Matrix Chain Multiplication

Linear Algebra and Random Processes (CS6015)
Assignment 2 Input/Output Specifications

Your program (matrix_chain_mul.c/matrix_chain_mul.cpp) written in C/C++ programming language, should read the files given under input section and generate the files specified under output section. All the files should be present at the same directory level.

1 Input

1. input.txt - File containing the input matrices and its dimensions as described in the problem statement

2 Output

1. output.txt - File containing the outputs described in the problem statement

3 Input File Format

The first line of the input contains a single integer n denoting the number of matrices in the chain.
The second line contains $(n+1)$ space-separated integers which corresponds to p_i defined in the problem definition.
The description of each matrix A_i in the chain follows.
The matrix A_i is decribed in p_{i-1} lines, where each of these lines contains p_i space-separated floating point numbers denoting a row of the matrix A_i.

3.1 Sample

3
1 2 3 4
1.0 2.0
1.0 2.0 3.0
4.0 5.0 6.0
1.0 2.0 3.0 4.0
5.0 6.0 7.0 8.0
9.0 10.0 11.0 12.0

4 Output File Format

This file contains five lines.
The first line contains the optimum pairing for calculating the product $A_1A_2...A_n$ expressed by using paranthesis along with the index i of the matrix A_i. No whitespace should be included in this line.
The second line contains one integer - the number of scalar multiplications to compute the given product $A_1A_2...A_n$ by performing pairing from left to right.
The third line contains one floating point number - the computation time for calculating the product by performing pairing from left to right (in seconds).
The fourth line contains one integer - the optimum number of scalar multiplications to compute the product $A_1A_2...A_n$
The last line contains one floating point number - the computation time for calculating the product using optimal pairing (in seconds).

The time elapsed can be computed using the following code snippet.

```
#include <time.h>

clock_t start, end;
double cpu_time_used;

start = clock();
... /* Code to be timed. */
end = clock();
cpu_time_used = ((double) (end - start)) / CLOCKS_PER_SEC;
```

4.1 Sample

$((A1A2)(A3))$
18
0.948
18
0.948