Concordia University
Department of Computer Science
and Software Engineering

Comparative Study of Programming Languages
COMP 6411 --- Winter 2014

Programming assignment #1

Deadline:	Wednesday January 29th, 2014
Evaluation:	10% of final mark
Late submission:	not accepted

Problem statement

You have to implement three different programs involving graph theory. Each algorithm must be first explained and presented abstractly using pseudo-code. You have to explain why each particular problem was chosen as a good candidate to eventually demonstrate statistical differences in execution using different programming languages. Each of the chosen algorithms must be implemented in two different programming languages (for a total of six programs). Each implementation must comply with the presented pseudo-code. The algorithms and the programming languages you use are left to your discretion, given the following restrictions:

For the programming languages used, you must choose one from each of the following two lists:

C, C#, Python, ML, Smalltalk

Java, Scheme, Fortran, C++, PHP, JavaScript, Haskell, Perl, Ada, Objective-C, AspectJ, Ruby, VisualBasic, Prolog, Pascal.

The problems to be programmed are left to your choice and may include the following:

- The traveling salesman problem
- The shortest path algorithm
- Minimum spanning tree
- Breadth-first search

In order to enable better comparison and testing of the programs, each implemented algorithm must read the problem's data from a file, and output the result to another file, along with various statistics on e.g. the execution time, memory consumption, etc that will eventually enable you to make a comparison of the execution of the different algorithms implemented in different programming languages. You will be judged also on the difference of languages used, e.g. using two object-oriented languages is unlikely to lead to interesting conclusions, but using languages belonging to different programming language paradigms is likely to lead to conclusions that can be more clearly explained.

In order to measure the execution times, memory consumption, as well as other factors that you find relevant, you have to either instrument your own programs or use a third-party solution to measure externally the execution. No matter what measuring solutions you are using, it must not interfere with the execution of the programs in a way as to invalidate the measurements. The same measuring technique or tools must be used throughout in order for the comparisons to be valid. It is up to you to figure out what you are measuring and how. You are expected to

conduct a series of runs using different data sets in order to be able to draw some curve diagrams showing different execution times and memory consumption over a significant range of number of graph nodes (e.g. 100, 1,000, 10,000, 100,000, 1,000,000 nodes).

Assignment submission requirements and procedure

You have to submit your assignment before midnight on the due date using the ENCS Electronic Assignment Submission system under the category "*programming assignment 1*". Late assignments are not accepted. The file submitted must be a **.zip** file containing:

- all your code (i.e. six different programs)
- one input file containing data to be read by the programs
- one output file for each program containing the result, as well as relevant statistics on the execution (minimally: execution time and memory consumption)
- a simple document containing:
  - the pseudo-code of the three algorithms implemented
  - instructions on how to compile and execute all your programs
  - presentation of the execution results in tabular and graph form
  - explanations as to why the two experiments lead to different results, i.e. what characteristics of the two programming languages and their implementation lead to differences in execution for the same algorithms.
  - a list of references used to gather information about the two languages' implementation details.

You are also responsible to give proper compilation and execution instructions to the marker in a README file. If the marker cannot compile and execute your programs, you might have to have a meeting for a demonstration.

Evaluation Criteria

Correctness of pseudo-code description, and implementation vs. pseudo-code	10 pts
Explained relevance of measurements, algorithms and test cases enabling interesting comparison	10 pts
Explained relevance of chosen languages likely to lead to differences in results	10 pts
Presentation, analysis and explanations of results	10 pts
References	10 pts
Total	**50 pts**