CSE 326 Spring 1995

Data Structures
Project 1
March 31, 1995

DUE: Friday, April 7, 1995 in class

Objectives

- To become familiar with developing programs under UNIX.
- To modify a simple program which uses most of the useful features of C++.
- To teach the basics of using the debugger.

Project Directory

All of the materials you will need are in `/cse/courses/cse326/project1`.

What to Do

1. You will need your own copy of the program to modify, so make a copy of the materials in the project directory above. You probably want to put the copied files into their own subdirectory of your home directory.

2. Most of the code for the project has been given to you. All that is missing is the implementation of the `Reverse` method for the `String` class. Your job is to write an implementation of `Reverse` and place it in the `String.C` file. Note that a prototype for this method already exists in the class specification in `String.h`.

   In order to get an exposure to template classes, I want you to use the provided `Stack` class in your implementation of `Reverse`. Although it is not strictly necessary, using a stack will simplify your code as well as help you gain more experience in using the language.

3. Once you have added the implementation of `Reverse` to `String.C` using Emacs or your favorite editor, you should try to compile your code by running `make` in your project directory. If your code compiles, you will end up with an executable called `string` in the same directory; to run it type `string`. If you've implemented your function correctly, you should get the following output:

   ```
   Hello
   Segmentation fault (core dumped)
   ```

Segmentation fault? Oops! Silly me, I guess I made a mistake somewhere in the code I gave you.

4. Your final task is to figure out what is causing the error in the program. Type

```
gdb string
```

to run the debugger on the program. Type `run` (or '`r`') at the `(gdb)` prompt to begin executing the program. The debugger will tell you where the error occured; use `where` to figure out how the program got to that point. Your job at this point is to figure out what is causing the error in the program by using the debugger and examining the code. You do not have to correct the error; you only need to figure out specifically what the problem is.

NOTE: The programming required for this project is not difficult. My solution took 15 lines including whitespace and comments. The difficulty lies in familiarizing yourself with the programming environment and, if you have not seen C++ before, the language. I strongly encourage you to start early so that you'll be able to get help if you have trouble with something.

What to Turn In

Since the point of this assignment is not really to produce a working program, we will not be using the electronic turnin system as we will be later in the quarter. Instead, the first thing you must hand in is a hardcopy of your `Reverse` routine (I don't need any of the other material because you shouldn't have changed it). The other thing I want is a specific description of the cause of the error in the program (i.e., what routine is failing and what is wrong with it). You do not have to provide a corrected version of the routine, although you can write one for practice if you would like.