CSCI 3415 – Principles of Programming Languages
Fall 2014 – Dr. Williams
Programming Assignment #2

Ada is a structured, statically typed, imperative, wide-spectrum, and object-oriented high-level computer programming language, extended from Pascal and other languages. It has built-in language support for explicit concurrency, offering tasks, synchronous message passing, protected objects, and non-determinism. Ada is an international standard; the current version (known as Ada 2012) is defined by ISO/IEC 8652:2012.

Part I – Install an Ada Compiler

The GNU Compiler Collection (GCC) includes front ends for C, C++, Objective-C, Fortran, Java, Ada, and Go, as well as libraries for these languages. Therefore, it can be used for many of the programming assignments for this class. The GCC web site is http://gcc.gnu.org.

The GCC is available for many platforms. It is a standard part of almost all Linux distributions. I use Ubuntu 10.04 LTS, which is available from http://www.ubuntu.com. But, you could use any Linux distribution or other platform.

If you don't have access to a Linux system, you can install it under a virtual machine on either Microsoft Windows or Apple OS X. I use Oracle VirtualBox available from https://www.virtualbox.org. After VirtualBox is installed, you can create a new virtual machine and install the Linux distribution of your choice.

For Microsoft Windows, you can also download the GNU Ada Translator (GNAT) GPL from the AdaCore web site http://www.adacore.com.

You may, of course, use any Ada compiler you choose.

Part II - Social Characteristics of the Marvel Universe

For this programming assignment, we will be using a data set from the Social Characteristics of the Marvel Universe web page(s). Specifically, the input file (porgat.txt) defines the social network we will examine.

Here is what the input file looks like.

```
*Vertices 19428 6486
1 "24-HOUR MAN/EMMANUEL"
2 "3-D MAN/CHARLES CHAN"
3 "4-D MAN/MERCURIO"
4 "8-BALL/"
5 "A"
6 "A'YIN"
7 "ABBOTT, JACK"
8 "ABCISSA"
9 "ABEL"
10 "ABOMINATION/EMIL BLO"
```

```
…
6484 "STORMER"
6485 "TIGER WYLDE"
6486 "ZONE"
6487 "AA2 35"
6488 "M/PRM 35"
6489 "M/PRM 36"
6490 "M/PRM 37"
6491 "WI? 9"
6492 "AVF 4"
6493 "AVF 5"
…
19426 "AA2 30"
19427 "AA2 20"
19428 "AA2 38"
*Edgeslist
1 6487
2 6488 6489 6490 6491 6492 6493 6494 6495 6496
3 6497 6498 6499 6500 6501 6502 6503 6504 6505
4 6506 6507 6508
5 6509 6510 6511
6 6512 6513 6514 6515
7 6516
8 6517 6518
9 6519 6520
10 6521 6522 6523 6524 6525 6526 6527 6528 6529 6530 6531 6532 6533 6534 6535
10 6536 6537 6538 6539 6540 6541 6542 6543 6544 6545 6546 6547 6548 6549 6550
10 6551 6552 6553 6554 6555 6556 6557 6558 6559 6560 6561 6562 6563 6564 6565
…
6484 18709
6485 15336
6486 15336
```

The "`*Vertices 19428 6486`" line says there are 19,428 total vertexes in the input file and that the first 6,486 of them are character vertexes and the remaining are comic book vertexes. Each vertex line contains the vertex number and a string with the character name or comic book title. The "`*Edgeslist`" line begins the edge list. Each edge line contains a source (character) vertex and one or more (apparently up to 15) target (comic book) vertexes.

Part III – Representing the Social Graph

Since the graph is a fixed size (and not too large) and all of the references are already integer indexes, an array representation is probably the most natural to use. To represent the information in the input file, we can use one array of strings to hold the vertex names – character names and comic book titles – and another two dimensional array of Booleans to hold the edges. [Due to the nature of the graph, there are large blocks of it that will always be false – for example, there are no character to character edges or comic book to comic book edges – and it is symmetric. Therefore, we could only store part of the graph if we wanted to save space.]

So, you need to be able to read and parse the input file and create the array of vertex names and the edge matrix.

Part IV - Marvel Universe looks almost like a real social network

There is a paper on arXiv.org titled Marvel Universe looks almost like a real social network that looks at the structure of this network. Here, two characters are considered linked if they jointly appear in some comic book.

We can make a collaboration matrix, which is a square (or diagonal if we prefer) matrix of integers where each cell (i, j) is a count of the number of comic books in which the pair of characters have jointly appeared. We can compute this by taking the i^{th} and j^{th} row (or column), anding them together to get a vector of Booleans where each true value is an instance where the two characters appear in the same comic book, then summing the number of true values as the value of the cell (i, j). If you apply this to every pair of characters, then the diagonal will contain the counts of the number of comics that each character occurs. Using this collaboration matrix, we can calculate the total number of collaborations and the number of collaborating pairs of characters. [Be careful to exclude the diagonal from the counts – we don't consider a person as collaborating with themselves – and symmetry – if x collaborates with y then that is stored twice (so essentially, we divide the final count(s) by 2.]

Part V – Statistics

For this programming assignment, we want to compute statistics on the Marvel Universe social network. At a minimum, we want to compute:

number of characters
number of comic books
minimum characters per comic book
maximum characters per comic book
average characters per comic book
minimum comic books per character
maximum comic books per character
average comic books per character
total number of collaborations
number of collaborating pairs
mean number of collaborators per character

Part VI – Specifics

Each student will be assigned to one of eight (8) teams. Each team will have approximately five (5) members – as it currently stands, there are 39 students registered for the class. So, one team will only have four (4).

Team 1	**Team 2**	**Team 3**	**Team 4**
David Andrews	Robert Fitzgerald	Matthew Lamont	Greg Peters
David Attid	Andy Gale	Jian Luo	Julio Soto
Cory Coellen	Carlos Goncalves e Silva	Ja Oh	Bandit Tanyaratsrisakul

Herman Colato	Steve Kosovich	Christopher Padgett	Vuong Tran
Kyle Etsler	Sean Kuhlman	Robert Perry	Chinh Vong

Team 5	**Team 6**	**Team 7**	**Team 8**
Tyler Cosner	April Hudspeth	Eric Nguyen	Alexander Six
Nicole Cranon	Michael Hunsinger	Raphael O'Flynn	Ian Wiggins
William Daniels	Long Huynh	Gabriella Ramirez	Anna Zhylava
Zack Donnelly	Jason Minor	Paul Rodriquez	Joseph Zurawski
Amos Gurung	Ivan Mora	Kyle Ryan	

The teams may arrange themselves however they please. But, each person must contribute to the overall effort.

Each team will develop and document an Ada social network statistics program as described above. Each team will turn in a team report describing their design and implementation. Also, all source code must be provided.

Each student will also turn in a short report on how they contributed to the team effort – for example what part of the program they developed, tested, documented, etc. They should also grade the performance of the other team members.