CSCE476/876 **Fall 2021**

Homework 2: Programming Assignment (Generic Version) Missionaries & Cannibals

Assigned on: Friday, September 10th, 2021

Due: Friday, September 17th, 2021

Contents

The goal of this assignment is to introduce you to the concept of Allen Algebra, and to bolster your understanding of the Farmer's Dilemma problem discussed in class by extending its solution to the Missionaries and Cannibals puzzle. You are free to choose any programming language you wish to implement a solution to the puzzle.

This homework is structured as follows:

- Section 1 introduces Allen's time relations. (20 points)
- Section 2 is the Missionaries and Cannibals puzzle itself. (60 points)

1 Allen's Time Relations (20 Points)

This section of the homework deals with time intervals, which are the building blocks for temporal reasoning. For more background on the subject, goto pages 448 and 449 of AIMA. Figure 1 introduces all 13 possible qualitative relationships that may exist between two intervals. These relations are called Allen relations for qualitative temporal reasoning after James Allen who identified them in his seminal paper [**?**].

You are asked to implement data structures that represent the intervals and methods to determine whether or not the predicates below hold. Turn in a file (via handin) called *allen.ext* (where *ext* is the extension for the language you chose, and a file *readme-allen.txt* that describes the data structures and functions used to complete this portion of the assignment, as well as the steps required to compile and run your program.

1. Implement a data type called `TimePoint` that has one data member, which is an integer (representing seconds).

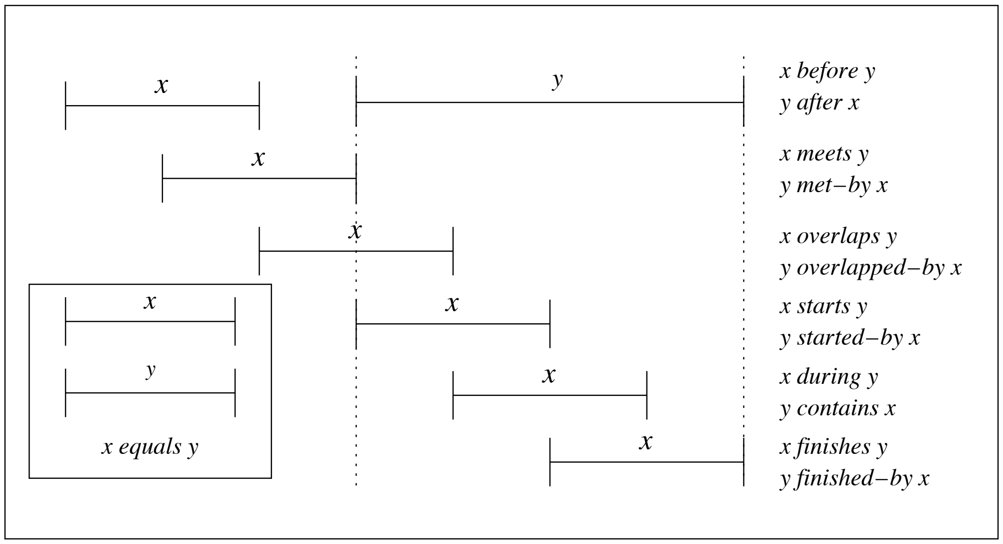

Figure 1: Predicates on time intervals

2. Implement a data type `Interval` that has the following data members: `beginTime`, and `endTime` which are of type `TimePoint`.

3. Write methods `Start` and `End` that take an instance of `Interval` and return the data members `beginTime` and `endTime`, respectively.

4. Write methods that implement the predicates listed below and illustrated in Figure 1, which take as input two objects of type `Interval` and return whether or not each of the following predicates holds.

   - *Meet(i,j)* $\Leftrightarrow$ *End(i) = Start(j)*
   - *Before(i,j)* $\Leftrightarrow$ *End(i)* $<$ *Start(j)*
   - *After(i,j)* $\Leftrightarrow$ *Before(j,i)*
   - *During(i, j)* $\Leftrightarrow$ *Start(j)* $\leq$ *Start(i)* $\bigwedge$ *End(i)* $\leq$ *End(j)*
   - *Overlap(i,j)* $\Leftrightarrow \exists$ *k During(k,i)* $\bigwedge$ *During(k, j)*
   - *Equals(i,j)* $\Leftrightarrow$ *Start(i) = Start(j)* $\bigwedge$ *End(i) = End(j)*
   - *Finishes(i,j)* $\Leftrightarrow$ *End(i) = End(j)*
   - *Contains(i,j)* $\Leftrightarrow$ *Start(i)* $<$ *Start(j)* $\bigwedge$ *End(i)* $>$ *End(j)*

Additionally, you should write a series of method calls that tests each of the above predicates with the intervals $i \leftarrow [5, 10]$ and $j \leftarrow [7, 13]$. When ran, your program should display the predicate or method being tested, its inputs, and the result.

2 Missionaries & Cannibals puzzle (60 Points)

The problem is defined as follows:

Three missionaries and three cannibals must cross a river using a boat that can carry at most two people at a time. All six people start on one bank of the river, and the goal is to find a series of boat rides that results in everyone on the second bank, with the following restrictions:

- The boat cannot cross the river with no one on board
- The cannibals on either bank cannot outnumber the missionaries on that bank (lest the cannibals eat the missionaries)

Write a program that solves this puzzle using recursion, and displays each of the boat rides used to solve the puzzle when ran. Turn in a file called *missionaries.ext* where *ext* is the extension used by your programming language, and a file *readme-missionaries* that details how to compile and run your program, and a description of how your program operates (functions, data structures, flow etc.).