CSCI 406: AlgoBOWL

Reminder: You are NOT allowed to consult the internet to solve this problem.

1 Problem Description

You are given n points $p_1, p_2, ..., p_n$ in three-dimensional space. Your task is to partition the n points into k sets so that the maximum distance between any two points in the same set is minimized. Your algorithm should use the Manhattan distance metric, which for two points (x_1, y_1, z_1) and (x_2, y_2, z_2), is defined as $|x_1 - x_2| + |y_1 - y_2| + |z_1 - z_2|$.

Input Format: The input will be provided in a text file. The first line contains the number of points, n. The second line contains the number of sets, k. The remaining n lines are the x, y, z coordinates of each point. Each line should contain 3 integers in the range [-1000, 1000] separated by a single space and the file should contain no extraneous spaces. If an integer is negative, the minus sign should not be separated from the number with any white-space, *e.g.* -10 is valid.

The following is a simple example input file, the comments below are for your understanding and should be omitted in an actual input file. This example is simple because all of the points are on the x-axis, which will not generally be true.

```
5   // n
2   // k
1  0  0  // first point
2  0  0  // second point
3  0  0  // third point
8  0  0  // fourth point
9  0  0  // fifth point
```

Let's assume that the points are named p_1, p_2, p_3, p_4 and p_5. They are to be divided into $k = 2$ sets based on the Manhattan metric. A possible partition is $\{p_1, p_3, p_5\}$ — $\{p_2, p_4\}$. The cost of this partition is 8 because the Manhattan distance between p_1 and p_5 is 8 and this is the largest distance between any pair of points in a set. Another possible partition is $\{p_1, p_2, p_3\}$ — $\{p_4, p_5\}$. It's cost is 2 which is the Manhattan distance between p_1 and p_3. Clearly this second partition is better than the first.

Input Restrictions:

- $3 \leq n \leq 1000$ (i.e., any input contains at least 3 and at most 1000 points).
- $2 \leq k \leq 20$ (i.e., the number of sets is between 2 and 20).
- $k < n$ (i.e., the number of sets must be less than the number of points)
- The coordinates of all points are integers in $[-1000, 1000]$. (Note that this means that the Manhattan distance between any pair of points will also be an integer.) No two points can have the same coordinates.

Output Format: The output is a text file consisting of $k+1$ lines: Line 1 of your output will contain the maximum distance between any two points in a set. Line 2 will contain point IDs assigned to the first set, separated by a single space, and Line 3 will contain point IDs assigned to the second set, likewise separated. In general, the kth set will be on the $k+1$th line. **Note: point IDs are one-indexed, in other words, the first point from the input file should have an ID of 1, not 0. Be careful, this is a common programming mistake as many languages use zero-indexed arrays**.

To illustrate, the output for the second split in the example is

```
2          // Maximum distance between any two points in a set
1 2 3      // Set 1
4 5        // Set 2
```

Again, your actual output file should not have comments or extraneous spaces.

Note: This problem is NP-complete, which means that it is unrealistic to expect that your algorithm will compute an optimal solution in a reasonable time frame. Please keep this in mind as you work on this project.

Good Luck!!!