You should **submit** an single electronic package that contains the source files for your work on the programming questions, by *following the submission procedure described in class and on the class newsgroup.*

You are welcome to use any inanimate resources (e.g., books, Web sites, publicly available code) to help you with your work. However, *all such help must be clearly noted* in your submissions. Further, no matter what you use, *you must be able to explain, in detail, how it works.* (You may be called upon to explain your homework individually.) Refer to the class policy for details, and ask for clarifications if you are unsure if something is allowed.

Questions marked with a ⋆ are optional but you are strongly encouraged to answer them for extra credit.

Your implementation must use clean, portable C++ that conforms to the C++17 standard and that minimizes dependencies. (If in doubt, please ask.) Packaging and documentation of code are worth a very significant portion of the grade.

Further details will be provided on the class newsgroup.

1. (30 pts.) Implement a program that is similar to the *Minesweeper* helper program described by Exercise 1.13 of the textbook,[1], but with the following differences and clarifications:
   - Instead of reading the input from a file as described there, this program should read its input from *standard input.*
   - It should write its output to *standard output.*
   - It should write any diagnostic messages to *standard error.*
   - The input does not begin with the two numbers denoting the number of rows and number of columns. Instead, the input consists of only the grid of characters that describes the minefield. (The number of rows and columns can be determined from this grid.)
2. (20 pts.) Implement a one-dimensional variant of the above program, which reads a one-row, n-column grid of characters that describes a one-dimensional minefield. The required output is a similar one-dimensional variant of the output of the earlier program.
3. (30 pts.) Implement a three-dimensional variant of the above program. The input format is based on slicing a three-dimensional cube along the third dimension using planes that are parallel to the first two. The input consists of the sequence of grids representing these slices (as above) with a single blank line separating adjacent grids (slices). The output consists of the a similar sequence of slices.

[1]Mark Allen Weiss, *Data Structures and Problem Solving in C++*, 2nd edition (Pearson, 1999).

4. (20 pts.) Write a two-page report documenting your program and, in particular, highlighting the interesting parts, parts that are incomplete or buggy, parts that were easy, difficult, etc. (Documented bugs will diminish their negative impact on the score.)

5. (20 $\star$ pts.) Develop, document, and implement a d-dimensional variant of the above program, where d is an arbitrary (finite) dimension of the input grid. Part of the work here is determining, justifying, and documenting the appropriate input and output formats in addition to the core computation. This work must be discussed on the class newsgroup well in advance of the submission date in order to qualify for any credit.