Université d'Ottawa
Faculté de génie

École de science
d'informatique
et de génie électrique

uOttawa

L'Université canadienne
Canada's university

University of Ottawa
Faculty of Engineering

School of Electrical
Engineering
and Computer Science

Assignment 0

CSI2120 Programming Paradigms

Winter 2020

Part 1 and 2 due on February 17th, 2020 before 11:00 pm in Virtual Campus

Part 3 and 4 due on April 4th, 2020 before 11:00 pm in Virtual Campus

[12 marks]

<u>Problem Description</u>

This assignment asks you to implement solutions to the stable matching problem. We will consider the problem of matching coop employers to coop students. In particular, we consider the problem of n employers and n students where each employer will hire a single student. Each employer produces a ranking of students who they prefer to hire for the coop term and each student rank all employers who they prefer to work for. As input to the stable matching algorithm, we therefore have two preference tables of size $n \; x \; n$: one for the employers and one for the students. This input will be stored as two csv (comma-separated-values) files and a simple example for $n = 3$, i.e., for three employers and three students in a very simple format.

Coop employer preferences:

```
Thales,Olivia,Jackson,Sophia
Canada Post,Sophia,Jackson,Olivia
Cisco,Olivia,Sophia,Jackson
```

Student preferences:

```
Olivia,Thales,Canada Post,Cisco
Jackson,Thales,Canada Post,Cisco
Sophia,Cisco,Thales,Canada Post
```

A stable matching solution for this problem is as follows

```
Pair: Canada Post - Jackson
Pair: Thales - Olivia
Pair: Cisco - Sophia
```

Why is this solution stable? We have to look at the definition of a stable match. A stable match is defined such that neither party to the match have a preferred party that also prefers them over their current match. E,g., Canada Post would prefer to hire Olivia over Jackson but Olivia is currently matched with Thales which she prefers over Canada Post. While Jackson would prefer to work for Thales over Canada Post but Thales is matched with Olivia which Thales prefers over Jackson. As a result, while neither Canada Post nor Jackson got their first choice, the match is stable as neither of them have a way to improve their current match. The other pairs are also stable which is left as an exercise to test. As a result, the solution provided is a stable matching and is also perfect. A perfect match is actually a simpler criterion, just requiring each employer being matched to a student and each student being matched to an employer.

In summary, in this assignment you will need to find a perfect and stable matching given preference tables by coop employers and by students. There will always be the same number of employers and students and every employer will only hire one student.

Algorithms:

The stable matching can be found with an iterative algorithm, the Gale-Shapley algorithm. The corresponding pseudocode is given below. The input is a list of preferences from n employers $L_{\{employer\ preferences\}}$ and a list of preferences from n students $L_{\{student\ preferences\}}$. The algorithm calculates an output of n stable matches M. In the algorithm the variable e stands for an employer and s for a student.

Gale-Shapley ($L_{\{employer\ preferences\}}$, $L_{\{student\ preferences\}}$.)

Initialize $M := \{\}$

`while` (some employer e is not matched to any student)

find most preferred student s on the list $L_{\{employer\ preferences\}}(e)$ to whom the employer e has not yet offered a job.

`if` (student s is unmatched)

Add the pair to the set of matches $(e, s) \rightarrow M$.

`else if` (s prefers e to employer e' of current match (e', s))

Replace the match $M \rightarrow (e', s)$ with $(e, s) \rightarrow M$

`else` s rejects offer from e

`return` the set of stable matches M

While Gale-Shaley is an iterative solution, the recursive McVitie-Wilson will be easier to implement in some paradigms. One can think of McVitie-Wilson as an alternating recursion of two functions: a function `offer` and `evaluate` (see the pseudocode on the next page). These two recursive function need to be called from a main loop which calls `offer` for each coop employer once.

Initialize $M := \{\}$

```
offer ( employer e )
    if (employer e is unmatched)
        find most preferred student s on the list L_{employer preferences}(e) to whom
                                        the employer e has not yet offered a job.
        if found evaluate match (e, s) by calling evaluate((e, s))
    return

evaluate ( match (e, s) )
    if (student s is unmatched)
        Add the pair to the set of matches (e, s) → M.
    else if (s prefers e to employer e' of current match (e', s) )
        Replace the match M → (e', s) with (e, s) → M
        offer(e')
       else     s rejects offer from e
                offer(e)
    return
```

Part 1: Object-oriented solution (Java) **[3 marks]**

Create the classes needed to solve the stable matching problem for coop employers and students with the iterative Gale-Shapley algorithm. Your program must be a Java application called `StableMatching` that takes as input the names of two files containing the preference of coop employers and students as csv files (*in this order*). Your program must save the stable matching to a csv file called `matches_java_`**`n`**`x`**`n`**`.csv` where **`n`** is the size of in the problem. The file is to be saved in the current directory.

In addition to the source code, you must also submit a UML class diagram showing all classes, their attributes, methods, and associations. You cannot use static methods (except `main`).

You must follow proper object-oriented design for full marks and hand-in your source tree in a single jar file. Your jar file must include all source code (`*.java`).

Part 2: Concurrent solution (Go) **[3 marks]**

Create a Go application that solve the stable matching problem for coop employers and students with the recursive McVitie-Wilson algorithm. Your program must produce a Go executable called `stable_matching.exe` that takes as input the names of two files containing the preference of coop employers and students as csv files. Your program must save the stable matching to a csv file called `matches_go_`**`n`**`x`**`n`**`.csv` where **n** is the size of in the problem. The file is to be saved in the current directory.

Your solution must execute the function `offer` for different employers concurrently.

You must follow proper imperative and concurrent design for full marks including the creation and use of packages. You must hand-in your source tree in a single zip file. Your zip file must **not** include any executable code but only the source code in the proper directory structure.

Part 3: Logic solution (Prolog) **[3 marks]**

Create a Prolog predicate

```
stableMatching( L_employer_preference, L_student_preference, M )
```

that is true if the stable matching problem for coop employers and students is solved by the set of matches M.

Your solution must include helper predicate called `findStableMatch` that takes as input the names of two files containing the preference of coop employers and students as csv files. Your helper predicate must save the stable matching to a csv file called `matches_prolog_`**`n`**`x`**`n`**`.csv` where **n** is the size of in the problem. The file is to be saved in the current directory.

Your solution for the predicate `stableMatching/3` must function as a predicate with all three parameters instantiated but also find **all** stable matching results for full marks.

Part 4: Functional solution (Scheme) **[3 marks]**

Create a Scheme function `stableMatching` that solves the stable matching problem for coop employers and students with the recursive McVitie-Wilson algorithm. With the above preference tables of size 3, the function would be called as follows:

```
(stableMatching L_employer_preferences L_student_preferences)
```

⇨ `(("Canada Post" "Jackson") ("Cisco" "Sophia") ("Thales" "Olivia"))`

Your solution must include helper functions for your function `findStableMatch` to be able to take as input the names of two files containing the preference of coop employers and students as csv files. Your helper predicate must save the stable matching to a csv file called `matches_scheme_`**`n`**`x`**`n`**`.csv` where **n** is the size of in the problem. The file is to be saved in the current directory.

Your solution must follow proper functional design for full marks.