Contestant Number: _____

Time: _____

Rank: _____

C++ PROGRAMMING
(335)

REGIONAL – 2017

Production Portion:

Program 1: Natural Language Processing: Named Entities	_____	(350 points)
TOTAL POINTS	_____	***(350 points)***

Failure to adhere to any of the following rules will result in disqualification:

1. **Contestant must hand in this test booklet and all printouts. Failure to do so will result in disqualification.**
2. **No equipment, supplies, or materials other than those specified for this event are allowed in the testing area. No previous BPA tests and/or sample tests or facsimile (handwritten, photocopied, or keyed) are allowed in the testing area.**
3. **Electronic devices will be monitored according to ACT standards.**

No more than ten (10) minutes orientation
No more than ninety (90) minutes testing time
No more than ten (10) minutes wrap-up

Property of Business Professionals of America.
May be reproduced only for use in the Business Professionals of America *Workplace Skills Assessment Program* competition.

Natural Language Processing: Named Entities

Have you chatted with Apple Siri, Google Now, Amazon Alexa or Microsoft Cortana? These amazing intelligent assistants employ Natural Language Processing (NLP). This is a leading edge field of computer science and artificial intelligence, concerned with the interactions between computers and human languages. Programmers like you are enabling computers to derive meaning from human or natural language input, as well as generate human language. For this exercise, you will use computer language (C++) to process human language!

1. Write a program that reads written natural language from provided file "human_jabber.txt". Your program will identify paragraphs, sentences and words. Words are separated by spaces, sentences by periods, and paragraphs are delimited by newlines ("\n"). Hint: most punctuation except periods can be discarded.

2. Your program will also read "named_entities.txt". This is a list of *proper nouns* which are often just capitalized words. Use it to identify named entities.

3. Your program will save to "output.csv" what was parsed (example below for format).

4. The program will output a total count to the **screen** of named entities, words, sentences and paragraphs (example below).

5. If the same word or named entity occurs again in the input, count it again. A name like "Paul Bunyan" counts as two named entities.

6. Congratulations! You've processed text in a way that a program like Siri can begin to interpret.

Steps

1. Build a reusable "readFile" function (to read input files), a "parser" function (to identify paragraphs, sentences, words and named entities) and a "writeFile" function to write the output file. Output totals to screen. The program should gracefully handles improper or missing input files, as well as ignore extra whitespace, punctuation and symbols.

2. The program will read files "human_jabber.txt" and "named_entities.txt" and output formatted csv, generated from the data structure.

Sample Input and Output:

1. Here is an example input file `human_jabber.txt`:

```
I am from Minnesota.  Paul Bunyan lives here.
Florida is warmer.  I might move.
Prince was from here so it's cool.
```

2. The file `named_entities.txt` contains:

```
Minnesota
Paul
Bunyan
Prince
Florida
```

3. Example `output.csv` shown. The output contains csv columns for word #, paragraph #, sentence #, type (word or namedEntity), and parsed word.

```
paragraph, sentence, type, word
w1, p1, s1, word, I
w2, p1, s1, word, am
w3, p1, s1, word, from
w4, p1, s1, namedEntity, Minnesota
w5, p1, s2, namedEntity, Paul
w6, p1, s2, namedEntity, Bunyan
w7, p1, s2, word, lives
w8, p1, s2, word, here
w9, p2, s3, namedEntity, Florida
w10, p2, s3, word, is
w11, p2, s3, word, warmer
w12, p2, s4, word, I
w13, p2, s4, word, might
w14, p2, s4, word, move
w15, p3, s5, namedEntity, Prince
w16, p3, s5, word, was
w17, p3, s5, word, from
w18, p3, s5, word, here
w19, p3, s5, word, so
w20, p3, s5, word, it's
w21, p3, s5, word, cool
```

4. The program will output this summary to the screen:

```
Words: 21
Named Entities: 5
Sentences: 5
Paragraphs: 3
```

5. You will have 90 minutes to complete your work.

6. Your name or school name should NOT appear on any work you submit for grading.

Development Standards

- Consistent naming should be used for variables and code.
- Classes, methods, and functions must be documented with comments explaining the purpose, the input parameters (if any), and the output (if any).

Your application will be graded on the following criteria:

Solution and Project	
Custom code is present	____ 10 points
All classes and methods/functions are customized	____ 10 points
Program Execution	
Program runs	____ 20 points

If program does not execute, then remaining items receive *partial credit* if credible code exists.

The program gracefully handles empty, improper or missing input files	____ 10 points
The program reads "human_jabber.txt" into a data structure	____ 15 points
The program reads "named_entities.txt" into a data structure	____ 15 points
The program saves "output.csv" containing dynamically generated csv	____ 15 points
The program outputs correct totals at end	____ 30 points
The "output.csv" correctly counts Words, Paragraphs and Sentences	____ 15 points
The "output.csv" has correct Words identified	____ 15 points
The "output.csv" has Named Entities correctly identified	____ 15 points
The program ignores input "," and parenthesis and doesn't add to csv	____ 10 points
The program correctly handles paragraph, sentence and word delimiters	____ 10 points
The program correctly handles (ignores) extra white space	____ 10 points

Source Code Review

Class code is commented, for each method, and as needed	____ 15 points
Code uses reasonable and consistent variable naming conventions	____ 15 points
The program contains well-formed function for readFile	____ 25 points
The program contains well-formed function for parser	____ 25 points
The program contains well-formed function for writeFile	____ 25 points
Processing exists for counting and displaying totals	____ 15 points
The program has punctuation processing	____ 10 points
The program has whitespace processing	____ 10 points
Code exists to trap for file errors	____ 10 points

Total Points: _____ / 350 points