

# SCHOOL OF ADVANCED TECHNOLOGY

### ICT - Applications & Programming

### Computer Engineering Technology – Computing Science



A21

Model Definitions (RE/Automaton)

Lab Professor / Lab Session:

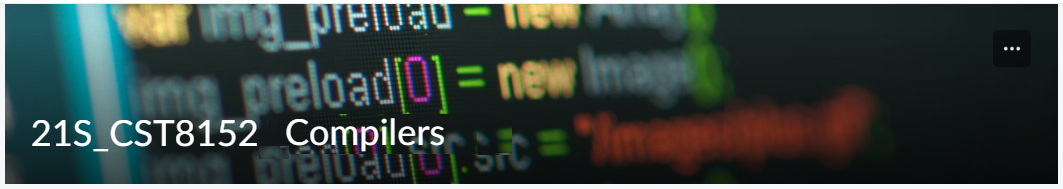
[Paulo Sousa / 311]

Team:

[Yen Huynh] - Id: [041 068 712] / [Mohammed Muhsin Shaik] - Id: [041 078 827]

Language Name [CoreX]

***This template is suggested (not mandatory) to answer A11 Specification.***



# MODEL TASK[[1]](#footnote-2) FOR ASSIGNMENT 2.1 – CoreX

* ***Note:*** *This task is part of the Assignment 2 from Compilers Course. Any problem, contact your lab professor.*

### Task 1: Language Symbols (2.0 marks)

**List examples:** Define main symbols to be used:

* Special symbols (ex: “{“, “(“, “<=”, etc.)
* Comments
* Variables
* Functions
* Literals (numbers, strings)
* Keywords
* Datatypes

**Answer:**

**Special symbols:** {, }, <, >, -, (, ), =, etc.

**Comments:**

**Single line comment (SLC):** Using pound or #

**Multi-line comment (MLC):** Select the multi-line and use Ctr + Shift + c

**Variables: (Identifiers)**

|  |  |  |
| --- | --- | --- |
| **Equal to operator** | **Leftward Operator** | **Rightward Operator** |
| Moha = “AClive” | Yen <- “AClive” | “AClive” -> Muhsin |

**Functions:**

|  |  |
| --- | --- |
| **Assign by using an arrow** | **Assign by using the equals sign** |
| sumXY <- function() {} | sumXY = function() {} |

**Literals (numbers, strings):**

|  |  |
| --- | --- |
| **Numbers (integers, floats)** | **String** |
| x <- 12L  y <- 1.2L | x <- “Hello World”   * **Using quotations as literals** |

**Keywords:**

In CoreX, we have: if, else, repeat, while, function, for, in, next, break, TRUE, FALSE, NULL, Inf, NaN, NA, NA\_integer\_, NA\_real\_, NA\_complex\_, NA\_character\_and ... (ellipsis)

**Ex:** x = 5

while(x != 7) {

if (x == 7) {break}

else{x++}

}

**Datatypes:** In CoreX, we have integer, string/ character, and float.

**Int:** x <- 5

**Char:** x <- “Hello World!”

**Float:** x <- 0.123

### Task 2: RE – Regular Expression (3.0 mark)

**Lexeme Classes:** Define the classes to be used in your regular expression:

**Answer:**

**L [0]:** Letters [A-Z | a-z]

**D [1]:** Digits [0-9]

**U [2]:** Underscore ( \_ )

**S [3]:** Slash: ( / )

**H [4]:** Hashtag ( # )

**N [5]:** New line ( \n )

**A [6]:** Asterisk ( \* )

**P [7]:** Period ( . )

**Q [8]:** Single Quotes ( ' )

**R [9]:** Double Quotes ( " )

**O [10]:** Other Characters

**Notes:**

+ Cannot declare twice for the class or exceed 1 alphabet -> for ex: QQ, QA. We can do Q2, Q1, etc.

+ Last class should be O (other): ^[L.D.CLASS that declare]

+ ^ means NOT

+ Define class should have a sequence for each: L [0], D[1], etc.

Define the RE to be used for: variables, literals and keywords:

**Answer:**

**Comments:**

**SLC:** H(^N)\*N

**MLC:** HH (^HH)\* HH

**Variables:**

**VID (Variables Identifier):** L[L | D | U]\*^[L | D | U]

**Methods:**

**MID (Methods Identifier):** L[L | D | U]\* ^[L | D | U]

**Literals:**

**Numbers:** D+

**SL (String literals): Single Quotes:** Q[^Q]\*Q & **Double Quotes:** R[^R]\*R

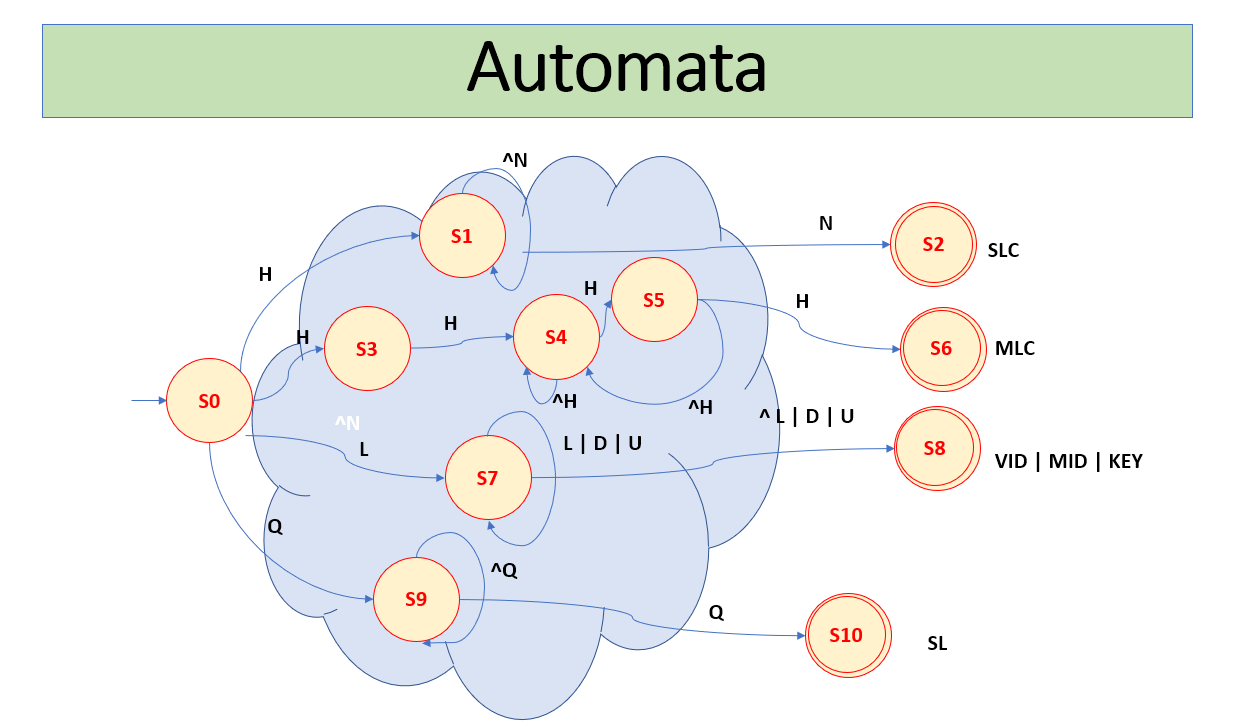
**Keywords:** L+= {if, else, etc}

|  |
| --- |
| **Example: (Outside topic)**  **\* Comments:**  Ex: SLC: H(^E)\*E means traversing HEHEHEHEHE until we enter “E”  MLC (multi-line comment): HH[HH^]HH  **\* Variables:**  VID (variable identifier) : L[L | D | U]\*    **\* Methods:**  **\* Literals:**  - Numbers: D+  - Strings:  **\* Keywords:**  L += {if/ else, while/ repeat/ for, function, in, next, break. TRUE/ FALSE, NULL, Inf/ NaN, etc} |

### Task 3: TD – Transition Diagram (3.0 marks)

**Activity:** Starting from the previous lexeme classes and obeying the RE, it is possible to define the automata for your language:

**Answer (draw / copy image of the automata):**



### Task 4: TT – Transition Table (2.0 marks)

**Activity:** Now, it is possible to define the TT for these lexemes:

**Answer (Transition Table):**

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **State \ Classes** | **0** | **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | **9** | **10** |
|  | **L(A-Z)** | **D(0-9)** | **U( \_ )** | **S( / )** | **H( # )** | **N( \n )** | **A ( \* )** | **P ( . )** | **Q ( ‘ )** | **R( “ )** | **O** |
| **S0** | **7** | **ES** | **ES** | **ES** | **1, 3** | **ES** | **ES** | **ES** | **9** | **ES** | **ES** |
| **S1** | **7** | **7** | **7** | **ES** | **4** | **1** | **ES** | **ES** | **9** | **ES** | **3** |
| **S2** | **FS** | **FS** | **FS** | **ES** | **5** | **FS** | **ES** | **ES** | **FS** | **ES** | **FS** |
| **S3** | **FS** | **FS** | **FS** | **ES** | **FS** | **FS** | **ES** | **ES** | **FS** | **ES** | **FS** |

*Update: Jun 1st 2024.*

1. Adapted from resources developed by Prof. Svillen Ranev (Algonquin College, 2019) [↑](#footnote-ref-2)