MEDGAR EVERS COLLEGE
The City University of New York

Project:
Memory Program

Directions:

For your computer program, you will need to create memory so that the computer can store programs and data. Memory is addressed storage locations all with a fixed size (word size). The memory for your computer will be stored in a file that must consist of only 32-bit binary strings (lines of 32 characters consisting only of zeroes and ones) with each binary string on its own line.

You will be creating a **Memory** class that

- must have a private **Bus** pointer field.
- must have either or both a private fstream object and/or private string object that will be used to access the memory file.
- must have a private void method named `Write()` that takes no parameters. It must be used to write data into the memory file at the specified address (line number).
- must have a private void method named `Read()` that takes no parameters. It must be used to read data from the memory file at the specified address (line number) to the bus.
- must have a default constructor that assigns NULL to the **Bus** pointer object and an empty string (or an empty file to) to the string object (or fstream object).
- must make the copy constructor and assignment operator private. As before, only their prototypes need to be declared.
- may have overloaded constructors that assigns the **Bus** object and memory file.
- must have public set methods for the **Bus** object and the memory file.
- must have a public bool method named `Valid()` that takes no parameters. It must only return true if a **Bus** is assigned (**Bus** field is not null), and the memory file can open and is in a valid format (nonempty file of 32-bit binary strings).
- must have a public void method named `Process()` that takes no parameters. If the **Memory** object is in a valid state, it analyzes the **Bus** object and determines which method, `Write()` or `Read()`, to call. This means it interprets the control line to determine if a signal is generated for either a write, read, or neither.

When creating your class, you need to add documentation that states what the read and write signals are (the documentation may be a comment in the header file). Furthermore, when a read or write is performed the control line should be updated with a new signal. The signal(s) can indicate that the read/write was successful or failed, or the signal could reset the bus; this signal(s) must be included in the documentation. Meanwhile, the writes and reads do not have to be direct (written to or read from the memory file immediately). Likewise, remember any constraints mentioned before must be maintained.

Last, besides creating the class, you must write a test program that creates a **Bus** object and **Memory** object. Next, it should modify the **Memory** object so that it is valid, but make sure its bus is the **Bus** object. Afterwards, perform at least two writes and two reads by modifying the **Bus** object. Additionally, provide the memory file that is being modified by the program.