path 6212
[Process 7] Searched 128852 paths and found best path 6191
[Process 2] Searched 129206 paths and found best path 6226
[Process 3] Searched 128647 paths and found best path 6191
[Process 4] Searched 129272 paths and found best path 6255
[Process 5] Searched 128966 paths and found best path 6157
[Process 5] Entering send mode...
Reducton Result: [0] 6157 [1] 5
[Process 0] Entering recv mode...
BestPathCost: 6157
470 80 65 194 105 229 72 1 131 11
```

job.34124.out

```
[Process 0] Searched 139147 paths and found best path 5460
[Process 1] Searched 138491 paths and found best path 5391
[Process 6] Searched 139416 paths and found best path 5411
[Process 7] Searched 138845 paths and found best path 5425
[Process 2] Searched 138729 paths and found best path 5359
[Process 3] Searched 139074 paths and found best path 5361
[Process 4] Searched 139243 paths and found best path 5421
[Process 5] Searched 137824 paths and found best path 5427
Reducton Result: [0] 5359 [1] 2
[Process 0] Entering recv mode...
BestPathCost: 5359
358 34 54 133 130 289 25 80 65 194
[Process 2] Entering send mode...
```

job.34125.out

```
[Process 0] Searched 141838 paths and found best path 5237
[Process 1] Searched 142225 paths and found best path 5241
[Process 6] Searched 141381 paths and found best path 5238
[Process 7] Searched 142235 paths and found best path 5227
[Process 2] Searched 141584 paths and found best path 5221
[Process 3] Searched 141248 paths and found best path 5220
[Process 4] Searched 140301 paths and found best path 5242
[Process 5] Searched 141534 paths and found best path 5260
[Process 3] Entering send mode...
Reducton Result: [0] 5220 [1] 3
[Process 0] Entering recv mode...
BestPathCost: 5220
511 170 310 94 122 50 149 171 21 195
```

job.34126.out

```
[Process 0] Searched 144144 paths and found best path 5227
[Process 1] Searched 143882 paths and found best path 5234
[Process 6] Searched 143767 paths and found best path 5265
[Process 7] Searched 144037 paths and found best path 5268
[Process 2] Searched 143926 paths and found best path 5269
[Process 3] Searched 143748 paths and found best path 5279
[Process 4] Searched 144194 paths and found best path 5221
[Process 5] Searched 144076 paths and found best path 5274
Reducton Result: [0] 5221 [1] 4
[Process 0] Entering recv mode...
BestPathCost: 5221
263 0 73 98 77 127 138 186 81 48
[Process 4] Entering send mode...
```

job.34134.out

```
[Process 6] Searched 147125 paths and found best path 43643
[Process 7] Searched 145528 paths and found best path 43588
[Process 1] Searched 146327 paths and found best path 43279
[Process 4] Searched 146170 paths and found best path 43191
[Process 5] Searched 145258 paths and found best path 43080
```

```
[Process 2] Searched 144154 paths and found best path 43495
[Process 0] Searched 146879 paths and found best path 43117
[Process 3] Searched 144559 paths and found best path 43369
[Process 5] Entering send mode...
Reducton Result: [0] 43080 [1] 5
[Process 0] Entering recv mode...
BestPathCost: 43080
227 493 603 844 994 39 816 103 85 294
```

2. Ant Colony

1 Iteration

```
Length: 5659

1 ->265 ->148 ->843 ->966 ->718 ->535 ->336 ->776 ->472 ->208 ->827 ->151 ->31 ->916 ->197
    ->359 ->915 ->846 ->476 ->
417 ->421 ->872 ->120 ->929 ->339 ->452 ->225 ->291 ->647 ->217 ->464 ->432 ->185 ->75 ->51
    ->976 ->218 ->124 ->180 ->
54 ->243 ->994 ->325 ->282 ->689 ->254 ->738 ->366 ->166 ->161 ->711 ->593 ->89 ->717 ->899
    ->302 ->534 ->855 ->938 ->
666 ->741 ->764 ->102 ->99 ->312 ->747 ->238 ->413 ->576 ->429 ->986 ->815 ->645 ->724 ->168
    ->809 ->25 ->736 ->676 ->
387 ->948 ->700 ->260 ->389 ->989 ->920 ->223 ->435 ->437 ->408 ->457 ->412 ->658 ->913
    ->162 ->353 ->816 ->540 ->346 ->
370 ->252 ->453 ->561 ->844 ->491 ->244 ->833 ->935 ->713 ->517 ->859 ->975 ->383 ->769
    ->708 ->344 ->44 ->129 ->614 ->
117 ->438 ->607 ->329 ->665 ->152 ->735 ->782 ->473 ->581 ->599 ->481 ->202 ->943 ->349
    ->267 ->489 ->264 ->562 ->471 ->
632 ->621 ->431 ->497 ->115 ->0 ->73 ->840 ->420 ->69 ->157 ->6 ->281 ->555 ->898 ->549
    ->817 ->794 ->884 ->999 ->
860 ->245 ->894 ->712 ->591 ->703 ->360 ->780 ->767 ->153 ->781 ->428 ->459 ->122 ->173
    ->951 ->227 ->753 ->221 ->692 ->
142 ->272 ->8 ->998 ->959 ->460 ->554 ->836 ->983 ->618 ->326 ->246 ->198 ->634 ->887 ->262
    ->914 ->289 ->396 ->132 ->
199 ->556 ->657 ->502 ->610 ->671 ->337 ->258 ->615 ->158 ->970 ->24 ->182 ->678 ->511 ->328
    ->533 ->490 ->695 ->373 ->
597 ->59 ->320 ->829 ->411 ->509 ->979 ->219 ->295 ->485 ->771 ->654 ->636 ->88 ->492 ->602
    ->589 ->500 ->187 ->340 ->
266 ->82 ->100 ->945 ->759 ->971 ->826 ->830 ->41 ->249 ->936 ->922 ->877 ->17 ->441 ->35
    ->10 ->477 ->150 ->280 ->
558 ->322 ->637 ->222 ->778 ->910 ->352 ->135 ->680 ->341 ->777 ->729 ->60 ->950 ->196 ->463
    ->317 ->427 ->235 ->644 ->
932 ->480 ->97 ->381 ->362 ->415 ->131 ->768 ->499 ->2 ->210 ->5 ->722 ->440 ->191 ->977
    ->145 ->789 ->171 ->292 ->
309 ->631 ->486 ->236 ->84 ->912 ->766 ->978 ->573 ->677 ->681 ->209 ->744 ->40 ->193 ->123
    ->583 ->369 ->832 ->174 ->
799 ->46 ->83 ->574 ->824 ->315 ->116 ->660 ->564 ->167 ->307 ->313 ->103 ->433 ->867 ->61
    ->586 ->806 ->363 ->284 ->
204 ->93 ->140 ->775 ->685 ->942 ->375 ->550 ->523 ->847 ->954 ->176 ->160 ->953 ->125 ->504
    ->393 ->105 ->673 ->622 ->
469 ->345 ->585 ->495 ->807 ->114 ->119 ->294 ->572 ->487 ->907 ->733 ->95 ->321 ->189 ->624
    ->896 ->895 ->358 ->34 ->
740 ->852 ->802 ->400 ->314 ->211 ->172 ->968 ->965 ->955 ->770 ->392 ->967 ->828 ->287
    ->761 ->141 ->465 ->958 ->897 ->
888 ->566 ->361 ->80 ->165 ->21 ->587 ->548 ->605 ->659 ->808 ->739 ->964 ->719 ->841 ->311
    ->752 ->571 ->625 ->969 ->
835 ->793 ->404 ->380 ->324 ->672 ->68 ->669 ->594 ->33 ->350 ->330 ->48 ->261 ->568 ->892
    ->797 ->728 ->575 ->91 ->
918 ->923 ->306 ->584 ->374 ->908 ->937 ->748 ->300 ->447 ->444 ->144 ->419 ->251 ->259 ->20
    ->228 ->720 ->611 ->494 ->
110 ->862 ->516 ->368 ->842 ->274 ->530 ->293 ->85 ->376 ->449 ->763 ->949 ->397 ->643 ->755
    ->751 ->820 ->783 ->595 ->
532 ->790 ->290 ->904 ->220 ->127 ->206 ->462 ->714 ->800 ->342 ->704 ->569 ->112 ->917
    ->445 ->963 ->687 ->347 ->906 ->
```

```
933 ->200 ->522 ->118 ->310 ->94 ->333 ->406 ->921 ->960 ->995 ->757 ->939 ->143 ->401 ->508
    ->693 ->475 ->443 ->512 ->
263 ->804 ->686 ->382 ->524 ->398 ->318 ->372 ->541 ->442 ->107 ->214 ->706 ->407 ->869
    ->170 ->623 ->467 ->183 ->616 ->
612 ->812 ->984 ->332 ->716 ->335 ->760 ->190 ->9 ->996 ->849 ->106 ->885 ->874 ->285 ->911
    ->825 ->134 ->255 ->43 ->
14 ->709 ->792 ->601 ->155 ->526 ->941 ->870 ->821 ->164 ->28 ->886 ->230 ->866 ->154 ->30
    ->81 ->656 ->505 ->875 ->
70 ->316 ->410 ->592 ->529 ->537 ->175 ->834 ->878 ->458 ->364 ->520 ->446 ->304 ->63 ->927
    ->925 ->466 ->853 ->648 ->
426 ->414 ->109 ->737 ->663 ->883 ->257 ->609 ->619 ->27 ->542 ->626 ->159 ->23 ->765 ->646
    ->365 ->696 ->186 ->104 ->
11 ->384 ->664 ->461 ->439 ->579 ->130 ->371 ->837 ->596 ->405 ->608 ->108 ->850 ->111 ->902
    ->425 ->474 ->319 ->7 ->
434 ->268 ->448 ->334 ->745 ->205 ->864 ->754 ->725 ->296 ->940 ->518 ->515 ->86 ->684 ->848
    ->698 ->546 ->980 ->796 ->
565 ->991 ->256 ->338 ->181 ->823 ->377 ->403 ->750 ->928 ->418 ->74 ->882 ->479 ->87 ->879
    ->667 ->378 ->944 ->805 ->
773 ->726 ->839 ->961 ->715 ->386 ->470 ->670 ->721 ->355 ->18 ->283 ->514 ->52 ->357 ->604
    ->424 ->539 ->354 ->831 ->
901 ->356 ->547 ->327 ->727 ->560 ->652 ->974 ->588 ->67 ->394 ->881 ->234 ->58 ->856 ->78
    ->651 ->484 ->957 ->66 ->
286 ->863 ->399 ->498 ->4 ->226 ->348 ->96 ->743 ->578 ->981 ->865 ->510 ->232 ->962 ->212
    ->250 ->422 ->707 ->640 ->
756 ->784 ->379 ->288 ->661 ->65 ->194 ->871 ->128 ->617 ->730 ->241 ->454 ->603 ->818 ->600
    ->598 ->147 ->675 ->931 ->
876 ->758 ->620 ->22 ->26 ->668 ->909 ->854 ->772 ->137 ->993 ->683 ->787 ->13 ->430 ->861
    ->36 ->451 ->478 ->580 ->
409 ->303 ->810 ->507 ->195 ->299 ->697 ->628 ->531 ->893 ->423 ->563 ->934 ->557 ->146
    ->301 ->845 ->786 ->343 ->79 ->
731 ->710 ->699 ->3 ->92 ->924 ->240 ->705 ->559 ->694 ->702 ->276 ->536 ->582 ->297 ->545
    ->37 ->53 ->801 ->179 ->
229 ->649 ->184 ->803 ->247 ->468 ->237 ->192 ->49 ->779 ->270 ->762 ->822 ->732 ->838 ->90
    ->639 ->972 ->55 ->990 ->
450 ->956 ->385 ->734 ->72 ->513 ->98 ->77 ->177 ->496 ->662 ->56 ->629 ->798 ->570 ->688
    ->590 ->635 ->926 ->239 ->
691 ->742 ->32 ->653 ->544 ->19 ->42 ->905 ->16 ->29 ->538 ->985 ->982 ->903 ->188 ->819
    ->126 ->488 ->395 ->215 ->
567 ->331 ->136 ->503 ->231 ->101 ->45 ->39 ->121 ->627 ->650 ->416 ->746 ->76 ->483 ->543
    ->519 ->224 ->298 ->64 ->
12 ->606 ->690 ->930 ->613 ->216 ->242 ->674 ->501 ->71 ->527 ->858 ->279 ->169 ->305 ->988
    ->367 ->701 ->749 ->633 ->
388 ->723 ->391 ->552 ->641 ->278 ->946 ->233 ->814 ->525 ->271 ->795 ->630 ->785 ->203
    ->890 ->788 ->987 ->273 ->577 ->
992 ->873 ->521 ->642 ->277 ->38 ->163 ->275 ->682 ->50 ->149 ->390 ->774 ->851 ->551 ->323
    ->456 ->436 ->178 ->973 ->
248 ->15 ->889 ->506 ->47 ->638 ->493 ->900 ->528 ->997 ->791 ->813 ->62 ->952 ->455 ->213
    ->947 ->269 ->655 ->919 ->
133 ->553 ->139 ->253 ->811 ->402 ->891 ->482 ->351 ->156 ->113 ->308 ->138 ->880 ->679
    ->857 ->207 ->57 ->201 ->868 ->
1 ->
Time: 26.4534653769806027

Length: 6225

0 ->73 ->391 ->422 ->306 ->475 ->443 ->256 ->222 ->778 ->910 ->901 ->448 ->434 ->413 ->148
    ->466 ->184 ->272 ->706 ->
990 ->769 ->702 ->668 ->479 ->875 ->996 ->849 ->977 ->145 ->265 ->625 ->381 ->362 ->415
    ->555 ->898 ->801 ->989 ->464 ->
84 ->912 ->378 ->561 ->721 ->315 ->116 ->227 ->477 ->958 ->223 ->377 ->881 ->234 ->444 ->971
    ->826 ->830 ->673 ->327 ->
954 ->429 ->986 ->815 ->645 ->342 ->797 ->411 ->322 ->894 ->439 ->366 ->725 ->348 ->677
    ->276 ->458 ->823 ->95 ->321 ->
282 ->341 ->777 ->843 ->966 ->718 ->226 ->928 ->636 ->263 ->453 ->445 ->847 ->984 ->180
    ->204 ->93 ->151 ->465 ->325 ->
670 ->485 ->405 ->450 ->956 ->617 ->819 ->236 ->533 ->963 ->25 ->992 ->660 ->402 ->925 ->649
    ->452 ->744 ->953 ->497 ->
853 ->401 ->917 ->959 ->838 ->613 ->903 ->300 ->114 ->841 ->304 ->150 ->280 ->467 ->560
```

```
    ->379 ->225 ->291 ->474 ->319 ->
708 ->344 ->44 ->911 ->289 ->396 ->773 ->457 ->412 ->357 ->459 ->970 ->390 ->979 ->955 ->436
    ->934 ->72 ->545 ->449 ->
763 ->312 ->747 ->836 ->983 ->460 ->188 ->251 ->935 ->713 ->230 ->760 ->944 ->359 ->496
    ->662 ->748 ->716 ->897 ->441 ->
534 ->855 ->471 ->632 ->338 ->978 ->407 ->699 ->461 ->137 ->205 ->105 ->906 ->972 ->245
    ->284 ->902 ->425 ->648 ->646 ->
492 ->8 ->874 ->994 ->651 ->848 ->421 ->872 ->637 ->899 ->829 ->657 ->46 ->728 ->489 ->907
    ->733 ->973 ->250 ->999 ->
860 ->523 ->476 ->417 ->918 ->806 ->431 ->803 ->175 ->866 ->916 ->751 ->343 ->914 ->807
    ->217 ->567 ->185 ->698 ->463 ->
317 ->427 ->752 ->571 ->232 ->144 ->113 ->370 ->307 ->864 ->290 ->904 ->786 ->672 ->237
    ->340 ->809 ->333 ->930 ->215 ->
128 ->802 ->400 ->149 ->136 ->488 ->395 ->780 ->481 ->873 ->915 ->846 ->220 ->799 ->961
    ->715 ->948 ->493 ->193 ->123 ->
498 ->559 ->967 ->367 ->704 ->174 ->854 ->913 ->162 ->353 ->310 ->998 ->350 ->840 ->420
    ->627 ->109 ->833 ->349 ->267 ->
825 ->451 ->397 ->643 ->388 ->238 ->790 ->96 ->743 ->604 ->684 ->623 ->782 ->394 ->576 ->908
    ->839 ->927 ->246 ->810 ->
472 ->208 ->538 ->247 ->468 ->212 ->887 ->262 ->765 ->923 ->301 ->845 ->712 ->218 ->694
    ->309 ->252 ->882 ->295 ->968 ->
731 ->710 ->334 ->106 ->885 ->739 ->964 ->339 ->863 ->487 ->735 ->58 ->856 ->705 ->125 ->37
    ->615 ->969 ->835 ->793 ->
404 ->892 ->736 ->438 ->329 ->665 ->722 ->440 ->818 ->296 ->940 ->140 ->880 ->774 ->980
    ->268 ->640 ->346 ->805 ->947 ->
867 ->676 ->962 ->658 ->792 ->157 ->932 ->798 ->859 ->975 ->383 ->759 ->758 ->499 ->937
    ->753 ->130 ->371 ->837 ->596 ->
219 ->239 ->573 ->253 ->240 ->364 ->255 ->711 ->727 ->418 ->656 ->124 ->574 ->942 ->936
    ->993 ->556 ->260 ->389 ->717 ->
135 ->286 ->974 ->241 ->933 ->160 ->729 ->741 ->161 ->889 ->771 ->153 ->111 ->737 ->269
    ->568 ->270 ->762 ->435 ->437 ->
408 ->308 ->618 ->326 ->589 ->368 ->530 ->812 ->779 ->316 ->410 ->375 ->273 ->554 ->919
    ->152 ->154 ->960 ->419 ->808 ->
231 ->652 ->187 ->900 ->884 ->387 ->470 ->199 ->761 ->141 ->822 ->732 ->277 ->186 ->893
    ->423 ->318 ->372 ->198 ->634 ->
924 ->374 ->494 ->624 ->896 ->895 ->358 ->929 ->890 ->99 ->455 ->163 ->709 ->328 ->595 ->261
    ->563 ->97 ->243 ->939 ->
143 ->302 ->742 ->565 ->991 ->196 ->285 ->842 ->274 ->298 ->598 ->147 ->447 ->345 ->170
    ->336 ->482 ->484 ->957 ->66 ->
490 ->491 ->783 ->654 ->335 ->294 ->281 ->862 ->566 ->287 ->607 ->641 ->745 ->945 ->642
    ->746 ->679 ->283 ->293 ->85 ->
376 ->666 ->373 ->597 ->638 ->324 ->41 ->369 ->832 ->271 ->667 ->905 ->941 ->870 ->821 ->164
    ->28 ->416 ->189 ->985 ->
950 ->827 ->278 ->946 ->639 ->608 ->166 ->789 ->909 ->661 ->129 ->207 ->921 ->446 ->385
    ->675 ->51 ->700 ->767 ->695 ->
680 ->486 ->738 ->772 ->987 ->266 ->82 ->244 ->635 ->380 ->692 ->142 ->249 ->382 ->428 ->68
    ->173 ->951 ->331 ->794 ->
669 ->360 ->14 ->393 ->197 ->678 ->177 ->714 ->800 ->659 ->83 ->473 ->581 ->495 ->102 ->200
    ->165 ->586 ->547 ->384 ->
257 ->943 ->424 ->347 ->108 ->850 ->79 ->80 ->337 ->258 ->146 ->352 ->644 ->528 ->254 ->356
    ->621 ->766 ->40 ->462 ->
221 ->781 ->776 ->755 ->610 ->469 ->172 ->791 ->886 ->53 ->214 ->730 ->593 ->89 ->562 ->592
    ->74 ->299 ->697 ->628 ->
49 ->796 ->599 ->724 ->824 ->126 ->851 ->606 ->690 ->224 ->121 ->740 ->852 ->754 ->804 ->686
    ->94 ->768 ->392 ->127 ->
206 ->865 ->69 ->159 ->844 ->663 ->883 ->259 ->707 ->81 ->701 ->749 ->600 ->834 ->878 ->398
    ->828 ->982 ->920 ->60 ->
23 ->549 ->590 ->409 ->314 ->211 ->320 ->305 ->585 ->582 ->288 ->478 ->117 ->228 ->132 ->194
    ->871 ->203 ->279 ->330 ->
48 ->403 ->182 ->788 ->233 ->876 ->631 ->949 ->541 ->442 ->817 ->619 ->764 ->689 ->134 ->21
    ->235 ->112 ->115 ->432 ->
515 ->86 ->988 ->664 ->480 ->363 ->248 ->682 ->532 ->814 ->454 ->603 ->76 ->683 ->355 ->681
    ->926 ->888 ->483 ->103 ->
433 ->995 ->311 ->110 ->75 ->756 ->564 ->167 ->70 ->31 ->861 ->36 ->616 ->770 ->691 ->176
    ->614 ->456 ->406 ->602 ->
39 ->354 ->426 ->414 ->611 ->938 ->583 ->155 ->56 ->784 ->548 ->133 ->292 ->43 ->734 ->578
    ->981 ->62 ->520 ->156 ->
131 ->553 ->158 ->857 ->693 ->891 ->61 ->647 ->536 ->787 ->650 ->190 ->868 ->179 ->229 ->361
```

```
     ->685 ->726 ->550 ->210 ->
629 ->922 ->877 ->591 ->703 ->674 ->209 ->242 ->720 ->609 ->584 ->332 ->813 ->785 ->719
     ->303 ->516 ->171 ->820 ->723 ->
626 ->168 ->605 ->120 ->90 ->65 ->775 ->365 ->87 ->879 ->52 ->633 ->18 ->544 ->795 ->965
     ->55 ->38 ->816 ->183 ->
671 ->216 ->192 ->202 ->931 ->976 ->512 ->579 ->15 ->92 ->527 ->858 ->997 ->688 ->91 ->430
     ->529 ->139 ->687 ->169 ->
655 ->323 ->552 ->620 ->588 ->831 ->696 ->275 ->178 ->601 ->952 ->104 ->612 ->98 ->67 ->119
     ->64 ->122 ->297 ->32 ->
507 ->195 ->78 ->47 ->399 ->811 ->869 ->757 ->191 ->531 ->577 ->33 ->351 ->59 ->77 ->107
     ->19 ->20 ->213 ->517 ->
513 ->537 ->101 ->24 ->88 ->63 ->313 ->100 ->201 ->580 ->569 ->522 ->543 ->518 ->526 ->653
     ->42 ->594 ->12 ->50 ->
572 ->181 ->521 ->57 ->22 ->26 ->524 ->546 ->622 ->34 ->551 ->570 ->35 ->506 ->587 ->504
     ->54 ->30 ->535 ->519 ->
558 ->17 ->525 ->16 ->29 ->509 ->630 ->575 ->539 ->118 ->138 ->503 ->2 ->11 ->386 ->264
     ->505 ->71 ->514 ->511 ->
542 ->508 ->750 ->9 ->4 ->510 ->540 ->557 ->10 ->501 ->502 ->5 ->6 ->13 ->3 ->7 ->500 ->27
     ->45 ->1 ->
0 ->
Time: 17.1059828097932041

Length: 7010

1 ->131 ->165 ->826 ->645 ->980 ->227 ->154 ->716 ->897 ->457 ->207 ->743 ->498 ->914 ->211
     ->474 ->919 ->456 ->184 ->
999 ->881 ->234 ->444 ->971 ->197 ->678 ->489 ->245 ->616 ->32 ->893 ->684 ->124 ->949 ->719
     ->724 ->470 ->413 ->148 ->
247 ->468 ->212 ->246 ->966 ->218 ->427 ->235 ->487 ->735 ->102 ->975 ->622 ->469 ->986
     ->232 ->629 ->981 ->226 ->660 ->
393 ->979 ->219 ->867 ->438 ->194 ->448 ->434 ->723 ->391 ->419 ->440 ->670 ->721 ->714
     ->633 ->730 ->241 ->933 ->160 ->
953 ->497 ->727 ->418 ->656 ->100 ->475 ->937 ->748 ->963 ->481 ->745 ->928 ->416 ->189
     ->939 ->967 ->220 ->354 ->495 ->
386 ->916 ->223 ->377 ->187 ->206 ->632 ->123 ->136 ->488 ->463 ->113 ->680 ->150 ->280
     ->467 ->934 ->446 ->233 ->993 ->
874 ->994 ->624 ->734 ->443 ->782 ->394 ->345 ->170 ->248 ->682 ->476 ->417 ->421 ->657 ->47
     ->677 ->574 ->712 ->163 ->
464 ->426 ->924 ->374 ->494 ->996 ->493 ->479 ->875 ->404 ->689 ->925 ->132 ->199 ->486
     ->383 ->454 ->603 ->433 ->646 ->
492 ->266 ->357 ->459 ->922 ->666 ->741 ->240 ->364 ->164 ->998 ->350 ->720 ->959 ->460
     ->942 ->230 ->654 ->636 ->729 ->
964 ->339 ->718 ->152 ->237 ->117 ->228 ->68 ->988 ->664 ->217 ->931 ->137 ->437 ->239 ->425
     ->648 ->904 ->109 ->737 ->
663 ->883 ->917 ->445 ->362 ->699 ->626 ->159 ->623 ->669 ->746 ->439 ->402 ->861 ->973
     ->683 ->286 ->974 ->588 ->652 ->
483 ->936 ->658 ->738 ->195 ->209 ->703 ->621 ->431 ->336 ->482 ->484 ->957 ->871 ->485
     ->405 ->968 ->965 ->90 ->876 ->
590 ->200 ->203 ->890 ->847 ->954 ->938 ->695 ->900 ->586 ->435 ->627 ->353 ->640 ->429
     ->359 ->216 ->225 ->715 ->948 ->
869 ->995 ->311 ->110 ->631 ->140 ->880 ->480 ->363 ->935 ->713 ->863 ->145 ->707 ->697
     ->628 ->739 ->637 ->706 ->407 ->
462 ->243 ->673 ->722 ->334 ->106 ->182 ->143 ->742 ->565 ->991 ->987 ->701 ->120 ->929
     ->135 ->985 ->982 ->698 ->491 ->
244 ->370 ->215 ->449 ->477 ->850 ->886 ->877 ->943 ->349 ->960 ->651 ->848 ->89 ->717 ->945
     ->342 ->186 ->375 ->166 ->
161 ->711 ->181 ->368 ->399 ->204 ->93 ->151 ->465 ->958 ->614 ->700 ->260 ->224 ->570 ->915
     ->178 ->149 ->367 ->704 ->
458 ->823 ->442 ->649 ->452 ->473 ->581 ->599 ->371 ->398 ->690 ->866 ->736 ->236 ->478
     ->144 ->984 ->609 ->249 ->887 ->
771 ->733 ->976 ->242 ->88 ->169 ->305 ->387 ->992 ->70 ->978 ->573 ->84 ->191 ->977 ->167
     ->642 ->432 ->99 ->842 ->
67 ->597 ->708 ->139 ->692 ->142 ->180 ->889 ->990 ->450 ->956 ->385 ->397 ->927 ->451 ->185
     ->75 ->691 ->176 ->453 ->
301 ->728 ->895 ->358 ->634 ->297 ->44 ->911 ->129 ->547 ->384 ->222 ->596 ->472 ->970 ->390
     ->662 ->913 ->162 ->686 ->
94 ->921 ->604 ->424 ->946 ->423 ->114 ->119 ->898 ->155 ->891 ->382 ->108 ->420 ->372 ->130
     ->412 ->422 ->685 ->198 ->
```

```
641 ->668 ->101 ->912 ->205 ->864 ->705 ->125 ->955 ->153 ->378 ->944 ->321 ->918 ->392 ->73
    ->840 ->231 ->582 ->923 ->
857 ->693 ->594 ->496 ->672 ->388 ->238 ->133 ->208 ->983 ->618 ->940 ->518 ->910 ->606
    ->909 ->661 ->65 ->725 ->400 ->
845 ->892 ->655 ->903 ->111 ->902 ->888 ->97 ->962 ->676 ->591 ->873 ->865 ->831 ->696 ->584
    ->168 ->607 ->859 ->548 ->
346 ->221 ->455 ->213 ->952 ->373 ->860 ->812 ->674 ->366 ->961 ->127 ->602 ->411 ->172
    ->376 ->856 ->679 ->870 ->821 ->
858 ->997 ->24 ->926 ->907 ->564 ->381 ->529 ->872 ->329 ->941 ->650 ->732 ->675 ->409 ->813
    ->702 ->112 ->899 ->118 ->
310 ->846 ->855 ->471 ->688 ->578 ->951 ->709 ->156 ->989 ->920 ->60 ->950 ->196 ->304 ->566
    ->121 ->740 ->868 ->179 ->
229 ->361 ->396 ->428 ->158 ->461 ->355 ->665 ->380 ->174 ->338 ->829 ->173 ->352 ->644
    ->932 ->403 ->95 ->183 ->671 ->
146 ->356 ->15 ->884 ->589 ->466 ->749 ->834 ->878 ->344 ->822 ->615 ->107 ->214 ->347 ->906
    ->972 ->55 ->613 ->827 ->
694 ->839 ->319 ->539 ->91 ->638 ->788 ->41 ->122 ->490 ->807 ->635 ->852 ->210 ->744 ->324
    ->401 ->879 ->667 ->905 ->
561 ->147 ->447 ->555 ->801 ->901 ->406 ->610 ->830 ->128 ->802 ->408 ->308 ->50 ->885 ->853
    ->87 ->896 ->330 ->562 ->
592 ->619 ->312 ->747 ->410 ->796 ->639 ->608 ->854 ->836 ->947 ->379 ->193 ->326 ->882
    ->295 ->348 ->436 ->331 ->832 ->
617 ->281 ->72 ->800 ->659 ->175 ->817 ->103 ->395 ->587 ->300 ->841 ->141 ->190 ->521 ->414
    ->816 ->540 ->595 ->828 ->
815 ->571 ->600 ->309 ->40 ->56 ->79 ->80 ->76 ->58 ->82 ->779 ->69 ->157 ->315 ->585 ->969
    ->593 ->778 ->66 ->
605 ->105 ->202 ->188 ->819 ->601 ->317 ->81 ->620 ->563 ->71 ->806 ->322 ->894 ->63 ->681
    ->647 ->430 ->531 ->625 ->
824 ->126 ->851 ->85 ->844 ->327 ->78 ->138 ->527 ->201 ->306 ->611 ->810 ->115 ->567 ->731
    ->710 ->192 ->556 ->53 ->
862 ->843 ->340 ->560 ->799 ->365 ->39 ->96 ->323 ->793 ->43 ->318 ->303 ->351 ->59 ->77
    ->577 ->52 ->833 ->360 ->
328 ->18 ->38 ->74 ->798 ->441 ->343 ->177 ->316 ->341 ->583 ->369 ->630 ->45 ->575 ->277
    ->415 ->64 ->787 ->804 ->
908 ->23 ->549 ->275 ->780 ->291 ->42 ->134 ->818 ->837 ->325 ->30 ->61 ->568 ->579 ->302
    ->31 ->27 ->51 ->838 ->
33 ->557 ->320 ->116 ->814 ->337 ->538 ->22 ->820 ->783 ->34 ->551 ->552 ->536 ->774 ->296
    ->776 ->29 ->515 ->86 ->
576 ->784 ->795 ->313 ->36 ->930 ->35 ->537 ->335 ->389 ->534 ->299 ->92 ->83 ->284 ->333
    ->21 ->290 ->545 ->687 ->
298 ->598 ->499 ->809 ->513 ->98 ->550 ->775 ->530 ->643 ->292 ->767 ->777 ->558 ->314 ->13
    ->48 ->781 ->28 ->20 ->
612 ->285 ->294 ->46 ->25 ->786 ->54 ->37 ->553 ->104 ->835 ->789 ->171 ->825 ->287 ->761
    ->544 ->797 ->811 ->265 ->
803 ->808 ->276 ->6 ->773 ->726 ->11 ->769 ->653 ->256 ->849 ->57 ->14 ->49 ->19 ->26 ->805
    ->288 ->10 ->580 ->
272 ->511 ->542 ->270 ->541 ->559 ->765 ->16 ->62 ->278 ->520 ->759 ->528 ->546 ->790 ->271
    ->762 ->792 ->12 ->5 ->
262 ->259 ->758 ->267 ->569 ->263 ->8 ->572 ->794 ->307 ->268 ->532 ->269 ->519 ->9 ->279
    ->768 ->517 ->772 ->543 ->
535 ->283 ->293 ->522 ->274 ->770 ->526 ->523 ->791 ->533 ->510 ->250 ->273 ->554 ->7 ->508
    ->258 ->289 ->17 ->525 ->
4 ->756 ->282 ->254 ->332 ->261 ->766 ->2 ->264 ->512 ->785 ->764 ->755 ->255 ->253 ->516
    ->760 ->751 ->514 ->763 ->
757 ->502 ->750 ->754 ->251 ->524 ->257 ->752 ->753 ->252 ->500 ->503 ->501 ->504 ->505
    ->506 ->509 ->507 ->0 ->3 ->
1 ->
Time: 12.9266952360048890

Length: 8530

2 ->849 ->249 ->227 ->122 ->738 ->366 ->971 ->589 ->868 ->678 ->727 ->686 ->594 ->477 ->958
    ->614 ->621 ->659 ->91 ->
602 ->822 ->83 ->473 ->223 ->714 ->338 ->978 ->218 ->433 ->867 ->230 ->59 ->121 ->582 ->604
    ->684 ->733 ->874 ->994 ->
624 ->817 ->619 ->481 ->873 ->865 ->831 ->441 ->371 ->837 ->75 ->448 ->833 ->349 ->493 ->479
    ->745 ->749 ->834 ->341 ->
348 ->460 ->847 ->954 ->429 ->986 ->945 ->238 ->244 ->370 ->736 ->236 ->478 ->462 ->983
```

```
    ->956 ->617 ->474 ->726 ->344 ->
44 ->458 ->459 ->591 ->106 ->120 ->991 ->967 ->970 ->989 ->119 ->64 ->951 ->480 ->363 ->434
    ->113 ->713 ->123 ->692 ->
585 ->368 ->361 ->955 ->436 ->331 ->870 ->496 ->836 ->850 ->111 ->104 ->992 ->930 ->998
    ->698 ->463 ->697 ->610 ->586 ->
744 ->598 ->959 ->124 ->454 ->985 ->107 ->214 ->706 ->109 ->737 ->605 ->225 ->715 ->948
    ->700 ->980 ->600 ->717 ->93 ->
475 ->443 ->940 ->414 ->372 ->198 ->357 ->620 ->724 ->470 ->80 ->76 ->968 ->237 ->731 ->101
    ->245 ->420 ->358 ->840 ->
484 ->465 ->975 ->322 ->786 ->442 ->848 ->81 ->494 ->599 ->362 ->234 ->732 ->595 ->699 ->195
    ->203 ->498 ->192 ->830 ->
962 ->301 ->100 ->87 ->92 ->860 ->174 ->188 ->471 ->601 ->317 ->95 ->742 ->694 ->89 ->863
    ->487 ->426 ->593 ->923 ->
108 ->711 ->491 ->197 ->612 ->311 ->220 ->965 ->55 ->917 ->743 ->981 ->497 ->990 ->707 ->721
    ->98 ->866 ->649 ->452 ->
73 ->550 ->210 ->334 ->369 ->206 ->533 ->719 ->841 ->622 ->673 ->857 ->231 ->194 ->171 ->489
    ->871 ->485 ->939 ->143 ->
856 ->246 ->248 ->360 ->488 ->207 ->722 ->326 ->556 ->842 ->68 ->173 ->243 ->852 ->618 ->567
    ->374 ->616 ->455 ->608 ->
854 ->704 ->997 ->215 ->449 ->216 ->872 ->984 ->931 ->718 ->94 ->723 ->583 ->982 ->345 ->314
    ->211 ->748 ->440 ->829 ->
584 ->412 ->116 ->467 ->741 ->324 ->935 ->577 ->609 ->447 ->728 ->855 ->708 ->999 ->72 ->679
    ->205 ->105 ->453 ->193 ->
603 ->818 ->359 ->343 ->202 ->815 ->571 ->232 ->938 ->472 ->417 ->588 ->607 ->329 ->805
    ->947 ->103 ->228 ->71 ->167 ->
746 ->439 ->656 ->427 ->235 ->239 ->993 ->45 ->943 ->373 ->812 ->464 ->84 ->710 ->689 ->469
    ->839 ->961 ->297 ->952 ->
423 ->712 ->691 ->333 ->688 ->590 ->413 ->364 ->388 ->490 ->729 ->60 ->950 ->827 ->468 ->212
    ->828 ->446 ->219 ->88 ->
615 ->365 ->696 ->824 ->69 ->102 ->995 ->597 ->566 ->476 ->558 ->112 ->62 ->342 ->797 ->933
    ->725 ->63 ->927 ->925 ->
327 ->988 ->241 ->924 ->351 ->534 ->739 ->240 ->705 ->796 ->226 ->354 ->555 ->690 ->547
    ->576 ->570 ->820 ->316 ->200 ->
281 ->862 ->28 ->789 ->613 ->695 ->450 ->82 ->960 ->832 ->843 ->859 ->548 ->74 ->495 ->438
    ->977 ->117 ->217 ->693 ->
204 ->79 ->321 ->57 ->844 ->356 ->662 ->300 ->806 ->944 ->336 ->340 ->400 ->350 ->720 ->328
    ->175 ->306 ->611 ->209 ->
592 ->191 ->295 ->486 ->285 ->181 ->823 ->347 ->587 ->99 ->572 ->158 ->131 ->483 ->809 ->966
    ->664 ->461 ->355 ->302 ->
396 ->551 ->709 ->701 ->996 ->716 ->335 ->804 ->528 ->546 ->596 ->912 ->53 ->242 ->549 ->280
    ->926 ->581 ->655 ->890 ->
196 ->540 ->561 ->466 ->346 ->221 ->672 ->979 ->973 ->46 ->802 ->52 ->974 ->339 ->667 ->808
    ->445 ->963 ->681 ->606 ->
569 ->395 ->564 ->320 ->273 ->554 ->527 ->530 ->177 ->769 ->415 ->337 ->921 ->545 ->782
    ->352 ->308 ->50 ->853 ->648 ->
222 ->410 ->972 ->747 ->916 ->431 ->332 ->183 ->858 ->184 ->869 ->170 ->310 ->115 ->851 ->85
    ->918 ->730 ->187 ->544 ->
233 ->296 ->898 ->155 ->825 ->800 ->409 ->303 ->810 ->276 ->96 ->492 ->685 ->942 ->77 ->813
    ->702 ->734 ->578 ->265 ->
299 ->509 ->735 ->515 ->86 ->666 ->794 ->157 ->680 ->541 ->97 ->182 ->543 ->284 ->61 ->913
    ->773 ->168 ->538 ->247 ->
21 ->290 ->623 ->835 ->793 ->43 ->949 ->397 ->138 ->186 ->278 ->665 ->846 ->914 ->419 ->437
    ->658 ->652 ->27 ->882 ->
669 ->557 ->161 ->47 ->902 ->425 ->169 ->305 ->563 ->934 ->703 ->499 ->19 ->790 ->313 ->771
    ->511 ->976 ->367 ->646 ->
915 ->683 ->286 ->399 ->457 ->190 ->780 ->657 ->675 ->922 ->740 ->826 ->929 ->402 ->861
    ->906 ->277 ->937 ->144 ->408 ->
573 ->677 ->953 ->568 ->579 ->56 ->110 ->401 ->274 ->560 ->160 ->816 ->185 ->964 ->142 ->180
    ->151 ->394 ->307 ->864 ->
42 ->905 ->941 ->946 ->920 ->536 ->430 ->529 ->154 ->319 ->482 ->422 ->208 ->15 ->382 ->428
    ->522 ->309 ->668 ->32 ->
150 ->674 ->910 ->39 ->542 ->521 ->770 ->635 ->517 ->325 ->282 ->164 ->353 ->928 ->229 ->889
    ->166 ->670 ->687 ->298 ->
660 ->40 ->178 ->671 ->418 ->785 ->575 ->520 ->647 ->919 ->559 ->784 ->580 ->36 ->774 ->312
    ->260 ->904 ->287 ->292 ->
908 ->67 ->676 ->318 ->644 ->26 ->638 ->390 ->682 ->894 ->65 ->661 ->90 ->289 ->49 ->385
    ->775 ->642 ->432 ->936 ->
270 ->165 ->172 ->791 ->886 ->778 ->66 ->633 ->798 ->510 ->639 ->907 ->535 ->54 ->799 ->179
```

```
    ->654 ->24 ->899 ->140 ->
814 ->525 ->153 ->781 ->888 ->213 ->653 ->136 ->330 ->162 ->969 ->776 ->636 ->130 ->761
    ->392 ->403 ->33 ->152 ->421 ->
539 ->911 ->772 ->987 ->224 ->795 ->763 ->663 ->398 ->266 ->391 ->901 ->932 ->444 ->48 ->821
    ->159 ->275 ->892 ->787 ->
532 ->451 ->637 ->519 ->146 ->283 ->293 ->537 ->384 ->257 ->294 ->957 ->768 ->630 ->304
    ->272 ->135 ->17 ->58 ->51 ->
838 ->411 ->201 ->269 ->819 ->291 ->114 ->23 ->765 ->134 ->176 ->315 ->268 ->900 ->199 ->803
    ->651 ->262 ->801 ->640 ->
70 ->378 ->643 ->279 ->12 ->632 ->263 ->553 ->631 ->118 ->78 ->13 ->896 ->31 ->435 ->792
    ->41 ->777 ->523 ->271 ->
264 ->145 ->139 ->565 ->387 ->389 ->895 ->767 ->267 ->189 ->149 ->897 ->807 ->261 ->766
    ->163 ->574 ->628 ->885 ->406 ->
508 ->507 ->650 ->416 ->424 ->788 ->516 ->259 ->20 ->514 ->29 ->552 ->404 ->148 ->562 ->881
    ->641 ->903 ->38 ->147 ->
37 ->513 ->141 ->258 ->254 ->759 ->8 ->381 ->757 ->756 ->137 ->253 ->456 ->256 ->380 ->883
    ->884 ->506 ->634 ->887 ->
524 ->18 ->376 ->758 ->14 ->34 ->811 ->379 ->876 ->762 ->880 ->754 ->760 ->22 ->845 ->526
    ->503 ->129 ->783 ->323 ->
779 ->504 ->891 ->7 ->10 ->25 ->383 ->512 ->627 ->386 ->132 ->645 ->35 ->753 ->502 ->909 ->4
    ->877 ->764 ->407 ->
156 ->531 ->288 ->125 ->405 ->133 ->625 ->126 ->30 ->11 ->875 ->128 ->375 ->393 ->893 ->626
    ->127 ->252 ->755 ->878 ->
879 ->629 ->5 ->377 ->518 ->505 ->501 ->500 ->255 ->752 ->750 ->9 ->751 ->6 ->3 ->251 ->250
    ->16 ->0 ->1 ->
2 ->
Time: 13.7177257919684052
```

2 Iteration

```
Length: 5659

1 ->265 ->148 ->843 ->966 ->718 ->535 ->336 ->776 ->472 ->208 ->827 ->151 ->31 ->916 ->197
    ->359 ->915 ->846 ->476 ->
417 ->421 ->872 ->120 ->929 ->339 ->452 ->225 ->291 ->647 ->217 ->464 ->432 ->185 ->75 ->51
    ->976 ->218 ->124 ->180 ->
54 ->243 ->994 ->325 ->282 ->689 ->254 ->738 ->366 ->166 ->161 ->711 ->593 ->89 ->717 ->899
    ->302 ->534 ->855 ->938 ->
666 ->741 ->764 ->102 ->99 ->312 ->747 ->238 ->413 ->576 ->429 ->986 ->815 ->645 ->724 ->168
    ->809 ->25 ->736 ->676 ->
387 ->948 ->700 ->260 ->389 ->989 ->920 ->223 ->435 ->437 ->408 ->457 ->412 ->658 ->913
    ->162 ->353 ->816 ->540 ->346 ->
370 ->252 ->453 ->561 ->844 ->491 ->244 ->833 ->935 ->713 ->517 ->859 ->975 ->383 ->769
    ->708 ->344 ->44 ->129 ->614 ->
117 ->438 ->607 ->329 ->665 ->152 ->735 ->782 ->473 ->581 ->599 ->481 ->202 ->943 ->349
    ->267 ->489 ->264 ->562 ->471 ->
632 ->621 ->431 ->497 ->115 ->0 ->73 ->840 ->420 ->69 ->157 ->6 ->281 ->555 ->898 ->549
    ->817 ->794 ->884 ->999 ->
860 ->245 ->894 ->712 ->591 ->703 ->360 ->780 ->767 ->153 ->781 ->428 ->459 ->122 ->173
    ->951 ->227 ->753 ->221 ->692 ->
142 ->272 ->8 ->998 ->959 ->460 ->554 ->836 ->983 ->618 ->326 ->246 ->198 ->634 ->887 ->262
    ->914 ->289 ->396 ->132 ->
199 ->556 ->657 ->502 ->610 ->671 ->337 ->258 ->615 ->158 ->970 ->24 ->182 ->678 ->511 ->328
    ->533 ->490 ->695 ->373 ->
597 ->59 ->320 ->829 ->411 ->509 ->979 ->219 ->295 ->485 ->771 ->654 ->636 ->88 ->492 ->602
    ->589 ->500 ->187 ->340 ->
266 ->82 ->100 ->945 ->759 ->971 ->826 ->830 ->41 ->249 ->936 ->922 ->877 ->17 ->441 ->35
    ->10 ->477 ->150 ->280 ->
558 ->322 ->637 ->222 ->778 ->910 ->352 ->135 ->680 ->341 ->777 ->729 ->60 ->950 ->196 ->463
    ->317 ->427 ->235 ->644 ->
932 ->480 ->97 ->381 ->362 ->415 ->131 ->768 ->499 ->2 ->210 ->5 ->722 ->440 ->191 ->977
    ->145 ->789 ->171 ->292 ->
309 ->631 ->486 ->236 ->84 ->912 ->766 ->978 ->573 ->677 ->681 ->209 ->744 ->40 ->193 ->123
    ->583 ->369 ->832 ->174 ->
799 ->46 ->83 ->574 ->824 ->315 ->116 ->660 ->564 ->167 ->307 ->313 ->103 ->433 ->867 ->61
    ->586 ->806 ->363 ->284 ->
204 ->93 ->140 ->775 ->685 ->942 ->375 ->550 ->523 ->847 ->954 ->176 ->160 ->953 ->125 ->504
    ->393 ->105 ->673 ->622 ->
```

```
469 ->345 ->585 ->495 ->807 ->114 ->119 ->294 ->572 ->487 ->907 ->733 ->95 ->321 ->189 ->624
    ->896 ->895 ->358 ->34 ->
740 ->852 ->802 ->400 ->314 ->211 ->172 ->968 ->965 ->955 ->770 ->392 ->967 ->828 ->287
    ->761 ->141 ->465 ->958 ->897 ->
888 ->566 ->361 ->80 ->165 ->21 ->587 ->548 ->605 ->659 ->808 ->739 ->964 ->719 ->841 ->311
    ->752 ->571 ->625 ->969 ->
835 ->793 ->404 ->380 ->324 ->672 ->68 ->669 ->594 ->33 ->350 ->330 ->48 ->261 ->568 ->892
    ->797 ->728 ->575 ->91 ->
918 ->923 ->306 ->584 ->374 ->908 ->937 ->748 ->300 ->447 ->444 ->144 ->419 ->251 ->259 ->20
    ->228 ->720 ->611 ->494 ->
110 ->862 ->516 ->368 ->842 ->274 ->530 ->293 ->85 ->376 ->449 ->763 ->949 ->397 ->643 ->755
    ->751 ->820 ->783 ->595 ->
532 ->790 ->290 ->904 ->220 ->127 ->206 ->462 ->714 ->800 ->342 ->704 ->569 ->112 ->917
    ->445 ->963 ->687 ->347 ->906 ->
933 ->200 ->522 ->118 ->310 ->94 ->333 ->406 ->921 ->960 ->995 ->757 ->939 ->143 ->401 ->508
    ->693 ->475 ->443 ->512 ->
263 ->804 ->686 ->382 ->524 ->398 ->318 ->372 ->541 ->442 ->107 ->214 ->706 ->407 ->869
    ->170 ->623 ->467 ->183 ->616 ->
612 ->812 ->984 ->332 ->716 ->335 ->760 ->190 ->9 ->996 ->849 ->106 ->885 ->874 ->285 ->911
    ->825 ->134 ->255 ->43 ->
14 ->709 ->792 ->601 ->155 ->526 ->941 ->870 ->821 ->164 ->28 ->886 ->230 ->866 ->154 ->30
    ->81 ->656 ->505 ->875 ->
70 ->316 ->410 ->592 ->529 ->537 ->175 ->834 ->878 ->458 ->364 ->520 ->446 ->304 ->63 ->927
    ->925 ->466 ->853 ->648 ->
426 ->414 ->109 ->737 ->663 ->883 ->257 ->609 ->619 ->27 ->542 ->626 ->159 ->23 ->765 ->646
    ->365 ->696 ->186 ->104 ->
11 ->384 ->664 ->461 ->439 ->579 ->130 ->371 ->837 ->596 ->405 ->608 ->108 ->850 ->111 ->902
    ->425 ->474 ->319 ->7 ->
434 ->268 ->448 ->334 ->745 ->205 ->864 ->754 ->725 ->296 ->940 ->518 ->515 ->86 ->684 ->848
    ->698 ->546 ->980 ->796 ->
565 ->991 ->256 ->338 ->181 ->823 ->377 ->403 ->750 ->928 ->418 ->74 ->882 ->479 ->87 ->879
    ->667 ->378 ->944 ->805 ->
773 ->726 ->839 ->961 ->715 ->386 ->470 ->670 ->721 ->355 ->18 ->283 ->514 ->52 ->357 ->604
    ->424 ->539 ->354 ->831 ->
901 ->356 ->547 ->327 ->727 ->560 ->652 ->974 ->588 ->67 ->394 ->881 ->234 ->58 ->856 ->78
    ->651 ->484 ->957 ->66 ->
286 ->863 ->399 ->498 ->4 ->226 ->348 ->96 ->743 ->578 ->981 ->865 ->510 ->232 ->962 ->212
    ->250 ->422 ->707 ->640 ->
756 ->784 ->379 ->288 ->661 ->65 ->194 ->871 ->128 ->617 ->730 ->241 ->454 ->603 ->818 ->600
    ->598 ->147 ->675 ->931 ->
876 ->758 ->620 ->22 ->26 ->668 ->909 ->854 ->772 ->137 ->993 ->683 ->787 ->13 ->430 ->861
    ->36 ->451 ->478 ->580 ->
409 ->303 ->810 ->507 ->195 ->299 ->697 ->628 ->531 ->893 ->423 ->563 ->934 ->557 ->146
    ->301 ->845 ->786 ->343 ->79 ->
731 ->710 ->699 ->3 ->92 ->924 ->240 ->705 ->559 ->694 ->702 ->276 ->536 ->582 ->297 ->545
    ->37 ->53 ->801 ->179 ->
229 ->649 ->184 ->803 ->247 ->468 ->237 ->192 ->49 ->779 ->270 ->762 ->822 ->732 ->838 ->90
    ->639 ->972 ->55 ->990 ->
450 ->956 ->385 ->734 ->72 ->513 ->98 ->77 ->177 ->496 ->662 ->56 ->629 ->798 ->570 ->688
    ->590 ->635 ->926 ->239 ->
691 ->742 ->32 ->653 ->544 ->19 ->42 ->905 ->16 ->29 ->538 ->985 ->982 ->903 ->188 ->819
    ->126 ->488 ->395 ->215 ->
567 ->331 ->136 ->503 ->231 ->101 ->45 ->39 ->121 ->627 ->650 ->416 ->746 ->76 ->483 ->543
    ->519 ->224 ->298 ->64 ->
12 ->606 ->690 ->930 ->613 ->216 ->242 ->674 ->501 ->71 ->527 ->858 ->279 ->169 ->305 ->988
    ->367 ->701 ->749 ->633 ->
388 ->723 ->391 ->552 ->641 ->278 ->946 ->233 ->814 ->525 ->271 ->795 ->630 ->785 ->203
    ->890 ->788 ->987 ->273 ->577 ->
992 ->873 ->521 ->642 ->277 ->38 ->163 ->275 ->682 ->50 ->149 ->390 ->774 ->851 ->551 ->323
    ->456 ->436 ->178 ->973 ->
248 ->15 ->889 ->506 ->47 ->638 ->493 ->900 ->528 ->997 ->791 ->813 ->62 ->952 ->455 ->213
    ->947 ->269 ->655 ->919 ->
133 ->553 ->139 ->253 ->811 ->402 ->891 ->482 ->351 ->156 ->113 ->308 ->138 ->880 ->679
    ->857 ->207 ->57 ->201 ->868 ->
1 ->
Time: 52.7259218120016158

Length: 6076
```

```
57
58 1 ->265 ->625 ->381 ->250 ->811 ->869 ->302 ->56 ->364 ->255 ->711 ->471 ->870 ->496 ->672
       ->752 ->856 ->860 ->523 ->
59 450 ->330 ->667 ->378 ->571 ->769 ->708 ->959 ->966 ->764 ->961 ->326 ->246 ->349 ->493
       ->223 ->435 ->792 ->911 ->480 ->
60 97 ->243 ->994 ->651 ->485 ->346 ->977 ->167 ->307 ->855 ->871 ->670 ->484 ->979 ->365 ->382
       ->428 ->459 ->591 ->984 ->
61 332 ->813 ->785 ->221 ->455 ->375 ->373 ->256 ->291 ->790 ->96 ->321 ->189 ->352 ->644 ->932
       ->990 ->936 ->993 ->913 ->
62 773 ->457 ->303 ->782 ->473 ->312 ->125 ->498 ->914 ->289 ->925 ->466 ->618 ->370 ->252
       ->453 ->445 ->717 ->324 ->859 ->
63 975 ->622 ->601 ->858 ->724 ->898 ->267 ->489 ->887 ->13 ->338 ->978 ->74 ->495 ->438 ->194
       ->448 ->175 ->866 ->335 ->
64 294 ->262 ->765 ->923 ->306 ->129 ->614 ->832 ->863 ->487 ->907 ->689 ->933 ->725 ->400
       ->956 ->815 ->972 ->245 ->616 ->
65 275 ->682 ->532 ->478 ->657 ->46 ->802 ->761 ->443 ->405 ->389 ->161 ->889 ->771 ->654 ->441
       ->894 ->439 ->366 ->987 ->
66 273 ->706 ->864 ->290 ->397 ->138 ->527 ->845 ->712 ->163 ->464 ->432 ->287 ->970 ->695
       ->680 ->341 ->897 ->462 ->215 ->
67 449 ->477 ->958 ->350 ->720 ->744 ->598 ->499 ->937 ->421 ->754 ->804 ->658 ->738 ->916
       ->197 ->359 ->915 ->850 ->947 ->
68 867 ->238 ->413 ->852 ->767 ->985 ->982 ->408 ->594 ->627 ->109 ->833 ->935 ->713 ->123
       ->173 ->840 ->420 ->707 ->368 ->
69 842 ->274 ->185 ->847 ->954 ->938 ->666 ->741 ->202 ->931 ->876 ->374 ->494 ->922 ->877
       ->647 ->217 ->567 ->581 ->655 ->
70 234 ->910 ->901 ->755 ->476 ->417 ->918 ->806 ->944 ->737 ->429 ->821 ->184 ->272 ->653
       ->544 ->605 ->270 ->778 ->66 ->
71 895 ->340 ->862 ->260 ->423 ->974 ->427 ->824 ->315 ->934 ->446 ->219 ->295 ->968 ->731
       ->141 ->465 ->325 ->574 ->691 ->
72 176 ->841 ->843 ->334 ->433 ->995 ->776 ->886 ->230 ->760 ->190 ->868 ->436 ->383 ->881
       ->157 ->805 ->288 ->816 ->183 ->
73 424 ->347 ->205 ->280 ->419 ->155 ->416 ->564 ->104 ->633 ->388 ->490 ->664 ->939 ->204
       ->283 ->293 ->908 ->839 ->927 ->
74 220 ->127 ->729 ->964 ->339 ->452 ->225 ->981 ->345 ->585 ->198 ->440 ->829 ->688 ->846
       ->282 ->683 ->797 ->637 ->819 ->
75 118 ->310 ->998 ->300 ->95 ->474 ->659 ->83 ->41 ->884 ->999 ->72 ->463 ->810 ->472 ->650
       ->322 ->983 ->174 ->799 ->
76 674 ->853 ->401 ->216 ->626 ->783 ->718 ->152 ->218 ->636 ->685 ->249 ->227 ->753 ->458
       ->928 ->111 ->407 ->646 ->756 ->
77 784 ->379 ->193 ->371 ->479 ->281 ->271 ->980 ->268 ->640 ->948 ->444 ->369 ->206 ->258
       ->394 ->576 ->714 ->809 ->703 ->
78 621 ->766 ->391 ->422 ->660 ->909 ->854 ->286 ->240 ->60 ->950 ->827 ->694 ->309 ->488 ->395
       ->849 ->851 ->410 ->241 ->
79 254 ->316 ->967 ->236 ->84 ->314 ->353 ->762 ->611 ->209 ->342 ->406 ->610 ->135 ->747 ->926
       ->903 ->188 ->470 ->199 ->
80 486 ->106 ->896 ->777 ->822 ->296 ->201 ->469 ->172 ->791 ->299 ->697 ->304 ->150 ->540
       ->165 ->826 ->251 ->844 ->491 ->
81 244 ->615 ->969 ->835 ->789 ->171 ->736 ->311 ->67 ->878 ->337 ->595 ->592 ->101 ->460 ->554
       ->836 ->893 ->684 ->124 ->
82 668 ->409 ->875 ->404 ->892 ->609 ->116 ->228 ->68 ->669 ->746 ->795 ->763 ->132 ->912 ->317
       ->917 ->149 ->367 ->794 ->
83 530 ->177 ->50 ->357 ->604 ->597 ->638 ->344 ->323 ->442 ->649 ->415 ->131 ->890 ->454 ->603
       ->426 ->412 ->467 ->558 ->
84 235 ->596 ->222 ->733 ->953 ->577 ->992 ->698 ->482 ->468 ->807 ->635 ->162 ->686 ->578
       ->951 ->356 ->879 ->329 ->98 ->
85 955 ->770 ->772 ->645 ->148 ->247 ->343 ->781 ->120 ->991 ->848 ->333 ->377 ->354 ->663
       ->883 ->257 ->943 ->434 ->456 ->
86 656 ->100 ->475 ->372 ->253 ->702 ->734 ->919 ->745 ->945 ->759 ->971 ->838 ->61 ->586 ->818
       ->600 ->986 ->800 ->700 ->
87 208 ->730 ->593 ->624 ->817 ->387 ->430 ->43 ->318 ->828 ->88 ->492 ->153 ->200 ->203 ->279
       ->385 ->775 ->710 ->301 ->
88 145 ->705 ->181 ->823 ->565 ->617 ->34 ->292 ->904 ->727 ->834 ->313 ->726 ->550 ->156 ->277
       ->186 ->187 ->402 ->599 ->
89 362 ->787 ->808 ->976 ->242 ->226 ->965 ->376 ->12 ->139 ->692 ->739 ->393 ->105 ->348 ->677
       ->681 ->962 ->212 ->921 ->
90 960 ->28 ->768 ->392 ->73 ->641 ->742 ->568 ->431 ->803 ->142 ->180 ->957 ->229 ->81 ->973
       ->997 ->549 ->590 ->425 ->
91 584 ->360 ->69 ->327 ->579 ->130 ->857 ->693 ->570 ->248 ->112 ->899 ->709 ->328 ->79 ->80
```

```
    ->76 ->573 ->518 ->47 ->
399 ->740 ->166 ->716 ->207 ->269 ->796 ->606 ->481 ->543 ->93 ->140 ->814 ->94 ->122 ->159
    ->874 ->361 ->632 ->263 ->
331 ->136 ->49 ->801 ->989 ->920 ->169 ->305 ->545 ->687 ->902 ->888 ->358 ->144 ->113 ->906
    ->566 ->665 ->722 ->64 ->
276 ->259 ->900 ->559 ->414 ->941 ->607 ->85 ->715 ->266 ->82 ->779 ->147 ->447 ->319 ->539
    ->91 ->211 ->403 ->182 ->
678 ->511 ->170 ->380 ->613 ->90 ->820 ->196 ->418 ->115 ->623 ->619 ->541 ->880 ->996 ->160
    ->103 ->224 ->298 ->555 ->
582 ->837 ->75 ->642 ->662 ->748 ->963 ->671 ->146 ->675 ->117 ->631 ->351 ->59 ->320 ->780
    ->774 ->872 ->930 ->239 ->
21 ->15 ->608 ->54 ->133 ->384 ->749 ->560 ->652 ->483 ->42 ->905 ->548 ->121 ->728 ->62
    ->99 ->355 ->634 ->536 ->
24 ->210 ->629 ->798 ->510 ->735 ->58 ->137 ->437 ->398 ->386 ->35 ->513 ->396 ->942 ->285
    ->20 ->213 ->517 ->673 ->
37 ->831 ->308 ->793 ->529 ->92 ->924 ->701 ->128 ->52 ->732 ->952 ->535 ->336 ->940 ->988
    ->32 ->587 ->261 ->563 ->
151 ->643 ->45 ->119 ->882 ->497 ->630 ->451 ->572 ->192 ->830 ->628 ->143 ->214 ->57 ->546
    ->699 ->538 ->168 ->873 ->
865 ->40 ->107 ->89 ->583 ->284 ->891 ->946 ->639 ->134 ->561 ->885 ->63 ->751 ->110 ->27
    ->547 ->237 ->602 ->411 ->
102 ->278 ->788 ->812 ->676 ->70 ->31 ->861 ->36 ->191 ->531 ->556 ->620 ->721 ->44 ->164
    ->25 ->39 ->65 ->53 ->
108 ->588 ->78 ->233 ->743 ->719 ->534 ->33 ->231 ->516 ->390 ->528 ->158 ->825 ->580 ->786
    ->522 ->612 ->18 ->38 ->
552 ->758 ->461 ->264 ->363 ->504 ->723 ->704 ->178 ->14 ->23 ->537 ->557 ->11 ->19 ->533
    ->750 ->195 ->696 ->690 ->
525 ->16 ->29 ->55 ->17 ->179 ->77 ->562 ->929 ->297 ->949 ->514 ->51 ->232 ->86 ->3 ->8
    ->26 ->515 ->553 ->
126 ->524 ->648 ->679 ->501 ->569 ->87 ->661 ->589 ->7 ->520 ->508 ->507 ->575 ->22 ->10
    ->526 ->551 ->154 ->757 ->
502 ->114 ->48 ->505 ->521 ->542 ->512 ->506 ->509 ->519 ->2 ->6 ->9 ->4 ->71 ->500 ->503
    ->30 ->5 ->0 ->
1 ->
Time: 34.8287891470827162

Length: 7088

2 ->210 ->471 ->688 ->846 ->476 ->159 ->636 ->886 ->482 ->484 ->957 ->383 ->715 ->386 ->919
    ->456 ->685 ->735 ->782 ->
245 ->698 ->463 ->182 ->214 ->978 ->479 ->875 ->417 ->206 ->632 ->621 ->151 ->465 ->748
    ->963 ->986 ->133 ->459 ->922 ->
740 ->868 ->179 ->924 ->409 ->234 ->444 ->971 ->197 ->645 ->148 ->247 ->108 ->711 ->181
    ->494 ->996 ->470 ->670 ->721 ->
714 ->725 ->400 ->914 ->192 ->898 ->155 ->416 ->189 ->883 ->955 ->224 ->487 ->644 ->332
    ->716 ->127 ->438 ->861 ->973 ->
683 ->477 ->150 ->280 ->419 ->123 ->173 ->840 ->420 ->707 ->640 ->907 ->729 ->964 ->719
    ->904 ->220 ->313 ->724 ->941 ->
244 ->418 ->656 ->427 ->235 ->980 ->680 ->341 ->351 ->947 ->145 ->139 ->959 ->460 ->712
    ->163 ->480 ->363 ->248 ->916 ->
217 ->464 ->457 ->207 ->737 ->429 ->936 ->658 ->738 ->195 ->209 ->958 ->699 ->327 ->727
    ->371 ->227 ->119 ->985 ->982 ->
720 ->611 ->999 ->848 ->939 ->972 ->55 ->917 ->445 ->717 ->945 ->370 ->890 ->99 ->998 ->350
    ->929 ->135 ->425 ->648 ->
646 ->710 ->168 ->809 ->966 ->218 ->393 ->990 ->450 ->673 ->879 ->667 ->903 ->188 ->198
    ->440 ->818 ->600 ->413 ->364 ->
760 ->944 ->321 ->211 ->172 ->499 ->937 ->421 ->722 ->454 ->603 ->433 ->373 ->597 ->638
    ->368 ->361 ->80 ->165 ->317 ->
225 ->981 ->226 ->746 ->439 ->882 ->495 ->358 ->634 ->887 ->396 ->428 ->826 ->830 ->962
    ->212 ->663 ->930 ->239 ->481 ->
157 ->483 ->435 ->627 ->485 ->453 ->654 ->335 ->804 ->686 ->578 ->692 ->960 ->651 ->243
    ->994 ->653 ->733 ->236 ->394 ->
993 ->385 ->143 ->741 ->161 ->713 ->230 ->866 ->931 ->137 ->205 ->105 ->202 ->160 ->953
    ->469 ->839 ->961 ->408 ->631 ->
949 ->678 ->489 ->694 ->854 ->913 ->708 ->344 ->822 ->732 ->675 ->467 ->934 ->979 ->365
    ->382 ->975 ->974 ->349 ->706 ->
607 ->918 ->730 ->241 ->612 ->637 ->222 ->726 ->473 ->581 ->867 ->933 ->498 ->897 ->180
    ->204 ->93 ->942 ->194 ->871 ->
```

```
497 ->381 ->900 ->586 ->402 ->599 ->362 ->662 ->911 ->129 ->614 ->700 ->823 ->442 ->107
    ->617 ->991 ->967 ->367 ->704 ->
474 ->852 ->572 ->326 ->968 ->965 ->309 ->668 ->909 ->104 ->992 ->845 ->411 ->144 ->925
    ->458 ->928 ->448 ->434 ->185 ->
847 ->976 ->242 ->120 ->950 ->196 ->184 ->893 ->731 ->709 ->200 ->203 ->140 ->880 ->639
    ->906 ->923 ->167 ->935 ->865 ->
331 ->984 ->641 ->745 ->860 ->834 ->407 ->718 ->152 ->154 ->742 ->493 ->193 ->449 ->567
    ->669 ->649 ->404 ->892 ->736 ->
423 ->110 ->437 ->398 ->138 ->186 ->175 ->620 ->829 ->657 ->42 ->744 ->598 ->983 ->174 ->338
    ->876 ->374 ->616 ->537 ->
219 ->610 ->671 ->94 ->739 ->240 ->443 ->794 ->684 ->124 ->574 ->691 ->176 ->491 ->357 ->969
    ->910 ->606 ->690 ->431 ->
387 ->948 ->121 ->424 ->347 ->587 ->475 ->943 ->307 ->626 ->384 ->749 ->940 ->850 ->111
    ->695 ->68 ->679 ->870 ->496 ->
672 ->237 ->602 ->415 ->938 ->832 ->178 ->601 ->635 ->162 ->353 ->391 ->85 ->376 ->666 ->375
    ->166 ->466 ->191 ->977 ->
208 ->392 ->888 ->97 ->378 ->884 ->589 ->136 ->488 ->395 ->215 ->995 ->90 ->342 ->187 ->340
    ->862 ->771 ->921 ->379 ->
153 ->132 ->199 ->486 ->412 ->422 ->660 ->98 ->902 ->687 ->585 ->951 ->705 ->405 ->389 ->989
    ->406 ->877 ->591 ->703 ->
674 ->853 ->122 ->297 ->462 ->221 ->359 ->901 ->932 ->403 ->630 ->592 ->337 ->814 ->310
    ->115 ->623 ->619 ->541 ->842 ->
158 ->478 ->580 ->84 ->455 ->213 ->952 ->249 ->87 ->596 ->336 ->106 ->896 ->661 ->65 ->246
    ->492 ->8 ->874 ->285 ->
360 ->348 ->677 ->681 ->647 ->430 ->296 ->201 ->369 ->851 ->177 ->316 ->410 ->565 ->468
    ->142 ->805 ->223 ->594 ->615 ->
190 ->780 ->366 ->987 ->701 ->128 ->294 ->723 ->116 ->970 ->869 ->841 ->622 ->905 ->954
    ->100 ->447 ->728 ->895 ->232 ->
629 ->697 ->628 ->885 ->89 ->863 ->149 ->171 ->801 ->147 ->693 ->817 ->118 ->51 ->806 ->170
    ->380 ->682 ->532 ->451 ->
45 ->605 ->72 ->312 ->747 ->926 ->988 ->927 ->426 ->125 ->452 ->810 ->472 ->696 ->372 ->183
    ->824 ->613 ->827 ->436 ->
559 ->414 ->109 ->441 ->894 ->63 ->229 ->915 ->345 ->314 ->78 ->40 ->103 ->228 ->113 ->836
    ->946 ->843 ->859 ->550 ->
156 ->920 ->564 ->446 ->997 ->791 ->624 ->734 ->308 ->618 ->169 ->560 ->490 ->664 ->233
    ->743 ->287 ->292 ->908 ->401 ->
216 ->114 ->555 ->857 ->231 ->889 ->330 ->676 ->70 ->329 ->304 ->146 ->665 ->604 ->328 ->595
    ->828 ->66 ->561 ->872 ->
813 ->702 ->112 ->899 ->134 ->864 ->69 ->390 ->798 ->377 ->881 ->878 ->354 ->831 ->82 ->855
    ->796 ->584 ->131 ->291 ->
633 ->388 ->238 ->56 ->61 ->76 ->547 ->566 ->795 ->318 ->873 ->858 ->279 ->346 ->323 ->301
    ->141 ->286 ->92 ->96 ->
334 ->164 ->765 ->652 ->833 ->303 ->783 ->549 ->590 ->808 ->689 ->577 ->91 ->797 ->86 ->576
    ->570 ->825 ->956 ->815 ->
356 ->755 ->432 ->339 ->844 ->284 ->333 ->50 ->812 ->355 ->820 ->593 ->800 ->659 ->311 ->67
    ->546 ->790 ->569 ->290 ->
75 ->642 ->52 ->582 ->837 ->802 ->81 ->538 ->512 ->789 ->71 ->299 ->816 ->49 ->281 ->73
    ->306 ->320 ->305 ->545 ->
558 ->322 ->786 ->643 ->60 ->266 ->44 ->62 ->849 ->57 ->778 ->102 ->74 ->788 ->563 ->126
    ->811 ->912 ->608 ->807 ->
891 ->536 ->302 ->47 ->399 ->539 ->324 ->397 ->544 ->571 ->625 ->583 ->343 ->557 ->88 ->54
    ->36 ->64 ->787 ->650 ->
48 ->821 ->772 ->282 ->531 ->295 ->762 ->553 ->101 ->777 ->526 ->271 ->835 ->575 ->277 ->38
    ->655 ->31 ->23 ->516 ->
793 ->43 ->274 ->95 ->319 ->542 ->270 ->461 ->554 ->548 ->58 ->33 ->39 ->530 ->761 ->275
    ->568 ->579 ->325 ->298 ->
29 ->263 ->792 ->41 ->37 ->53 ->518 ->515 ->130 ->609 ->521 ->770 ->18 ->79 ->315 ->24 ->784
    ->819 ->276 ->83 ->
540 ->524 ->588 ->803 ->529 ->265 ->352 ->262 ->278 ->520 ->510 ->766 ->77 ->562 ->856 ->769
    ->288 ->775 ->28 ->768 ->
573 ->779 ->260 ->764 ->799 ->46 ->264 ->753 ->533 ->551 ->552 ->758 ->838 ->16 ->300 ->272
    ->781 ->21 ->289 ->15 ->
35 ->10 ->25 ->258 ->269 ->273 ->534 ->267 ->543 ->774 ->34 ->268 ->750 ->32 ->773 ->508
    ->27 ->254 ->759 ->528 ->
507 ->535 ->283 ->514 ->556 ->752 ->17 ->13 ->252 ->776 ->20 ->756 ->22 ->26 ->751 ->259
    ->253 ->767 ->785 ->11 ->
19 ->117 ->527 ->293 ->522 ->519 ->504 ->757 ->763 ->261 ->59 ->257 ->513 ->250 ->523 ->525
    ->754 ->505 ->517 ->503 ->
```

```
14 ->30 ->251 ->255 ->256 ->502 ->506 ->509 ->12 ->5 ->6 ->9 ->4 ->511 ->500 ->501 ->0 ->7
    ->1 ->3 ->
2 ->
Time: 26.5923148076981306

Length: 8789

4 ->88 ->962 ->844 ->491 ->486 ->854 ->733 ->236 ->84 ->309 ->591 ->873 ->366 ->961 ->297
    ->462 ->105 ->202 ->664 ->
217 ->725 ->697 ->610 ->99 ->572 ->709 ->956 ->981 ->62 ->342 ->103 ->433 ->373 ->860 ->424
    ->843 ->597 ->566 ->990 ->
171 ->985 ->692 ->100 ->749 ->222 ->984 ->731 ->90 ->332 ->716 ->818 ->834 ->220 ->994 ->494
    ->192 ->812 ->350 ->720 ->
695 ->826 ->830 ->440 ->829 ->621 ->600 ->986 ->800 ->91 ->954 ->938 ->472 ->326 ->246 ->612
    ->466 ->672 ->979 ->737 ->
92 ->741 ->490 ->485 ->939 ->199 ->214 ->347 ->471 ->870 ->964 ->489 ->245 ->73 ->993 ->370
    ->838 ->714 ->809 ->618 ->
823 ->937 ->227 ->944 ->359 ->343 ->593 ->89 ->327 ->727 ->560 ->358 ->395 ->564 ->122 ->976
    ->928 ->207 ->106 ->736 ->
341 ->348 ->249 ->936 ->995 ->496 ->480 ->119 ->684 ->706 ->864 ->492 ->721 ->694 ->111
    ->722 ->357 ->604 ->403 ->95 ->
474 ->726 ->344 ->301 ->123 ->971 ->940 ->745 ->945 ->487 ->872 ->120 ->70 ->239 ->978 ->734
    ->162 ->711 ->842 ->186 ->
104 ->481 ->493 ->223 ->622 ->601 ->317 ->81 ->48 ->244 ->991 ->987 ->724 ->477 ->861 ->973
    ->46 ->728 ->958 ->556 ->
998 ->93 ->852 ->588 ->607 ->438 ->329 ->748 ->851 ->85 ->708 ->827 ->841 ->475 ->923 ->867
    ->230 ->59 ->77 ->698 ->
573 ->847 ->934 ->363 ->248 ->117 ->228 ->71 ->848 ->110 ->221 ->665 ->676 ->808 ->191 ->328
    ->334 ->484 ->957 ->871 ->
924 ->374 ->187 ->194 ->935 ->620 ->449 ->225 ->715 ->988 ->989 ->464 ->570 ->943 ->746
    ->439 ->855 ->972 ->747 ->238 ->
56 ->364 ->234 ->595 ->699 ->461 ->617 ->730 ->586 ->688 ->497 ->853 ->101 ->286 ->240 ->479
    ->116 ->72 ->463 ->52 ->
582 ->592 ->219 ->603 ->548 ->605 ->482 ->468 ->108 ->495 ->578 ->67 ->211 ->707 ->368 ->948
    ->183 ->616 ->999 ->531 ->
224 ->704 ->178 ->673 ->850 ->679 ->685 ->942 ->903 ->188 ->442 ->476 ->417 ->323 ->232
    ->574 ->712 ->218 ->322 ->210 ->
589 ->701 ->124 ->180 ->354 ->212 ->828 ->455 ->608 ->54 ->96 ->738 ->575 ->520 ->681 ->237
    ->419 ->615 ->338 ->498 ->
430 ->460 ->340 ->614 ->832 ->863 ->624 ->859 ->975 ->678 ->197 ->585 ->951 ->241 ->454
    ->174 ->683 ->690 ->431 ->336 ->
691 ->164 ->869 ->590 ->200 ->792 ->82 ->689 ->121 ->247 ->206 ->932 ->179 ->611 ->992 ->411
    ->499 ->196 ->349 ->427 ->
235 ->980 ->555 ->857 ->231 ->652 ->974 ->804 ->658 ->371 ->814 ->525 ->970 ->700 ->729
    ->456 ->321 ->742 ->959 ->118 ->
158 ->866 ->335 ->671 ->459 ->316 ->361 ->80 ->76 ->483 ->372 ->541 ->448 ->346 ->473 ->401
    ->865 ->831 ->441 ->356 ->
409 ->314 ->443 ->919 ->559 ->86 ->209 ->333 ->713 ->57 ->732 ->675 ->467 ->558 ->360 ->743
    ->598 ->451 ->968 ->436 ->
547 ->243 ->797 ->933 ->719 ->97 ->291 ->330 ->205 ->229 ->435 ->801 ->602 ->822 ->997 ->24
    ->577 ->318 ->819 ->280 ->
926 ->581 ->159 ->623 ->669 ->594 ->394 ->576 ->429 ->703 ->226 ->399 ->740 ->868 ->641 ->65
    ->946 ->949 ->79 ->710 ->
648 ->193 ->568 ->561 ->898 ->874 ->413 ->51 ->806 ->278 ->190 ->613 ->659 ->777 ->115 ->432
    ->723 ->837 ->596 ->405 ->
696 ->584 ->470 ->422 ->204 ->283 ->213 ->840 ->420 ->161 ->469 ->345 ->718 ->793 ->735
    ->453 ->849 ->977 ->112 ->423 ->
653 ->421 ->74 ->798 ->661 ->109 ->562 ->606 ->351 ->534 ->445 ->963 ->982 ->325 ->23 ->549
    ->275 ->352 ->907 ->579 ->
176 ->583 ->518 ->369 ->165 ->242 ->25 ->524 ->546 ->790 ->290 ->904 ->353 ->61 ->744 ->324
    ->702 ->888 ->488 ->313 ->
36 ->667 ->810 ->303 ->955 ->64 ->173 ->98 ->107 ->311 ->391 ->619 ->543 ->538 ->168 ->845
    ->177 ->50 ->149 ->367 ->
686 ->281 ->916 ->900 ->815 ->774 ->833 ->952 ->203 ->47 ->87 ->216 ->114 ->677 ->893 ->996
    ->953 ->63 ->914 ->465 ->
739 ->299 ->39 ->44 ->587 ->300 ->145 ->139 ->565 ->83 ->580 ->21 ->674 ->35 ->181 ->856
    ->147 ->447 ->337 ->563 ->
983 ->825 ->824 ->69 ->390 ->680 ->820 ->38 ->17 ->571 ->144 ->113 ->198 ->305 ->331 ->34
```

```
    ->437 ->155 ->339 ->705 ->
365 ->32 ->457 ->921 ->960 ->821 ->184 ->567 ->967 ->634 ->160 ->816 ->185 ->846 ->233 ->544
    ->551 ->450 ->910 ->398 ->
138 ->402 ->157 ->201 ->781 ->428 ->891 ->536 ->302 ->950 ->400 ->557 ->320 ->319 ->539
    ->925 ->782 ->717 ->646 ->915 ->
663 ->37 ->965 ->55 ->166 ->167 ->307 ->277 ->920 ->306 ->418 ->785 ->654 ->146 ->208 ->392
    ->452 ->312 ->922 ->30 ->
131 ->553 ->406 ->434 ->182 ->408 ->49 ->175 ->388 ->163 ->805 ->140 ->527 ->78 ->315 ->892
    ->40 ->883 ->554 ->304 ->
289 ->396 ->542 ->521 ->966 ->26 ->189 ->397 ->927 ->666 ->779 ->969 ->835 ->789 ->909 ->803
    ->651 ->660 ->530 ->643 ->
152 ->458 ->517 ->773 ->170 ->662 ->295 ->802 ->18 ->68 ->151 ->862 ->28 ->20 ->640 ->58
    ->947 ->533 ->513 ->153 ->
766 ->416 ->102 ->668 ->537 ->639 ->393 ->512 ->839 ->787 ->650 ->415 ->796 ->599 ->362
    ->906 ->930 ->43 ->687 ->276 ->
913 ->908 ->929 ->135 ->807 ->13 ->172 ->772 ->137 ->215 ->410 ->762 ->272 ->682 ->642 ->404
    ->886 ->288 ->478 ->911 ->
811 ->895 ->53 ->148 ->765 ->550 ->523 ->791 ->526 ->941 ->569 ->136 ->630 ->310 ->94 ->293
    ->446 ->515 ->33 ->381 ->
142 ->794 ->256 ->282 ->656 ->426 ->382 ->287 ->931 ->387 ->545 ->693 ->768 ->670 ->890
    ->887 ->631 ->917 ->647 ->11 ->
645 ->912 ->378 ->637 ->813 ->150 ->894 ->519 ->130 ->609 ->514 ->27 ->784 ->884 ->858 ->279
    ->169 ->412 ->628 ->885 ->
386 ->836 ->389 ->529 ->265 ->761 ->655 ->31 ->657 ->129 ->273 ->540 ->905 ->901 ->355 ->292
    ->407 ->644 ->12 ->636 ->
75 ->156 ->414 ->776 ->141 ->260 ->296 ->143 ->257 ->878 ->379 ->19 ->897 ->879 ->881 ->899
    ->298 ->132 ->902 ->425 ->
882 ->380 ->385 ->285 ->635 ->383 ->528 ->125 ->384 ->627 ->788 ->516 ->509 ->154 ->918
    ->134 ->255 ->127 ->375 ->376 ->
759 ->308 ->7 ->769 ->270 ->10 ->504 ->266 ->8 ->756 ->508 ->42 ->764 ->377 ->195 ->817
    ->771 ->511 ->753 ->254 ->
29 ->16 ->66 ->633 ->783 ->128 ->294 ->799 ->795 ->22 ->444 ->625 ->775 ->889 ->786 ->522
    ->284 ->532 ->757 ->896 ->
268 ->2 ->6 ->14 ->767 ->263 ->41 ->60 ->507 ->126 ->763 ->262 ->778 ->267 ->133 ->875 ->755
    ->876 ->880 ->552 ->
638 ->274 ->770 ->269 ->502 ->510 ->503 ->506 ->271 ->632 ->758 ->780 ->877 ->629 ->259 ->45
    ->261 ->9 ->760 ->626 ->
649 ->754 ->505 ->535 ->253 ->0 ->264 ->3 ->752 ->5 ->750 ->751 ->258 ->15 ->1 ->252 ->251
    ->250 ->500 ->501 ->
4 ->
Time: 25.9395619044080377
```

4 Iteration

```
Length: 5659

1 ->265 ->148 ->843 ->966 ->718 ->535 ->336 ->776 ->472 ->208 ->827 ->151 ->31 ->916 ->197
    ->359 ->915 ->846 ->476 ->
417 ->421 ->872 ->120 ->929 ->339 ->452 ->225 ->291 ->647 ->217 ->464 ->432 ->185 ->75 ->51
    ->976 ->218 ->124 ->180 ->
54 ->243 ->994 ->325 ->282 ->689 ->254 ->738 ->366 ->166 ->161 ->711 ->593 ->89 ->717 ->899
    ->302 ->534 ->855 ->938 ->
666 ->741 ->764 ->102 ->99 ->312 ->747 ->238 ->413 ->576 ->429 ->986 ->815 ->645 ->724 ->168
    ->809 ->25 ->736 ->676 ->
387 ->948 ->700 ->260 ->389 ->989 ->920 ->223 ->435 ->437 ->408 ->457 ->412 ->658 ->913
    ->162 ->353 ->816 ->540 ->346 ->
370 ->252 ->453 ->561 ->844 ->491 ->244 ->833 ->935 ->713 ->517 ->859 ->975 ->383 ->769
    ->708 ->344 ->44 ->129 ->614 ->
117 ->438 ->607 ->329 ->665 ->152 ->735 ->782 ->473 ->581 ->599 ->481 ->202 ->943 ->349
    ->267 ->489 ->264 ->562 ->471 ->
632 ->621 ->431 ->497 ->115 ->0 ->73 ->840 ->420 ->69 ->157 ->6 ->281 ->555 ->898 ->549
    ->817 ->794 ->884 ->999 ->
860 ->245 ->894 ->712 ->591 ->703 ->360 ->780 ->767 ->153 ->781 ->428 ->459 ->122 ->173
    ->951 ->227 ->753 ->221 ->692 ->
142 ->272 ->8 ->998 ->959 ->460 ->554 ->836 ->983 ->618 ->326 ->246 ->198 ->634 ->887 ->262
    ->914 ->289 ->396 ->132 ->
199 ->556 ->657 ->502 ->610 ->671 ->337 ->258 ->615 ->158 ->970 ->24 ->182 ->678 ->511 ->328
    ->533 ->490 ->695 ->373 ->
```

```
597 ->59 ->320 ->829 ->411 ->509 ->979 ->219 ->295 ->485 ->771 ->654 ->636 ->88 ->492 ->602
    ->589 ->500 ->187 ->340 ->
266 ->82 ->100 ->945 ->759 ->971 ->826 ->830 ->41 ->249 ->936 ->922 ->877 ->17 ->441 ->35
    ->10 ->477 ->150 ->280 ->
558 ->322 ->637 ->222 ->778 ->910 ->352 ->135 ->680 ->341 ->777 ->729 ->60 ->950 ->196 ->463
    ->317 ->427 ->235 ->644 ->
932 ->480 ->97 ->381 ->362 ->415 ->131 ->768 ->499 ->2 ->210 ->5 ->722 ->440 ->191 ->977
    ->145 ->789 ->171 ->292 ->
309 ->631 ->486 ->236 ->84 ->912 ->766 ->978 ->573 ->677 ->681 ->209 ->744 ->40 ->193 ->123
    ->583 ->369 ->832 ->174 ->
799 ->46 ->83 ->574 ->824 ->315 ->116 ->660 ->564 ->167 ->307 ->313 ->103 ->433 ->867 ->61
    ->586 ->806 ->363 ->284 ->
204 ->93 ->140 ->775 ->685 ->942 ->375 ->550 ->523 ->847 ->954 ->176 ->160 ->953 ->125 ->504
    ->393 ->105 ->673 ->622 ->
469 ->345 ->585 ->495 ->807 ->114 ->119 ->294 ->572 ->487 ->907 ->733 ->95 ->321 ->189 ->624
    ->896 ->895 ->358 ->34 ->
740 ->852 ->802 ->400 ->314 ->211 ->172 ->968 ->965 ->955 ->770 ->392 ->967 ->828 ->287
    ->761 ->141 ->465 ->958 ->897 ->
888 ->566 ->361 ->80 ->165 ->21 ->587 ->548 ->605 ->659 ->808 ->739 ->964 ->719 ->841 ->311
    ->752 ->571 ->625 ->969 ->
835 ->793 ->404 ->380 ->324 ->672 ->68 ->669 ->594 ->33 ->350 ->330 ->48 ->261 ->568 ->892
    ->797 ->728 ->575 ->91 ->
918 ->923 ->306 ->584 ->374 ->908 ->937 ->748 ->300 ->447 ->444 ->144 ->419 ->251 ->259 ->20
    ->228 ->720 ->611 ->494 ->
110 ->862 ->516 ->368 ->842 ->274 ->530 ->293 ->85 ->376 ->449 ->763 ->949 ->397 ->643 ->755
    ->751 ->820 ->783 ->595 ->
532 ->790 ->290 ->904 ->220 ->127 ->206 ->462 ->714 ->800 ->342 ->704 ->569 ->112 ->917
    ->445 ->963 ->687 ->347 ->906 ->
933 ->200 ->522 ->118 ->310 ->94 ->333 ->406 ->921 ->960 ->995 ->757 ->939 ->143 ->401 ->508
    ->693 ->475 ->443 ->512 ->
263 ->804 ->686 ->382 ->524 ->398 ->318 ->372 ->541 ->442 ->107 ->214 ->706 ->407 ->869
    ->170 ->623 ->467 ->183 ->616 ->
612 ->812 ->984 ->332 ->716 ->335 ->760 ->190 ->9 ->996 ->849 ->106 ->885 ->874 ->285 ->911
    ->825 ->134 ->255 ->43 ->
14 ->709 ->792 ->601 ->155 ->526 ->941 ->870 ->821 ->164 ->28 ->886 ->230 ->866 ->154 ->30
    ->81 ->656 ->505 ->875 ->
70 ->316 ->410 ->592 ->529 ->537 ->175 ->834 ->878 ->458 ->364 ->520 ->446 ->304 ->63 ->927
    ->925 ->466 ->853 ->648 ->
426 ->414 ->109 ->737 ->663 ->883 ->257 ->609 ->619 ->27 ->542 ->626 ->159 ->23 ->765 ->646
    ->365 ->696 ->186 ->104 ->
11 ->384 ->664 ->461 ->439 ->579 ->130 ->371 ->837 ->596 ->405 ->608 ->108 ->850 ->111 ->902
    ->425 ->474 ->319 ->7 ->
434 ->268 ->448 ->334 ->745 ->205 ->864 ->754 ->725 ->296 ->940 ->518 ->515 ->86 ->684 ->848
    ->698 ->546 ->980 ->796 ->
565 ->991 ->256 ->338 ->181 ->823 ->377 ->403 ->750 ->928 ->418 ->74 ->882 ->479 ->87 ->879
    ->667 ->378 ->944 ->805 ->
773 ->726 ->839 ->961 ->715 ->386 ->470 ->670 ->721 ->355 ->18 ->283 ->514 ->52 ->357 ->604
    ->424 ->539 ->354 ->831 ->
901 ->356 ->547 ->327 ->727 ->560 ->652 ->974 ->588 ->67 ->394 ->881 ->234 ->58 ->856 ->78
    ->651 ->484 ->957 ->66 ->
286 ->863 ->399 ->498 ->4 ->226 ->348 ->96 ->743 ->578 ->981 ->865 ->510 ->232 ->962 ->212
    ->250 ->422 ->707 ->640 ->
756 ->784 ->379 ->288 ->661 ->65 ->194 ->871 ->128 ->617 ->730 ->241 ->454 ->603 ->818 ->600
    ->598 ->147 ->675 ->931 ->
876 ->758 ->620 ->22 ->26 ->668 ->909 ->854 ->772 ->137 ->993 ->683 ->787 ->13 ->430 ->861
    ->36 ->451 ->478 ->580 ->
409 ->303 ->810 ->507 ->195 ->299 ->697 ->628 ->531 ->893 ->423 ->563 ->934 ->557 ->146
    ->301 ->845 ->786 ->343 ->79 ->
731 ->710 ->699 ->3 ->92 ->924 ->240 ->705 ->559 ->694 ->702 ->276 ->536 ->582 ->297 ->545
    ->37 ->53 ->801 ->179 ->
229 ->649 ->184 ->803 ->247 ->468 ->237 ->192 ->49 ->779 ->270 ->762 ->822 ->732 ->838 ->90
    ->639 ->972 ->55 ->990 ->
450 ->956 ->385 ->734 ->72 ->513 ->98 ->77 ->177 ->496 ->662 ->56 ->629 ->798 ->570 ->688
    ->590 ->635 ->926 ->239 ->
691 ->742 ->32 ->653 ->544 ->19 ->42 ->905 ->16 ->29 ->538 ->985 ->982 ->903 ->188 ->819
    ->126 ->488 ->395 ->215 ->
567 ->331 ->136 ->503 ->231 ->101 ->45 ->39 ->121 ->627 ->650 ->416 ->746 ->76 ->483 ->543
    ->519 ->224 ->298 ->64 ->
```

```
12 ->606 ->690 ->930 ->613 ->216 ->242 ->674 ->501 ->71 ->527 ->858 ->279 ->169 ->305 ->988
    ->367 ->701 ->749 ->633 ->
388 ->723 ->391 ->552 ->641 ->278 ->946 ->233 ->814 ->525 ->271 ->795 ->630 ->785 ->203
    ->890 ->788 ->987 ->273 ->577 ->
992 ->873 ->521 ->642 ->277 ->38 ->163 ->275 ->682 ->50 ->149 ->390 ->774 ->851 ->551 ->323
    ->456 ->436 ->178 ->973 ->
248 ->15 ->889 ->506 ->47 ->638 ->493 ->900 ->528 ->997 ->791 ->813 ->62 ->952 ->455 ->213
    ->947 ->269 ->655 ->919 ->
133 ->553 ->139 ->253 ->811 ->402 ->891 ->482 ->351 ->156 ->113 ->308 ->138 ->880 ->679
    ->857 ->207 ->57 ->201 ->868 ->
1 ->
Time: 105.5457831178791821

Length: 6072

0 ->582 ->923 ->713 ->230 ->760 ->944 ->805 ->802 ->761 ->141 ->465 ->958 ->223 ->377 ->881
    ->234 ->444 ->971 ->826 ->
830 ->673 ->327 ->954 ->429 ->986 ->332 ->716 ->897 ->441 ->28 ->768 ->392 ->967 ->367 ->704
     ->474 ->319 ->708 ->959 ->
966 ->718 ->226 ->928 ->636 ->263 ->453 ->445 ->847 ->984 ->180 ->204 ->93 ->151 ->643 ->904
     ->797 ->411 ->322 ->894 ->
439 ->227 ->477 ->150 ->280 ->467 ->560 ->379 ->730 ->241 ->933 ->835 ->793 ->404 ->892
    ->736 ->438 ->329 ->665 ->152 ->
154 ->960 ->832 ->863 ->487 ->907 ->733 ->973 ->250 ->999 ->860 ->245 ->616 ->275 ->352
    ->763 ->312 ->747 ->836 ->983 ->
460 ->188 ->251 ->935 ->205 ->105 ->906 ->972 ->55 ->917 ->476 ->417 ->489 ->887 ->262 ->464
     ->84 ->912 ->378 ->561 ->
721 ->315 ->116 ->420 ->627 ->427 ->752 ->571 ->769 ->702 ->668 ->479 ->875 ->996 ->849
    ->977 ->145 ->265 ->148 ->466 ->
184 ->272 ->706 ->990 ->927 ->220 ->799 ->866 ->916 ->197 ->359 ->496 ->662 ->911 ->289
    ->925 ->132 ->864 ->290 ->397 ->
138 ->186 ->893 ->423 ->974 ->338 ->978 ->407 ->699 ->461 ->137 ->437 ->408 ->457 ->412
    ->422 ->306 ->475 ->937 ->753 ->
130 ->371 ->837 ->596 ->219 ->295 ->485 ->405 ->450 ->956 ->617 ->819 ->236 ->898 ->801
    ->901 ->448 ->434 ->413 ->576 ->
908 ->401 ->167 ->307 ->855 ->471 ->632 ->123 ->498 ->914 ->807 ->217 ->567 ->185 ->698
    ->463 ->317 ->255 ->711 ->727 ->
418 ->656 ->124 ->684 ->337 ->243 ->994 ->325 ->670 ->484 ->957 ->66 ->490 ->491 ->783 ->654
     ->335 ->389 ->161 ->889 ->
771 ->909 ->854 ->913 ->979 ->955 ->436 ->934 ->72 ->545 ->449 ->888 ->97 ->381 ->362 ->415
    ->41 ->369 ->206 ->462 ->
221 ->455 ->803 ->175 ->834 ->878 ->344 ->44 ->655 ->348 ->677 ->276 ->458 ->318 ->968 ->267
     ->873 ->915 ->846 ->282 ->
341 ->777 ->843 ->340 ->809 ->710 ->301 ->459 ->970 ->390 ->400 ->725 ->766 ->163 ->903
    ->300 ->114 ->841 ->304 ->63 ->
229 ->361 ->773 ->168 ->845 ->839 ->961 ->715 ->948 ->493 ->193 ->78 ->998 ->350 ->840 ->492
     ->509 ->478 ->657 ->46 ->
728 ->62 ->896 ->895 ->358 ->634 ->924 ->240 ->364 ->388 ->238 ->790 ->96 ->321 ->189 ->624
    ->817 ->387 ->470 ->199 ->
214 ->347 ->244 ->370 ->252 ->882 ->159 ->844 ->356 ->621 ->431 ->931 ->876 ->374 ->494
    ->922 ->868 ->179 ->176 ->160 ->
953 ->497 ->853 ->274 ->366 ->987 ->273 ->554 ->919 ->595 ->261 ->563 ->600 ->396 ->428
    ->940 ->140 ->880 ->774 ->373 ->
256 ->222 ->778 ->910 ->714 ->375 ->166 ->326 ->246 ->349 ->136 ->488 ->395 ->780 ->481
    ->745 ->945 ->759 ->303 ->368 ->
530 ->812 ->792 ->157 ->932 ->798 ->859 ->975 ->383 ->993 ->556 ->260 ->224 ->121 ->740
    ->852 ->618 ->264 ->363 ->248 ->
682 ->31 ->354 ->495 ->578 ->981 ->884 ->101 ->286 ->92 ->610 ->135 ->425 ->648 ->646 ->756
    ->564 ->104 ->687 ->902 ->
683 ->355 ->681 ->762 ->435 ->311 ->110 ->862 ->345 ->170 ->336 ->482 ->468 ->212 ->382
    ->108 ->850 ->947 ->867 ->676 ->
962 ->658 ->419 ->440 ->818 ->296 ->201 ->353 ->391 ->669 ->594 ->828 ->982 ->800 ->659
    ->808 ->231 ->969 ->729 ->741 ->
202 ->943 ->424 ->35 ->877 ->738 ->772 ->294 ->281 ->271 ->980 ->268 ->131 ->890 ->342 ->287
     ->607 ->333 ->992 ->660 ->
402 ->599 ->523 ->717 ->324 ->672 ->237 ->192 ->253 ->174 ->299 ->697 ->421 ->872 ->637
    ->748 ->963 ->25 ->285 ->842 ->
67 ->666 ->533 ->572 ->442 ->486 ->653 ->40 ->816 ->183 ->824 ->109 ->833 ->171 ->292 ->309
```

```
    ->472 ->208 ->538 ->247 ->
343 ->737 ->269 ->568 ->856 ->705 ->125 ->37 ->615 ->107 ->155 ->56 ->695 ->680 ->820 ->316
    ->870 ->821 ->164 ->278 ->
946 ->639 ->346 ->195 ->209 ->270 ->985 ->950 ->827 ->694 ->111 ->330 ->667 ->712 ->218
    ->858 ->279 ->719 ->254 ->293 ->
85 ->918 ->806 ->583 ->153 ->200 ->165 ->586 ->547 ->384 ->749 ->633 ->443 ->17 ->398 ->726
    ->394 ->591 ->703 ->951 ->
331 ->555 ->678 ->177 ->720 ->744 ->249 ->936 ->297 ->787 ->804 ->686 ->650 ->861 ->36 ->144
     ->113 ->976 ->764 ->689 ->
302 ->42 ->905 ->941 ->305 ->585 ->198 ->641 ->742 ->565 ->991 ->848 ->89 ->700 ->767 ->810
    ->172 ->791 ->886 ->73 ->
550 ->210 ->589 ->339 ->930 ->838 ->613 ->770 ->691 ->426 ->414 ->611 ->156 ->277 ->357
    ->604 ->597 ->989 ->920 ->60 ->
266 ->82 ->452 ->225 ->291 ->647 ->536 ->454 ->765 ->112 ->899 ->829 ->688 ->91 ->430 ->775
    ->365 ->696 ->584 ->360 ->
14 ->393 ->120 ->929 ->94 ->739 ->964 ->995 ->776 ->755 ->751 ->178 ->283 ->88 ->813 ->310
    ->320 ->671 ->447 ->789 ->
71 ->409 ->314 ->635 ->162 ->629 ->211 ->403 ->685 ->735 ->782 ->473 ->581 ->334 ->106 ->885
     ->874 ->19 ->652 ->187 ->
900 ->559 ->784 ->548 ->133 ->406 ->921 ->446 ->385 ->143 ->142 ->794 ->645 ->433 ->619
    ->541 ->127 ->640 ->601 ->734 ->
308 ->50 ->149 ->65 ->865 ->69 ->857 ->693 ->891 ->380 ->692 ->100 ->87 ->351 ->664 ->480
    ->785 ->575 ->22 ->259 ->
707 ->81 ->701 ->926 ->988 ->871 ->822 ->732 ->196 ->540 ->83 ->574 ->942 ->207 ->743 ->626
    ->169 ->469 ->376 ->74 ->
811 ->869 ->724 ->796 ->606 ->786 ->709 ->328 ->663 ->883 ->257 ->609 ->53 ->242 ->90 ->129
    ->614 ->456 ->64 ->284 ->
61 ->76 ->573 ->518 ->47 ->399 ->754 ->758 ->499 ->649 ->372 ->86 ->781 ->235 ->288 ->298
    ->598 ->722 ->126 ->851 ->
410 ->592 ->191 ->531 ->625 ->746 ->679 ->553 ->158 ->432 ->515 ->542 ->879 ->815 ->938 ->11
     ->386 ->115 ->181 ->823 ->
95 ->831 ->508 ->182 ->788 ->117 ->228 ->68 ->723 ->146 ->675 ->51 ->612 ->590 ->570 ->825
    ->451 ->642 ->949 ->79 ->
80 ->546 ->258 ->551 ->552 ->620 ->54 ->939 ->32 ->507 ->535 ->20 ->48 ->603 ->239 ->21 ->49
     ->779 ->147 ->173 ->
98 ->77 ->965 ->70 ->593 ->12 ->139 ->795 ->630 ->24 ->232 ->416 ->580 ->103 ->27 ->532
    ->814 ->525 ->605 ->9 ->
213 ->517 ->216 ->483 ->119 ->18 ->544 ->631 ->215 ->608 ->602 ->39 ->203 ->674 ->504 ->34
    ->190 ->521 ->233 ->511 ->
952 ->313 ->134 ->26 ->638 ->651 ->623 ->503 ->579 ->15 ->997 ->43 ->23 ->323 ->661 ->4
    ->128 ->52 ->16 ->543 ->
519 ->558 ->59 ->122 ->731 ->45 ->750 ->6 ->569 ->194 ->75 ->537 ->30 ->534 ->33 ->557 ->29
    ->516 ->58 ->577 ->
644 ->102 ->99 ->588 ->527 ->38 ->13 ->3 ->8 ->566 ->57 ->510 ->7 ->529 ->622 ->118 ->528
    ->524 ->526 ->10 ->
562 ->549 ->506 ->587 ->2 ->5 ->628 ->500 ->505 ->520 ->539 ->757 ->502 ->513 ->501 ->1
    ->514 ->522 ->690 ->512 ->
0 ->
Time: 71.5437925239093602

Length: 6970

0 ->282 ->968 ->237 ->602 ->212 ->727 ->686 ->135 ->972 ->319 ->482 ->226 ->746 ->479 ->745
    ->205 ->864 ->739 ->457 ->
204 ->571 ->232 ->962 ->658 ->738 ->195 ->478 ->657 ->46 ->728 ->489 ->213 ->932 ->480 ->930
     ->998 ->712 ->975 ->383 ->
715 ->948 ->493 ->491 ->197 ->645 ->934 ->979 ->219 ->239 ->145 ->707 ->721 ->714 ->671
    ->337 ->243 ->994 ->651 ->244 ->
615 ->914 ->211 ->999 ->860 ->992 ->660 ->402 ->453 ->445 ->717 ->899 ->709 ->127 ->729
    ->741 ->161 ->469 ->986 ->133 ->
665 ->152 ->476 ->248 ->942 ->936 ->993 ->913 ->708 ->959 ->966 ->718 ->949 ->79 ->425 ->474
     ->919 ->459 ->970 ->989 ->
464 ->432 ->185 ->698 ->463 ->697 ->421 ->722 ->440 ->191 ->224 ->121 ->740 ->166 ->716
    ->207 ->743 ->498 ->897 ->441 ->
443 ->798 ->859 ->225 ->981 ->193 ->449 ->477 ->958 ->223 ->377 ->194 ->496 ->662 ->748
    ->963 ->481 ->202 ->943 ->241 ->
933 ->725 ->348 ->460 ->455 ->711 ->471 ->385 ->675 ->484 ->710 ->229 ->361 ->403 ->685
    ->987 ->724 ->470 ->199 ->486 ->
```

```
412 ->973 ->683 ->286 ->240 ->439 ->227 ->154 ->960 ->681 ->209 ->703 ->951 ->705 ->405
    ->247 ->468 ->618 ->940 ->429 ->
187 ->206 ->462 ->221 ->230 ->706 ->990 ->450 ->956 ->495 ->438 ->861 ->437 ->181 ->649
    ->184 ->918 ->607 ->641 ->742 ->
338 ->978 ->407 ->699 ->461 ->137 ->164 ->982 ->720 ->744 ->953 ->497 ->656 ->427 ->171
    ->736 ->236 ->394 ->881 ->234 ->
732 ->695 ->680 ->140 ->880 ->996 ->160 ->925 ->466 ->672 ->388 ->723 ->391 ->422 ->629
    ->854 ->733 ->976 ->218 ->433 ->
888 ->730 ->472 ->208 ->392 ->967 ->220 ->354 ->663 ->216 ->483 ->435 ->627 ->485 ->346
    ->977 ->167 ->642 ->374 ->688 ->
637 ->222 ->596 ->612 ->376 ->666 ->632 ->123 ->173 ->734 ->475 ->937 ->416 ->494 ->922
    ->228 ->419 ->155 ->444 ->971 ->
826 ->605 ->928 ->418 ->980 ->401 ->238 ->413 ->148 ->203 ->890 ->99 ->73 ->840 ->162 ->353
    ->640 ->447 ->678 ->177 ->
670 ->694 ->111 ->737 ->92 ->492 ->509 ->245 ->894 ->594 ->488 ->691 ->426 ->924 ->409 ->587
    ->882 ->159 ->636 ->886 ->
404 ->689 ->134 ->893 ->684 ->623 ->467 ->910 ->901 ->448 ->175 ->692 ->142 ->969 ->417
    ->144 ->984 ->180 ->889 ->330 ->
667 ->905 ->941 ->954 ->938 ->151 ->465 ->719 ->97 ->378 ->944 ->157 ->659 ->83 ->473 ->581
    ->150 ->540 ->165 ->921 ->
446 ->233 ->909 ->661 ->129 ->614 ->456 ->321 ->189 ->939 ->907 ->669 ->620 ->935 ->713
    ->863 ->196 ->950 ->235 ->487 ->
735 ->58 ->856 ->679 ->847 ->916 ->217 ->931 ->876 ->704 ->174 ->299 ->701 ->120 ->991 ->638
    ->368 ->399 ->363 ->653 ->
871 ->633 ->214 ->920 ->350 ->170 ->163 ->903 ->188 ->974 ->349 ->136 ->957 ->857 ->693
    ->891 ->946 ->639 ->906 ->923 ->
867 ->101 ->912 ->141 ->190 ->613 ->90 ->342 ->186 ->452 ->442 ->454 ->603 ->333 ->926 ->866
    ->182 ->143 ->878 ->344 ->
955 ->436 ->560 ->652 ->367 ->327 ->873 ->915 ->178 ->149 ->792 ->911 ->106 ->885 ->917
    ->215 ->995 ->311 ->110 ->631 ->
800 ->700 ->524 ->398 ->726 ->839 ->961 ->326 ->246 ->198 ->634 ->887 ->904 ->109 ->562
    ->929 ->172 ->499 ->616 ->533 ->
572 ->406 ->410 ->375 ->875 ->70 ->593 ->624 ->896 ->458 ->558 ->112 ->892 ->408 ->650 ->322
    ->983 ->609 ->249 ->382 ->
428 ->68 ->988 ->927 ->451 ->617 ->819 ->853 ->122 ->297 ->545 ->687 ->644 ->332 ->183 ->424
    ->108 ->420 ->358 ->823 ->
95 ->597 ->566 ->848 ->89 ->583 ->153 ->200 ->900 ->884 ->387 ->430 ->296 ->201 ->306 ->584
    ->168 ->610 ->586 ->362 ->
415 ->41 ->621 ->431 ->622 ->601 ->635 ->852 ->588 ->370 ->125 ->829 ->411 ->131 ->359 ->702
    ->423 ->114 ->119 ->985 ->
357 ->604 ->643 ->317 ->81 ->371 ->648 ->646 ->365 ->696 ->690 ->400 ->845 ->947 ->146 ->301
    ->630 ->846 ->855 ->809 ->
834 ->313 ->851 ->668 ->964 ->339 ->380 ->682 ->31 ->107 ->91 ->72 ->309 ->179 ->176 ->366
    ->274 ->35 ->877 ->591 ->
836 ->850 ->158 ->396 ->132 ->85 ->130 ->598 ->945 ->543 ->611 ->156 ->369 ->832 ->595 ->830
    ->673 ->965 ->55 ->434 ->
113 ->749 ->600 ->44 ->390 ->555 ->898 ->801 ->585 ->599 ->345 ->52 ->582 ->626 ->169 ->93
    ->64 ->569 ->84 ->104 ->
872 ->329 ->98 ->902 ->62 ->849 ->813 ->310 ->674 ->592 ->74 ->811 ->869 ->654 ->86 ->576
    ->908 ->76 ->573 ->677 ->
574 ->664 ->628 ->49 ->75 ->51 ->731 ->45 ->575 ->277 ->676 ->833 ->997 ->822 ->96 ->334
    ->564 ->381 ->529 ->324 ->
47 ->87 ->67 ->105 ->147 ->50 ->395 ->27 ->379 ->831 ->804 ->550 ->210 ->589 ->868 ->100
    ->78 ->351 ->554 ->952 ->
373 ->559 ->414 ->816 ->570 ->820 ->625 ->824 ->126 ->606 ->315 ->116 ->806 ->883 ->318
    ->372 ->77 ->352 ->393 ->870 ->
821 ->772 ->294 ->59 ->579 ->874 ->19 ->790 ->71 ->818 ->39 ->115 ->561 ->844 ->356 ->547
    ->384 ->835 ->61 ->340 ->
862 ->771 ->347 ->842 ->298 ->29 ->538 ->23 ->331 ->289 ->335 ->389 ->118 ->590 ->553 ->139
    ->565 ->343 ->320 ->128 ->
293 ->841 ->843 ->242 ->316 ->341 ->386 ->812 ->355 ->302 ->56 ->364 ->520 ->82 ->22 ->793
    ->879 ->69 ->747 ->280 ->
65 ->805 ->802 ->761 ->544 ->551 ->552 ->536 ->269 ->568 ->567 ->794 ->782 ->325 ->30 ->770
    ->57 ->38 ->655 ->323 ->
784 ->548 ->305 ->556 ->619 ->312 ->138 ->63 ->783 ->779 ->808 ->231 ->307 ->539 ->837 ->865
    ->810 ->276 ->527 ->28 ->
577 ->360 ->328 ->304 ->94 ->124 ->279 ->827 ->530 ->777 ->526 ->490 ->807 ->518 ->36 ->542
    ->773 ->508 ->546 ->284 ->
```

```
308 ->291 ->103 ->300 ->18 ->54 ->799 ->825 ->797 ->858 ->16 ->791 ->608 ->796 ->557 ->88
    ->532 ->814 ->314 ->13 ->
275 ->66 ->895 ->53 ->767 ->803 ->42 ->580 ->21 ->290 ->795 ->40 ->774 ->780 ->17 ->336
    ->776 ->256 ->271 ->192 ->
33 ->775 ->283 ->756 ->828 ->287 ->292 ->43 ->14 ->34 ->268 ->6 ->281 ->263 ->763 ->24 ->285
    ->258 ->549 ->817 ->
762 ->272 ->781 ->788 ->117 ->270 ->778 ->267 ->257 ->261 ->48 ->80 ->10 ->259 ->253 ->397
    ->523 ->787 ->578 ->265 ->
255 ->768 ->838 ->26 ->278 ->647 ->525 ->25 ->60 ->515 ->264 ->512 ->522 ->519 ->504 ->32
    ->266 ->102 ->769 ->288 ->
37 ->20 ->521 ->757 ->528 ->531 ->766 ->815 ->510 ->8 ->765 ->563 ->273 ->260 ->15 ->507
    ->759 ->758 ->764 ->4 ->
295 ->12 ->5 ->9 ->3 ->262 ->786 ->750 ->7 ->11 ->250 ->303 ->760 ->251 ->541 ->254 ->789
    ->751 ->785 ->534 ->
516 ->252 ->755 ->511 ->517 ->513 ->752 ->506 ->754 ->1 ->2 ->753 ->502 ->503 ->537 ->535
    ->501 ->514 ->505 ->500 ->
0 ->
Time: 52.3425998152233660

Length: 8721

4 ->470 ->621 ->51 ->976 ->353 ->585 ->599 ->481 ->479 ->745 ->749 ->222 ->211 ->729 ->741
    ->352 ->355 ->860 ->245 ->
539 ->995 ->496 ->593 ->288 ->341 ->351 ->476 ->558 ->112 ->457 ->857 ->616 ->455 ->610
    ->586 ->205 ->229 ->238 ->591 ->
984 ->366 ->987 ->724 ->109 ->612 ->998 ->959 ->192 ->556 ->620 ->721 ->173 ->734 ->852
    ->227 ->89 ->717 ->93 ->475 ->
937 ->231 ->101 ->707 ->368 ->361 ->685 ->942 ->871 ->958 ->321 ->706 ->607 ->85 ->715 ->988
    ->246 ->349 ->493 ->492 ->
733 ->236 ->208 ->317 ->81 ->973 ->248 ->117 ->228 ->720 ->328 ->348 ->436 ->936 ->370 ->456
    ->811 ->869 ->604 ->597 ->
566 ->990 ->927 ->686 ->94 ->740 ->978 ->748 ->606 ->589 ->498 ->198 ->331 ->88 ->615 ->338
    ->576 ->679 ->847 ->934 ->
981 ->865 ->701 ->115 ->695 ->680 ->820 ->983 ->618 ->940 ->850 ->111 ->596 ->219 ->603
    ->337 ->595 ->817 ->842 ->921 ->
91 ->497 ->360 ->743 ->979 ->737 ->590 ->87 ->240 ->848 ->225 ->726 ->344 ->301 ->123 ->956
    ->736 ->438 ->977 ->491 ->
592 ->74 ->495 ->997 ->477 ->861 ->970 ->700 ->989 ->296 ->201 ->369 ->340 ->614 ->359 ->343
    ->313 ->624 ->831 ->488 ->
329 ->562 ->856 ->516 ->972 ->691 ->962 ->335 ->804 ->573 ->677 ->485 ->939 ->739 ->176
    ->122 ->480 ->97 ->243 ->994 ->
851 ->565 ->617 ->730 ->588 ->788 ->233 ->993 ->874 ->200 ->991 ->967 ->999 ->552 ->966
    ->182 ->316 ->339 ->863 ->623 ->
217 ->464 ->332 ->183 ->333 ->992 ->845 ->712 ->867 ->676 ->841 ->483 ->714 ->800 ->342
    ->704 ->569 ->194 ->209 ->564 ->
104 ->371 ->968 ->92 ->374 ->187 ->204 ->971 ->838 ->746 ->609 ->249 ->662 ->207 ->419 ->964
    ->237 ->930 ->819 ->601 ->
155 ->710 ->478 ->953 ->577 ->110 ->437 ->608 ->602 ->567 ->731 ->855 ->708 ->827 ->468
    ->212 ->829 ->99 ->572 ->223 ->
594 ->460 ->813 ->189 ->732 ->196 ->304 ->327 ->727 ->687 ->347 ->471 ->870 ->354 ->210
    ->334 ->106 ->486 ->107 ->214 ->
302 ->742 ->694 ->69 ->63 ->945 ->298 ->957 ->823 ->362 ->947 ->832 ->578 ->692 ->100 ->447
    ->345 ->854 ->474 ->319 ->
542 ->555 ->434 ->113 ->713 ->489 ->454 ->699 ->195 ->864 ->490 ->664 ->218 ->433 ->619
    ->541 ->916 ->197 ->120 ->929 ->
62 ->220 ->965 ->955 ->64 ->461 ->439 ->46 ->728 ->203 ->124 ->180 ->54 ->846 ->944 ->336
    ->482 ->363 ->472 ->417 ->
315 ->598 ->868 ->23 ->825 ->824 ->121 ->149 ->670 ->814 ->314 ->443 ->142 ->866 ->399 ->980
    ->581 ->655 ->234 ->802 ->
445 ->963 ->986 ->307 ->806 ->943 ->424 ->108 ->420 ->926 ->828 ->918 ->738 ->575 ->462
    ->346 ->923 ->583 ->681 ->985 ->
688 ->536 ->582 ->837 ->414 ->693 ->570 ->199 ->839 ->961 ->326 ->306 ->719 ->716 ->543
    ->611 ->494 ->553 ->221 ->901 ->
667 ->903 ->188 ->442 ->473 ->191 ->466 ->672 ->68 ->946 ->357 ->181 ->521 ->57 ->844 ->465
    ->171 ->446 ->605 ->928 ->
170 ->683 ->690 ->431 ->232 ->416 ->580 ->951 ->533 ->574 ->429 ->270 ->448 ->833 ->935
    ->428 ->167 ->239 ->906 ->247 ->
373 ->469 ->172 ->791 ->526 ->941 ->467 ->284 ->61 ->803 ->705 ->365 ->119 ->318 ->873 ->826
```

```
     ->645 ->584 ->53 ->226 ->
554 ->324 ->174 ->697 ->421 ->80 ->515 ->67 ->105 ->82 ->960 ->821 ->969 ->835 ->789 ->812
    ->674 ->853 ->152 ->735 ->
453 ->244 ->568 ->622 ->156 ->948 ->268 ->300 ->95 ->525 ->548 ->458 ->975 ->322 ->786 ->451
     ->917 ->215 ->561 ->872 ->
952 ->216 ->544 ->193 ->772 ->294 ->166 ->849 ->905 ->954 ->938 ->151 ->862 ->518 ->367
    ->794 ->157 ->418 ->98 ->77 ->
698 ->159 ->904 ->797 ->933 ->725 ->72 ->312 ->550 ->84 ->309 ->179 ->924 ->805 ->982 ->281
    ->175 ->484 ->308 ->282 ->
449 ->549 ->350 ->330 ->747 ->996 ->311 ->391 ->59 ->34 ->487 ->613 ->411 ->910 ->634 ->160
    ->185 ->711 ->787 ->66 ->
689 ->435 ->241 ->840 ->822 ->83 ->559 ->892 ->925 ->78 ->412 ->116 ->675 ->931 ->718 ->535
    ->834 ->783 ->118 ->158 ->
792 ->41 ->427 ->235 ->295 ->139 ->305 ->545 ->529 ->56 ->364 ->673 ->818 ->422 ->144 ->551
    ->323 ->190 ->900 ->8 ->
325 ->206 ->150 ->28 ->682 ->744 ->781 ->776 ->974 ->858 ->722 ->279 ->356 ->409 ->587 ->600
     ->894 ->230 ->42 ->949 ->
678 ->899 ->778 ->646 ->499 ->809 ->703 ->186 ->648 ->666 ->546 ->426 ->784 ->795 ->320 ->65
     ->830 ->440 ->257 ->290 ->
86 ->684 ->761 ->723 ->902 ->785 ->859 ->661 ->27 ->702 ->280 ->36 ->96 ->168 ->808 ->780
    ->291 ->897 ->441 ->163 ->
372 ->50 ->641 ->530 ->660 ->909 ->886 ->73 ->657 ->669 ->557 ->161 ->528 ->450 ->537 ->177
    ->769 ->415 ->76 ->639 ->
631 ->793 ->404 ->907 ->579 ->505 ->242 ->90 ->21 ->289 ->131 ->709 ->38 ->532 ->310 ->459
    ->297 ->398 ->810 ->47 ->
43 ->911 ->782 ->147 ->79 ->224 ->148 ->843 ->883 ->40 ->816 ->256 ->286 ->396 ->538 ->271
    ->762 ->164 ->765 ->563 ->
71 ->527 ->571 ->777 ->523 ->15 ->75 ->380 ->663 ->506 ->423 ->653 ->896 ->534 ->277 ->432
    ->509 ->452 ->303 ->262 ->
801 ->799 ->178 ->410 ->58 ->33 ->39 ->44 ->32 ->154 ->913 ->162 ->898 ->652 ->919 ->403
    ->102 ->659 ->285 ->912 ->
696 ->640 ->654 ->48 ->261 ->775 ->145 ->265 ->413 ->269 ->519 ->649 ->763 ->656 ->292 ->49
    ->895 ->45 ->522 ->358 ->
395 ->771 ->393 ->650 ->879 ->153 ->520 ->508 ->507 ->647 ->632 ->815 ->774 ->643 ->137
    ->514 ->950 ->884 ->511 ->184 ->
798 ->134 ->165 ->405 ->141 ->37 ->407 ->463 ->52 ->908 ->25 ->114 ->19 ->915 ->671 ->401
    ->274 ->267 ->202 ->767 ->
263 ->55 ->430 ->914 ->425 ->389 ->560 ->887 ->540 ->932 ->408 ->275 ->768 ->807 ->13 ->629
    ->293 ->406 ->264 ->213 ->
260 ->779 ->386 ->35 ->10 ->146 ->665 ->890 ->255 ->258 ->129 ->547 ->531 ->893 ->299 ->626
    ->796 ->143 ->773 ->658 ->
920 ->60 ->266 ->889 ->378 ->400 ->283 ->383 ->29 ->633 ->790 ->888 ->758 ->922 ->287 ->272
    ->140 ->70 ->31 ->11 ->
103 ->392 ->760 ->136 ->169 ->388 ->402 ->30 ->394 ->397 ->390 ->127 ->138 ->668 ->130 ->642
     ->638 ->14 ->510 ->382 ->
276 ->444 ->387 ->385 ->764 ->644 ->26 ->880 ->375 ->128 ->517 ->756 ->755 ->376 ->766 ->630
     ->24 ->651 ->636 ->836 ->
770 ->635 ->753 ->379 ->876 ->637 ->132 ->512 ->133 ->503 ->757 ->751 ->524 ->278 ->377
    ->881 ->126 ->885 ->891 ->381 ->
628 ->882 ->20 ->125 ->627 ->135 ->17 ->878 ->384 ->18 ->273 ->877 ->875 ->625 ->2 ->759
    ->502 ->5 ->6 ->253 ->
254 ->252 ->7 ->259 ->16 ->22 ->9 ->3 ->12 ->752 ->251 ->750 ->754 ->504 ->250 ->513 ->501
    ->500 ->0 ->1 ->
4 ->
Time: 51.3755521140992641
```

Determinant Problem - data tables for graphs

Runtime

Size:

Thread_Count	16	32	64	128	256	496	512	1000	1024	2000	2048	3000	4000	4096
1	4.17E-05	6.72E-05	2.98E-04	1.94E-03	1.47E-02	1.05E-01	1.16E-01	8.55E-01	9.16E-01	7.17E+00	7.73E+00	2.51E+01	5.94E+01	6.39E+01
2	1.59E-04	2.31E-04	4.45E-04	1.62E-03	8.95E-03	5.76E-02	6.39E-02	4.53E-01	4.89E-01	3.46E+00	3.74E+00	1.37E+01	3.42E+01	3.69E+01
4	1.98E-04	2.84E-04	4.62E-04	1.38E-03	5.21E-03	3.05E-02	3.38E-02	2.32E-01	2.51E-01	1.79E+00	1.88E+00	9.06E+00	2.64E+01	2.78E+01
8	3.01E-04	3.54E-04	5.40E-04	1.39E-03	4.07E-03	1.73E-02	1.91E-02	1.22E-01	1.27E-01	9.14E-01	9.57E-01	7.37E+00	2.35E+01	2.56E+01
16	4.85E-04	6.40E-04	8.67E-04	1.79E-03	3.83E-03	1.24E-02	1.32E-02	7.12E-02	7.66E-02	4.89E-01	5.25E-01	6.73E+00	2.37E+01	2.58E+01
32	2.25E-02	2.27E-02	1.86E-02	5.81E-01	9.92E-01	2.65E-02	2.12E+00	1.12E+00	1.24E-01	6.53E+00	3.02E+00	7.15E+00	2.40E+01	2.71E+01

Speedup

Size:

Thread_Count	16	32	64	128	256	496	512	1000	1024	2000	2048	3000	4000	4096
1	1.00E+00	1.00E+00	1.00E+00	1.00E+00	1.00E+00	1.00E+00	1.00E+00	1.00E+00	1.00E+00	1.00E+00	1.00E+00	1.00E+00	1.00E+00	1.00E+00
2	2.62E-01	2.91E-01	6.70E-01	1.20E+00	1.64E+00	1.82E+00	1.81E+00	1.89E+00	1.87E+00	2.07E+00	2.07E+00	1.84E+00	1.74E+00	1.73E+00
4	2.11E-01	2.36E-01	6.46E-01	1.41E+00	2.82E+00	3.44E+00	3.42E+00	3.69E+00	3.65E+00	4.00E+00	4.10E+00	2.77E+00	2.25E+00	2.30E+00
8	1.39E-01	1.90E-01	5.52E-01	1.39E+00	3.61E+00	6.08E+00	6.07E+00	7.01E+00	7.22E+00	7.84E+00	8.08E+00	3.40E+00	2.52E+00	2.50E+00
16	8.61E-02	1.05E-01	3.44E-01	1.09E+00	3.83E+00	8.46E+00	8.79E+00	1.20E+01	1.20E+01	1.47E+01	1.47E+01	3.73E+00	2.51E+00	2.48E+00
32	1.86E-03	2.96E-03	1.60E-02	3.35E-03	1.48E-02	3.96E+00	5.46E-02	7.61E-01	7.39E+00	1.10E+00	2.56E+00	3.51E+00	2.48E+00	2.36E+00

Efficiency

Size:

Thread_Count	16	32	64	128	256	496	512	1000	1024	2000	2048	3000	4000	4096
1	6.25E-02	3.13E-02	1.56E-02	7.81E-03	3.91E-03	2.02E-03	1.95E-03	1.00E-03	9.77E-04	5.00E-04	4.88E-04	3.33E-04	2.50E-04	2.44E-04
2	1.64E-02	9.09E-03	1.05E-02	9.35E-03	6.41E-03	3.68E-03	3.53E-03	1.89E-03	1.83E-03	1.03E-03	1.01E-03	6.12E-04	4.34E-04	4.23E-04
4	1.32E-02	7.38E-03	1.01E-02	1.10E-02	1.10E-02	6.94E-03	6.69E-03	3.69E-03	3.57E-03	2.00E-03	2.00E-03	9.22E-04	5.63E-04	5.60E-04
8	8.66E-03	5.94E-03	8.63E-03	1.09E-02	1.41E-02	1.23E-02	1.18E-02	7.01E-03	7.05E-03	3.92E-03	3.94E-03	1.13E-03	6.31E-04	6.10E-04
16	5.38E-03	3.28E-03	5.38E-03	8.50E-03	1.50E-02	1.71E-02	1.72E-02	1.20E-02	1.17E-02	7.33E-03	7.19E-03	1.24E-03	6.27E-04	6.06E-04
32	1.16E-04	9.25E-05	2.50E-04	2.61E-05	5.78E-05	7.99E-03	1.07E-04	7.61E-04	7.22E-03	5.49E-04	1.25E-03	1.17E-03	6.20E-04	5.76E-04

Figure 9: Data tables for the LU Decomposition analysis graphs.

Traveling Salesperson Problem - data tables for graphs

2. Ant Colony

Runtime
Iterations

Thread Count	1	2	4
1	26.45347	52.72592	105.5458
2	17.10598	34.82879	71.54379
4	12.9267	26.59231	52.3426
8	13.71773	25.93956	51.37555

Speedup
Iterations

Thread Count	1	2	4
1	1	1	1
2	1.546445	1.51386	1.475261
4	2.046421	1.98275	2.016441
8	1.928415	2.032645	2.054397

Efficiency
Iterations

Thread Count	1	2	4
1	1	0.5	0.25
2	1.546445	0.75693	0.368815
4	2.046421	0.991375	0.50411
8	1.928415	1.016323	0.513599

Figure 10: Data tables for the Ant Colony TSP analysis graphs.