STL Assignment

You are given a text file that contains the name and an integer ID of a student.
The student entries are comma separated, and within a student entry, the name and integer ID is separated by a semi-colon.
Example:

John Doe;1234,James Bond;7,
Tracy Brown;177,
Jack Smith;172, Jim Black,TimWhite;88

You can assume that a student entry is not broken up across a line. However, if there is a student entry without the ID (like Jim Black in the example above), then your code should assign some default ID (of your choosing), print an error message "Student <name> did not have an ID", and continue processing other names in the input.

1. You will need to define a simple Student class which stores the name (use `std::string` class for the name) and the integer ID.
2. When you read the file and create the Student objects, these Student objects should be added to a vector of Student objects.

   Now that all the Student objects read from the input file are in this vector, your code should create a vector of vector of Student (`vector < vector< Student> > vecVecStudents`) where `vecVecStudents[ ii ]` is a vector of Student objects that start with the same letter. In other words, each vector inside the outermost vector contains students whose name starts with the same letter (case insensitive).
3. For the example above, you would end up with two vectors inside the outermost vector since the names start with either 'J' or 'T'.
4. You can use the same samples given above, but if you create your own sample entries, make sure to satisfy the below conditions
   a. All names should not start with the same letter.
   b. At least one name should not have an ID (like Jim Black in the example above).
5. Finally, print all the names to standard output. For example:

```
int numVecs = vecVecStudents.size();
for (int ii = 0; ii < numVecs; ++ ii )
{
   PrintVec(vecVecStudents[ ii ] );
}
```

   Where PrintVec takes the vector of Student and prints the name of each student to standard output.

6. Organize your code properly into functions / classes such that it is easy to read… I know that is a vague requirement, but this is where you have to use your judgment, and this is what you exercise in the real world.