For this project, you will write a program in C++ that implements the maximum subsequence sum functions discussed in class.

Your code must be modular and use functions, but not classes, templates, or vectors. Create a separate function for each of the four maximum subsequence sum algorithms from the slides/text, and place them all in a single source file named ss.c.

In the main function, read in the values of the subsequence to an array of integers using the cin object. The maximum number of integers entered is guaranteed to be no greater than 2500. Each function should then be called in the order given in the slides. Modify the interface to each function so that it receives parameters for the array and the number of integers in the array. Each function should return a double, indicating the number of microseconds required for the algorithm to find the sequence.

Each function should process the array as indicated in class, but with timing code added. To capture the run time, a call to the linux clock function in time.h should be made before the outer loop begin to record the start time (for algorithm 3, record the start time just before the recursive function is called in the driver). Once the loop ends, call clock again to find the end time. To compute the run time, find the difference between these two times and scale it by CLOCK_PER_SEC, which is machine-specific. This value can then be multiplied by 1,000,000 to compute microseconds.

At the end of each function, the maximum sum should be printed before returning the number of microseconds. The main function should print the times for each algorithm after all the functions have completed. You can check the output of your program against the results posted on the course webpage. To print large values without scientific exponents, you can use std::fixed and std::setprecision(0) (need to include iomanip).

Your program can be compiled with the command:

```bash
g++ mss.cpp -o mss
```

..and run with the command:

```bash
./mss < test8.txt
```