CS 635 Advanced Object-Oriented Design and Programming
Fall Semester, 2021
Assignment 4
Assignment Index

San Diego State University -- This page last updated 9/27/21

In-Memory Database with Persistence

Due Dec 13

We will use a Chain-of-Responsibility like solution to finding substrings in a string with wild characters. We will use the characters “*” and “.” for wild characters. The character “.” matches any single character. The character “*” matches zero or more characters. So “c.t” with match “cat”, “cbt”, etc. While “c*t” with match “ct”, “cat”, “caat”, “cbt”, “casdert” and many others. Now a Match object will find the first match of a pattern in a string. For example in:

```
match = new Match( “c.t” );
startIndex = match.findFirstIn( “cacacat”);
```

startIndex will be 4 (zero based indexing). Now it is not too hard to implement a pattern matcher, or to use regular expressions. The goal is to implement the Chain-of-Responsibility, so do not use regular expressions or other built-in pattern matchers in your language. For practice we will use a chain of objects to do the matching. There will be one object in the chain for each character in the pattern. I will use “ca.t” to illustrate. The chain for this pattern will contain at least four objects. The first object (head) in the chain needs to find the first occurrence of the character c in the target string and pass the target string and the location in the string to the next object in chain. If the next object in the chain reports success, then the head of the chain returns the location of the pattern. If the next object reports failure to match, the head object needs to find the second occurrence of c in the string. The head object will try all occurrences of c in the string until a match is found or it is determined that the pattern is not in the string. The second object in the chain behaves slightly differently. It is given a position and the string from the head object. If the next character is an “a” it passes the request to match to the next object in the chain. It then reports to the head object the result returned by the rest of the chain. If the next character is not an “a” it returns failure to the head chain.

You will need to think about the how to implement the objects that handle the wild characters. The "." character is straight forward. The "*" behaves differently. Note the two following cases.

```
match = new Match( “c*t” );
assertEquals(0, match.findFirstIn( “cacacat”))

match = new Match( “c*tb” );
assertEquals(2, match.findFirstIn( “bacatatb”))
```

Also there are several ways to handle the end of the chain.

Grading

Item	Percent of Grade
Working Code	20%
Unit Tests	10%
Proper implementation of Patterns	50%
Quality of Code	20%

What to Turn in

Turn in your project on the Course Portal.

Late Policy

An assignment turned in 1-7 days late, will lose 5% of the total value of the assignment per day late. The eight day late the penalty will be 40% of the assignment, the ninth day late the penalty will be 60%, after the ninth day late the penalty will be 90%. Once a solution to an assignment has been posted or discussed in class, the assignment will no longer be accepted. Late penalties are always rounded up to the next integer value.