**University of Central Punjab**

**Faculty of Information Technology**



**Compiler Construction**

Project (Phase-1) (Section All)

Spring (2020)

Submission Before: 6:00PM - 6-04-2020

(Late will be penalty of deduction of 2 absolute marks per day)

UCP-Compiler:

In lexical following token types required to be implemented.

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | Detail | Example |
| 1 | Identifiers | (\_|L)(L|\_|D|@)\*(D|\_) | \_r@te2, \_rate\_, r@t@e2 …etc |
| 2 | Numbers | [+-]?(D+)(\.D+)?, and exponent numbers. | 3.43433E+13, |
| 3 | Operators | <,>,!=, <>, :=, ==, \*, +, /,-, >>,<<, ++, +=, &&, ||, >=, =<, %, |  |
| 4 | Punctuations | [,{,(,),},] |  |
| 5 | Keyword | |  |  | | --- | --- | | **C-Compiler** | **UCP-Compiler** | | while | loop | | do-while | do-loop | | for | tour | | if | iff | | If-else | iif-else | | cin >> , cout << | input -> , output <- |   **Remaining:**  asm else new this auto enum operator throw bool explicit private true break export protected try case extern public typedef catch false char float typename class return union const friend short unsigned goto signed using continue sizeof virtual default inline static void delete int volatile do long struct double switch namespace template |  |

# Assignment Description:

For this assignment, you have to implement a **lexical analyzer**, also called a scanner. This assignment includes following parts:

|  |  |  |  |
| --- | --- | --- | --- |
|  | **PARTS** | **Output** | **Marks** |
| **1** | Generate FA for each regular expression | Document | 15 |
| **2** | Code - Table Driven Approach | Source Code Files | 85 |
|  | Total |  | 100 |

# Tools:

Language (For Development): C++

**Note: Student cannot use built-in data structure. Student can use his own data structure Hash Table, Linked List which he/she developed in data structure course. In this case student has to show me the code before using it.**

# Evaluating Criteria:

1. Source code should reflect the detail given in documents (other parts).
2. A text file with valid source code will be input of the scanner and Token file will be output of the scanner tool.
3. All points discussed in the class related to scanner implementation should reflect in the assignment.
4. Application should not do which is not supposed to do.

**SCANNER**

Token.txt

Sourcecode.txt

Error.txt

1. In case of invalid source code tool must generate error list.

**GOOD LUCK**