INDIAN STATISTICAL INSTITUTE

MTech(CS) I year 2020-2021

Subject: Computing Laboratory

Assignment 3

Total: $3 \times 20 = 60$ marks Deadline: 07:00am, 27 March, 2021

SUBMISSION INSTRUCTIONS

1. You may consult course material, or material from other Internet sources; you may also discuss the problems and their possible solutions with your classmates. You are permitted to use any code provided to you as part of the course material, but apart from this exception, **all code must be written by you without looking at anyone else's code**. Please acknowledge all sources that you take help from.
2. You may write your programs in C/C++ or Python.
3. Please make sure that your programs adhere strictly to the specified input and output format. **You may lose marks if your program violates the input and output requirements.**
4. Please adhere to the file naming conventions discussed in class (i.e., your file names should be of the form `cs20XX-assign3-progY` with a `.c` or `.py` extension). Note that your file names **do not** need to contain a 6-8 character long password like alphanumeric string.
5. Please upload your programs to `https://www.dropbox.com/request/OuiXqlxYZkguRbxBanSi`.

NOTE: Unless otherwise specified, all programs should take the required input from stdin, and print the desired output to stdout.

Q1. Let $P_1, P_2, \ldots, P_n$ be n points on the plane such that no three of them are collinear. Recall that collinear points are points that lie on the same line. Let us denote the slope of the line segment P_iP_j (connecting P_i and P_j, where $i < j$), by $m_{i,j}$. Write a program to identify the pair of points P_i, P_j for which $|m_{i,j}|$ is maximum (i.e., P_iP_j will be the steepest among all the line segments connecting the given points). You may assume that no two segments have the same slope.

Input Format

Input will be provided via standard input, and will start with a single positive integer n on the first line, followed by n lines, each of which will contain 2 floating point numbers corresponding to the X and Y coordinates of a point P_i $(1 \leq i \leq n)$.

Output Format

If the steepest line segment is P_iP_j $(i < j)$, your program should print the two indices i and j on a single line, separated by one or more spaces. Note that the points are indexed starting from 1, not 0.

Sample Input 0

```
3
1 1
2 2
2 3
```

Sample Output 0

```
2 3
```

Sample Input 1

```
5
0.1 1.2
2.1 2.2
2.4 3.7
0.2 0.5
3 5
```

Sample Output 1

```
1 4
```

Sample Input 2

```
3
0.1 0.2
0 0.3
-0.1 -0.2
```

Sample Output 2

```
2 3
```

Q2. Consider a nearly sorted array A of n distinct integers. Every element in A is at most k positions away from its actual location in the sorted form (in ascending order) of A. Write a program that sorts the said array in ascending order in $O(n \log k)$ time. For example, if we consider $k = 2$, an element at index 7 in the sorted array, can be at indices 5, 6, 7, 8, 9 in the given array A.

Input Format

Input will be provided via standard input. It will start with a pair of positive integers n and k on the first line. The next line will contain n distinct integers separated by spaces.

Output Format

The elements of the array are to be printed in ascending order to standard output.

Sample Input 0

```
7 3
6 5 3 2 8 10 9
```

Sample Output 0

```
2 3 5 6 8 9 10
```

Sample Input 1

```
8 4
10 9 8 7 4 70 60 50
```

Sample Output 1

```
4 7 8 9 10 50 60 70
```

Sample Input 2

```
4 1
1 3 2 4
```

Sample Output 2

```
1 2 3 4
```

Q3. Recall that Huffman coding is a variable-length, prefix-free coding scheme that seeks to assign shorter bitstrings to more frequently occurring characters. See *Algorithms*, Fourth Edition, Sedgewick and Wayne, Chapter 5, pp. 826–838 for a detailed discussion of Huffman coding.

- Write a program that encodes a given text file using Huffman coding.
- Also write a matching decoder program that decodes the encoded file to produce the original text file.

Input/output format:
Your programs should take the names of the input and output files as the first and second command-line arguments respectively. For the encoder program, the input will be a plain text file; the encoded contents of the input file should be written to the output file. Conversely, for the decoder program, the input will be an encoded file generated earlier using the encoder program, and the output will be a plain text file. The example below shows how your programs should be run.

```
$ ./cs20XX-assign3-prog3a input.txt input.enc          # Encoding
$ ./cs20XX-assign3-prog3b input.enc output.txt         # Decoding
```

In the above example, the encoder program takes `input.txt` (a plain text file) as input, and writes its contents in Huffman-encoded form to `input.enc`. The decoder program takes `input.enc` as input, decodes it, and writes the resulting plain text to the file `output.txt`.

If you have written your programs correctly, the files `input.txt` and `output.txt` should be identical. If you have the `diff` utility available on your computer, you may run the following command to check whether the files are indeed identical.

```
$ diff -qs input.txt output.txt
```

`diff` will report whether the files are identical or different.

You may assume that the input text file will contain only printable, ASCII characters (i.e., characters for which the C library function `isprint()` returns a non-zero value).

- We will use text files containing Python code or literary text (e.g., the one available from `https://www.dropbox.com/s/q6koolyqmefte07/tom-sawyer.txt?dl=1`) to test your program.
- **3–5 bonus marks** will be given to a correct program if it either
  (a) produces, on average, the smallest output file from among all submissions, or
  (b) has the lowest average running time among all submissions.