

A dark blue vertical bar runs down the left side of the slide. A blue arrow points to the right from the bar, containing the text 'CSC 3315'.

CSC 3315

Team report: lexer, part 3

Hajar El Boutahiri <80389>

Youssef Yousfi <85369>

Hiba Aqqaoui <94519>

Several thin, curved lines in shades of blue and grey originate from the bottom left corner and sweep upwards and to the right, creating a decorative, organic shape.

I. Alternative choice

The choice of implementing a lexer from scratch using Python and not using lexer constructor tools came from the fact that each teammate wanted to know how a lexical analyzer works in depth. Manually coding the lexer has also been quite convenient when it came to debugging as we could keep track of different iterations throughout the source code file. It is also beneficial for us as we are also going to implement the parser afterwards.

II. Lexer's design and functions

We start by reading from the source code file, we read word by word, and character by character. Then, we use `process_lexeme()`, the main lexing function that lexes or in other words, puts together operators, punctuation, reserved words. The lexemes are stored in a list `string_lexeme`. Tokens `getToken(lexeme)` are defined by a function that returns a list for each lexeme (containing a unique numerical identifier and token). For the symbol table, we used a two-dimensional list in which reserved words of our language are pre-stored and to which we will be adding variables. The list is two dimensional in order to hold the 'symbols' and their type (reserved word or variable name).

For the literal table, the literals are matched with their name and type. The literal and symbol table both can be output in the terminal.

III. Lexer User Manual :

The py file, the source code in flower language (in a text-file named 'file.txt') and the output file('file2.txt') should have the same file directories. The input is written in the flower language. The output consists of the lexemes, their respective tokens, and the lines where they occur in the input.

IV. Comments :

We kept the character literals because we use them in our language. Also, we wrote productions for arithmetic expressions, just not the way we have them in the slides.

V. Lexer Performance :

These code examples are mainly to show that each category works, so some things, such as the statements following an if statement, have been skipped to generate a smaller output.

Punctuation	Code : <pre>int main(void){ int a; int b ; int c ; }</pre>
	Output:

	<pre>1 Code Line Lexemes: Tokens: 2 1 int INT_RES 3 1 main MAIN_RES 4 1 (L_PAREN 5 1 void VOID_RES 6 1) R_PAREN 7 1 { LCBRK 8 2 int INT_RES 9 2 a ID 10 2 ; S_COLON 11 3 int INT_RES 12 3 b ID 13 3 ; S_COLON 14 4 int INT_RES 15 4 c ID 16 4 ; S_COLON 17 5 } RCBRK 18</pre>
Unknown	Code :
Lexemes	<pre>int main(void)^ int a\$ if (a>b# if (a<b& if (a>=b : if (a<=b! if (a!=b if (a==b~ a=b 6Rt }</pre>
	Output :

Error: Unkown Lexeme: ^ Line: 1

Error: Unkown Lexeme: a\$ Line: 2

Error: Unkown Lexeme: b# Line: 3

Error: Unkown Lexeme: b& Line: 4

Error: Unkown Lexeme: : Line: 5

Error: Unkown Lexeme: ! Line: 6

Error: Unkown Lexeme: b| Line: 7

Error: Unkown Lexeme: b~ Line: 8

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL Python + ~ [ ] [X]
Error: Unkown Lexeme: 6Rt Line: 9

**** If you want to visualize the Symbol Table and Literal Table Press 1 else Press 0 ****
1
    Symbol Table:

Lexeme:                                Type of Lexeme:
main                                    RES_WORD
void                                    RES_WORD
back                                    RES_WORD
SET                                     RES_WORD
IF                                      RES_WORD
ELSE                                    RES_WORD
WHILE                                   RES_WORD
BEGIN                                  RES_WORD
END                                    RES_WORD
int                                     RES_WORD
bool                                    RES_WORD
char                                    RES_WORD
Goup                                    RES_WORD
GODOWN                                 RES_WORD
GOLEFT                                 RES_WORD
GORIGHT                                RES_WORD
```

Symbol Table and Literal Table	<div> Code : <pre> int main(void){ int a; int c; int d; c= 4; d=5; a=2; } </pre> </div> <div> Output: <pre> ***** If you want to visualize the Symbol Table and Literal Table Press 1 else Press 0 ***** 1 Symbol Table: Lexeme: Type of Lexeme: main RES_WORD void RES_WORD back RES_WORD SET RES_WORD IF RES_WORD ELSE RES_WORD WHILE RES_WORD BEGIN RES_WORD END RES_WORD int RES_WORD bool RES_WORD char RES_WORD GOUN RES_WORD GODOWN RES_WORD GOLEFT RES_WORD GORIGHT RES_WORD PICKFLOWER RES_WORD a ID c ID d ID Literal Table: </pre> </div>
---	---

	<div>Literal Table:</div> <table><tr><th>Type:</th><th>Name:</th></tr><tr><td>Int Literal</td><td>4</td></tr><tr><td>Int Literal</td><td>5</td></tr><tr><td>Int Literal</td><td>2</td></tr></table>	Type:	Name:	Int Literal	4	Int Literal	5	Int Literal	2																																																																																																								
Type:	Name:																																																																																																																
Int Literal	4																																																																																																																
Int Literal	5																																																																																																																
Int Literal	2																																																																																																																
Operators	<div>Code :</div> <pre>int main(void){ int a,b; if (a>b) if (a<b) if (a>=b) if (a<=b) if (a!=b) if (a==b) a=b; }</pre> <div>Output:</div> <table><tr><th>1</th><th>Code Line</th><th>Lexemes:</th><th>Tokens:</th></tr><tr><td>2</td><td>1</td><td>int</td><td>INT_RES</td></tr><tr><td>3</td><td>1</td><td>main</td><td>MAIN_RES</td></tr><tr><td>4</td><td>1</td><td>(</td><td>L_PAREN</td></tr><tr><td>5</td><td>1</td><td>void</td><td>VOID_RES</td></tr><tr><td>6</td><td>1</td><td>)</td><td>R_PAREN</td></tr><tr><td>7</td><td>1</td><td>{</td><td>LCBRK</td></tr><tr><td>8</td><td>2</td><td>int</td><td>INT_RES</td></tr><tr><td>9</td><td>2</td><td>a</td><td>ID</td></tr><tr><td>10</td><td>2</td><td>,</td><td>COLON</td></tr><tr><td>11</td><td>2</td><td>b</td><td>ID</td></tr><tr><td>12</td><td>2</td><td>;</td><td>S_COLON</td></tr><tr><td>13</td><td>3</td><td>if</td><td>IF_RES</td></tr><tr><td>14</td><td>3</td><td>(</td><td>L_PAREN</td></tr><tr><td>15</td><td>3</td><td>a</td><td>ID</td></tr><tr><td>16</td><td>3</td><td>></td><td>BT_OP</td></tr><tr><td>17</td><td>3</td><td>b</td><td>ID</td></tr><tr><td>18</td><td>3</td><td>)</td><td>R_PAREN</td></tr><tr><td>19</td><td>4</td><td>if</td><td>IF_RES</td></tr><tr><td>20</td><td>4</td><td>(</td><td>L_PAREN</td></tr><tr><td>21</td><td>4</td><td>a</td><td>ID</td></tr><tr><td>22</td><td>4</td><td><</td><td>LT_OP</td></tr><tr><td>23</td><td>4</td><td>b</td><td>ID</td></tr><tr><td>24</td><td>4</td><td>)</td><td>R_PAREN</td></tr><tr><td>25</td><td>5</td><td>if</td><td>IF_RES</td></tr><tr><td>26</td><td>5</td><td>(</td><td>L_PAREN</td></tr><tr><td>27</td><td>5</td><td>a</td><td>ID</td></tr><tr><td>28</td><td>5</td><td>>=</td><td>RTOE_OP</td></tr></table>	1	Code Line	Lexemes:	Tokens:	2	1	int	INT_RES	3	1	main	MAIN_RES	4	1	(L_PAREN	5	1	void	VOID_RES	6	1)	R_PAREN	7	1	{	LCBRK	8	2	int	INT_RES	9	2	a	ID	10	2	,	COLON	11	2	b	ID	12	2	;	S_COLON	13	3	if	IF_RES	14	3	(L_PAREN	15	3	a	ID	16	3	>	BT_OP	17	3	b	ID	18	3)	R_PAREN	19	4	if	IF_RES	20	4	(L_PAREN	21	4	a	ID	22	4	<	LT_OP	23	4	b	ID	24	4)	R_PAREN	25	5	if	IF_RES	26	5	(L_PAREN	27	5	a	ID	28	5	>=	RTOE_OP
1	Code Line	Lexemes:	Tokens:																																																																																																														
2	1	int	INT_RES																																																																																																														
3	1	main	MAIN_RES																																																																																																														
4	1	(L_PAREN																																																																																																														
5	1	void	VOID_RES																																																																																																														
6	1)	R_PAREN																																																																																																														
7	1	{	LCBRK																																																																																																														
8	2	int	INT_RES																																																																																																														
9	2	a	ID																																																																																																														
10	2	,	COLON																																																																																																														
11	2	b	ID																																																																																																														
12	2	;	S_COLON																																																																																																														
13	3	if	IF_RES																																																																																																														
14	3	(L_PAREN																																																																																																														
15	3	a	ID																																																																																																														
16	3	>	BT_OP																																																																																																														
17	3	b	ID																																																																																																														
18	3)	R_PAREN																																																																																																														
19	4	if	IF_RES																																																																																																														
20	4	(L_PAREN																																																																																																														
21	4	a	ID																																																																																																														
22	4	<	LT_OP																																																																																																														
23	4	b	ID																																																																																																														
24	4)	R_PAREN																																																																																																														
25	5	if	IF_RES																																																																																																														
26	5	(L_PAREN																																																																																																														
27	5	a	ID																																																																																																														
28	5	>=	RTOE_OP																																																																																																														

	<pre>28 5 >= BTOE_OP 29 5 b ID 30 5) R_PAREN 31 6 if IF_RES 32 6 (L_PAREN 33 6 a ID 34 6 <= LTOE_OP 35 6 b ID 36 6) R_PAREN 37 7 if IF_RES 38 7 (L_PAREN 39 7 a ID 40 7 != NEQL_OP 41 7 b ID 42 7) R_PAREN 43 8 if IF_RES 44 8 (L_PAREN 45 8 a ID 46 8 == EQL_OP 47 8 b ID 48 8) R_PAREN 49 9 a ID 50 9 = ASSIGN_OP 51 9 b ID 52 9 ; S_COLON 53 11 } RCBRK 54</pre>	
<div>Reserved</div> <div>Words</div>	<div>Code:</div> <pre>int main(void){ int i,b,c; SET i = 0; if (b>c)[GOUP(); GODOWN();] else[GOLEFT(); GORIGHT();] WHILE i<4 BEGIN PICKFLOWER(); SET i = i + 1 END back 0; }</pre> <div>Output:</div>	

	1	Code Line	Lexemes:	Tokens:
	2	1	int	INT_RES
	3	1	main	MAIN_RES
	4	1	(L_PAREN
	5	1	void	VOID_RES
	6	1)	R_PAREN
	7	1	{	LCBRK
	8	2	int	INT_RES
	9	2	i	ID
	10	2	;	S_COLON
	11	3	int	INT_RES
	12	3	b	ID
	13	4	int	INT_RES
	14	4	c	ID
	15	4	;	S_COLON
	16	5	SET	SET_RES
	17	5	i	ID
	18	5	=	ASSIGN_OP
	19	5	0	INT_LIT
	20	5	;	S_COLON
	21	6	if	IF_RES
	22	6	(L_PAREN
	23	6	b	ID
	24	6	>	BT_OP
	25	6	c	ID
	26	6)	R_PAREN
	27	6	[LSQR
	28	7	GOLD	GOLL_RES
	28	7	GOUP	GOO_RES
	29	7	(L_PAREN
	30	7)	R_PAREN
	31	7	;	S_COLON
	32	8	GODOWN	GOD_RES
	33	8	(L_PAREN
	34	8)	R_PAREN
	35	8	;	S_COLON
	36	9]	RSQR
	37	10	else	ELSE_RES
	38	10	[LSQR
	39	11	GOLEFT	GOL_RES
	40	11	(L_PAREN
	41	11)	R_PAREN
	42	11	;	S_COLON
	43	12	GORIGHT	GOR_RES
	44	12	(L_PAREN
	45	12)	R_PAREN
	46	12	;	S_COLON
	47	13]	RSQR
	48	14	WHILE	WHILE_RES
	49	14	i	ID
	50	14	<	LT_OP
	51	14	4	INT_LIT
	52	15	BEGIN	BEGIN_RES
	53	16	PICKFLOWER	PICKF_RES
	54	16	(L_PAREN
	55	16)	R_PAREN
	Ln 27, Col 57 Spaces: 4 UTF-8 CRLF Plain Text Colorize: 0			

		46	12	;	S_COLON
		47	13]	RSQR
		48	14	WHILE	WHILE_RES
		49	14	i	ID
		50	14	<	LT_OP
		51	14	4	INT_LIT
		52	15	BEGIN	BEGIN_RES
		53	16	PICKFLOWER	PICKF_RES
		54	16	(L_PAREN
		55	16)	R_PAREN
		56	16	;	S_COLON
		57	17	SET	SET_RES
		58	17	i	ID
		59	17	=	ASSIGN_OP
		60	17	i	ID
		61	17	+	ADD_OP
		62	17	1	INT_LIT
		63	18	END	END_RES
		64	19	back	BACK_RES
		65	19	0	INT_LIT
		66	19	;	S_COLON
		67	20	}	RCBRK
		68			

User-	Code:																																																																																				
Defined IDs	<pre>int main(void){ int a; bool b; char c; }</pre>																																																																																				
	Output: <table><tr><th>1</th><th>Code</th><th>Line</th><th>Lexemes:</th><th>Tokens:</th></tr><tr><td>2</td><td>1</td><td></td><td>int</td><td>INT_RES</td></tr><tr><td>3</td><td>1</td><td></td><td>main</td><td>MAIN_RES</td></tr><tr><td>4</td><td>1</td><td></td><td>(</td><td>L_PAREN</td></tr><tr><td>5</td><td>1</td><td></td><td>void</td><td>VOID_RES</td></tr><tr><td>6</td><td>1</td><td></td><td>)</td><td>R_PAREN</td></tr><tr><td>7</td><td>1</td><td></td><td>{</td><td>LCBRK</td></tr><tr><td>8</td><td>2</td><td></td><td>int</td><td>INT_RES</td></tr><tr><td>9</td><td>2</td><td></td><td>a</td><td>ID</td></tr><tr><td>10</td><td>2</td><td></td><td>;</td><td>S_COLON</td></tr><tr><td>11</td><td>3</td><td></td><td>bool</td><td>BOOL_RES</td></tr><tr><td>12</td><td>3</td><td></td><td>b</td><td>ID</td></tr><tr><td>13</td><td>3</td><td></td><td>;</td><td>S_COLON</td></tr><tr><td>14</td><td>4</td><td></td><td>char</td><td>CHAR_RES</td></tr><tr><td>15</td><td>4</td><td></td><td>c</td><td>ID</td></tr><tr><td>16</td><td>4</td><td></td><td>;</td><td>S_COLON</td></tr><tr><td>17</td><td>5</td><td></td><td>}</td><td>RCBRK</td></tr></table>	1	Code	Line	Lexemes:	Tokens:	2	1		int	INT_RES	3	1		main	MAIN_RES	4	1		(L_PAREN	5	1		void	VOID_RES	6	1)	R_PAREN	7	1		{	LCBRK	8	2		int	INT_RES	9	2		a	ID	10	2		;	S_COLON	11	3		bool	BOOL_RES	12	3		b	ID	13	3		;	S_COLON	14	4		char	CHAR_RES	15	4		c	ID	16	4		;	S_COLON	17	5		}
1	Code	Line	Lexemes:	Tokens:																																																																																	
2	1		int	INT_RES																																																																																	
3	1		main	MAIN_RES																																																																																	
4	1		(L_PAREN																																																																																	
5	1		void	VOID_RES																																																																																	
6	1)	R_PAREN																																																																																	
7	1		{	LCBRK																																																																																	
8	2		int	INT_RES																																																																																	
9	2		a	ID																																																																																	
10	2		;	S_COLON																																																																																	
11	3		bool	BOOL_RES																																																																																	
12	3		b	ID																																																																																	
13	3		;	S_COLON																																																																																	
14	4		char	CHAR_RES																																																																																	
15	4		c	ID																																																																																	
16	4		;	S_COLON																																																																																	
17	5		}	RCBRK																																																																																	

Numeric Literals and String Literals	Code: int main(void){ int a ; ctchar b ; SET a = 214; SET b = "str" }																																																																																																			
	Output: <table><tr><th>1</th><th>Code Line</th><th>Lexemes:</th><th>Tokens:</th></tr><tr><td>2</td><td>1</td><td>int</td><td>INT_RES</td></tr><tr><td>3</td><td>1</td><td>main</td><td>MAIN_RES</td></tr><tr><td>4</td><td>1</td><td>(</td><td>L_PAREN</td></tr><tr><td>5</td><td>1</td><td>void</td><td>VOID_RES</td></tr><tr><td>6</td><td>1</td><td>)</td><td>R_PAREN</td></tr><tr><td>7</td><td>1</td><td>{</td><td>LCBRK</td></tr><tr><td>8</td><td>2</td><td>int</td><td>INT_RES</td></tr><tr><td>9</td><td>2</td><td>a</td><td>ID</td></tr><tr><td>10</td><td>2</td><td>;</td><td>S_COLON</td></tr><tr><td>11</td><td>3</td><td>ctchar</td><td>CTCHAR_RES</td></tr><tr><td>12</td><td>3</td><td>b</td><td>ID</td></tr><tr><td>13</td><td>3</td><td>;</td><td>S_COLON</td></tr><tr><td>14</td><td>4</td><td>SET</td><td>SET_RES</td></tr><tr><td>15</td><td>4</td><td>a</td><td>ID</td></tr><tr><td>16</td><td>4</td><td>=</td><td>ASSIGN_OP</td></tr><tr><td>17</td><td>4</td><td>214</td><td>INT_LIT</td></tr><tr><td>18</td><td>4</td><td>;</td><td>S_COLON</td></tr><tr><td>19</td><td>5</td><td>SET</td><td>SET_RES</td></tr><tr><td>20</td><td>5</td><td>b</td><td>ID</td></tr><tr><td>21</td><td>5</td><td>=</td><td>ASSIGN_OP</td></tr><tr><td>22</td><td>5</td><td>"</td><td>Q_MARKS</td></tr><tr><td>23</td><td>5</td><td>str</td><td>STR_LIT</td></tr><tr><td>24</td><td>5</td><td>"</td><td>Q_MARKS</td></tr><tr><td>25</td><td>6</td><td>}</td><td>RCBRK</td></tr></table>	1	Code Line	Lexemes:	Tokens:	2	1	int	INT_RES	3	1	main	MAIN_RES	4	1	(L_PAREN	5	1	void	VOID_RES	6	1)	R_PAREN	7	1	{	LCBRK	8	2	int	INT_RES	9	2	a	ID	10	2	;	S_COLON	11	3	ctchar	CTCHAR_RES	12	3	b	ID	13	3	;	S_COLON	14	4	SET	SET_RES	15	4	a	ID	16	4	=	ASSIGN_OP	17	4	214	INT_LIT	18	4	;	S_COLON	19	5	SET	SET_RES	20	5	b	ID	21	5	=	ASSIGN_OP	22	5	"	Q_MARKS	23	5	str	STR_LIT	24	5	"	Q_MARKS	25	6	}
1	Code Line	Lexemes:	Tokens:																																																																																																	
2	1	int	INT_RES																																																																																																	
3	1	main	MAIN_RES																																																																																																	
4	1	(L_PAREN																																																																																																	
5	1	void	VOID_RES																																																																																																	
6	1)	R_PAREN																																																																																																	
7	1	{	LCBRK																																																																																																	
8	2	int	INT_RES																																																																																																	
9	2	a	ID																																																																																																	
10	2	;	S_COLON																																																																																																	
11	3	ctchar	CTCHAR_RES																																																																																																	
12	3	b	ID																																																																																																	
13	3	;	S_COLON																																																																																																	
14	4	SET	SET_RES																																																																																																	
15	4	a	ID																																																																																																	
16	4	=	ASSIGN_OP																																																																																																	
17	4	214	INT_LIT																																																																																																	
18	4	;	S_COLON																																																																																																	
19	5	SET	SET_RES																																																																																																	
20	5	b	ID																																																																																																	
21	5	=	ASSIGN_OP																																																																																																	
22	5	"	Q_MARKS																																																																																																	
23	5	str	STR_LIT																																																																																																	
24	5	"	Q_MARKS																																																																																																	
25	6	}	RCBRK																																																																																																	
Whitespace	Code: int main(void){ int a; bool b; b = True; a = 14; }																																																																																																			
	Output:																																																																																																			

= lexemes			
1	Code	Line	Lexemes: Tokens:
2	1	int	INT_RES
3	1	main	MAIN_RES
4	1	(L_PAREN
5	1	void	VOID_RES
6	1)	R_PAREN
7	1	{	LCBRK
8	2	int	INT_RES
9	2	a	ID
10	2	;	S_COLON
11	3	bool	BOOL_RES
12	3	b	ID
13	3	;	S_COLON
14	4	b	ID
15	4	=	ASSIGN_OP
16	4	True	BOOL_LIT
17	4	;	S_COLON
18	5	a	ID
19	5	=	ASSIGN_OP
20	5	14	INT_LIT
21	5	;	S_COLON
22	6	}	RCBRK
22			