Sprint Evaluation

In this sprint we have been planning to write a parser for our regex engine. We have mostly done this. Our parser takes input as a string, converts it into a form, which we call normalized form, that we are going to use for regex resolver. Normalized form of a string is basically a form; that only contains words, parentheses, and "|", "&", "*" as operands. The need of such a conversion stems from the structure of the parse tree which is to be embodied by the postfix form of this infix normalized form. Taking normalized form for granted, we have been able to morph it into postfix form and parse given query regex into terms subsequently. There are some missing parts which are explained in backlog updates part. We have also needed some basic data structures. Therefore, we have implemented a stack, a queue and a binary tree class. Furthermore, we have also started to construct the parse tree for the terms which are going to be matched with the documents including them through an index structure.

During this term we had some problems that we want to point out in this document. Firstly, we had to remember the key features of the data structures that we were going to use. Therefore, we worked on them briefly. Secondly, we had some difficulties on adjusting precedence of regex characters. We also had difficulties on replacing regex operands and special characters to obtain normalized form of it when we were parsing. This could have been done easily by using more than one function or iteration over the input string. However, having been conscious of the dramatic decrease in the performance of the program led us adopting the tricky but correct approach of parsing the string at once.

Team evaluation

In this sprint our cooperation was great. We generally used pair programming technique in this sprint. Every Deadliner helped each other and for some difficult tasks, we coded together. We have met four times a week. We are not planning to change our strategy for the time being.

T1: Stack Implementation
T2: Queue Implementation
T3: Binary Tree Implementation
T4: Regex infix to postfix transformation
T5: Construction of parse tree over postfix form
T6: C++11 unique_ptr learning

Task	Assigned Member	1st week	2nd week	3rd week
T4, T5, T6	Fatih Burak BELCE	T1	T4	T5, T6
T1, T5, T6	Mustafa GÜVEN	T1	T4	T5, T6
T2, T5, T6	Oğuzhan DEMİR	T2	T3	T5, T6
T3, T5, T6	Özgür BASKIN	T2	T3	T5, T6

Backlog Updates

As we remarked in Start-up document, we implemented the parser in this sprint. But, it happened to have some deficient parts. We were planning to implement all regex operands, however, since there were some unclear parts, we decided to implement core operands of regex and left the rest for Sprint 3. To be more clear, our implementation includes “., *, |” operands of regex and '&' as our default assumption. The other operands such as “+, ?, []” will be implemented in Sprint 3 after discussing with Sengör Hoca about our strategy.