

EECS 498 W26 System Design of a Search Engine

02 String and vector

Due Sunday, January 25, 2026, 11:59 pm

The purpose of this assignment is to help you get started creating some of the basic templates you will need to build your search engines, starting with string and vector.

Your templates only need to support the specified minimum functionality. But many teams will decide to do more, recognizing it's likely to make them more productive later.

Starter files

You are given these starter files in `StringVectorStarter.zip`.

<code>string.h</code>	Starter file for a basic string template.
<code>Vector.h</code>	Starter file for a basic vector template.

The assignment

Fill in the necessary code in the `string.h` and `vector.h` header files to match the RME comments. Submit these files to the autograder at <https://autograder.io/web/project/3487>.

The starter files provided do not have any tests or sanity checks. You are highly encouraged to write your own.

You may not use the standard library except `std::cout`, `std::cin`, `std::endl`, `std::ostream`, `std::move` and `std::initializer_list` from the headers `<iostream>`, `<utility>`, and `<initializer_list>`. `<new>` is not required to pass the autograder tests but may be used.

Do not attempt to use the standard library to trivialize the assignment.

You will be tested on basic and edge case functionality, memory handling, and performance. There are no private test cases.

Enhancements

Many teams decide that doing more just the minimum needed to satisfy this assignment will make them more productive later. Here are some examples of extensions you might consider.

1. Automatic tests (Continuous Integration) to ensure your library isn't broken by "small changes."
2. Insert and Erase functions as seen in the standard library.

3. Bounds checking on indexing operations, i.e., throw an exception when accessing out of bounds.
4. `.find()` methods to find the index or iterator to a given element in a vector or string.
5. A method to find the index or iterator to a substring within a given string.
6. A method to split a string based on a delimiter or predicate.
7. A method to join a vector of strings with a separator.
8. Use your Vector to implement a Queue, Stack, Deque etc.