

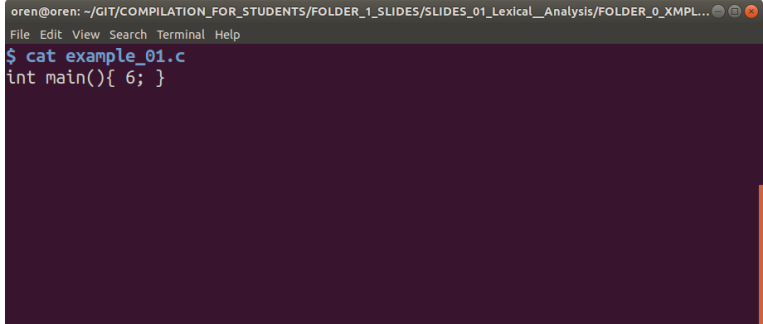
Lexical Analysis

Verifying that every word is legal

November 3, 2019

Warm up examples

► Numbers

A terminal window with a dark background and light text. The title bar at the top reads "oren@oren: ~/GIT/COMPILATION_FOR_STUDENTS/FOLDER_1_SLIDES/SLIDES_01_Lexical__Analysis/FOLDER_0_XMPL...". Below the title bar is a menu bar with "File", "Edit", "View", "Search", "Terminal", and "Help". The main area of the terminal shows a command prompt "\$" followed by the command "cat example_01.c". The output of the command is "int main(){ 6; }".

```
oren@oren: ~/GIT/COMPILATION_FOR_STUDENTS/FOLDER_1_SLIDES/SLIDES_01_Lexical__Analysis/FOLDER_0_XMPL...
File Edit View Search Terminal Help
$ cat example_01.c
int main(){ 6; }
```

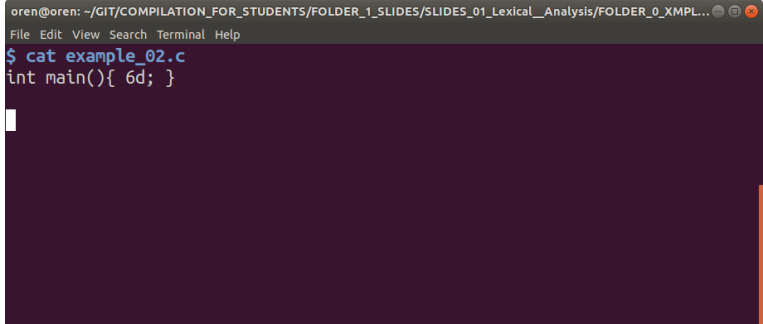
Warm up examples

► Numbers

```
oren@oren: ~/GIT/COMPILATION_FOR_STUDENTS/FOLDER_1_SLIDES/SLIDES_01_Lexical_Analysis/FOLDER_0_XMPL...  
File Edit View Search Terminal Help  
$ cat example_01.c  
int main(){ 6; }  
  
$ gcc example_01.c  
/home/oren/GIT/COMPILATION_FOR_STUDENTS/FOLDER_1_SLIDES/SLIDES_01_Lexical_Analysis/FOLDER_0_XMPLFILES/example_01.c: In function 'main':  
/home/oren/GIT/COMPILATION_FOR_STUDENTS/FOLDER_1_SLIDES/SLIDES_01_Lexical_Analysis/FOLDER_0_XMPLFILES/example_01.c:1:13: warning: statement with no effect [-Wunused-value]  
int main(){ 6; }  
            ^
```

Warm up examples

► Numbers

A terminal window with a dark purple background and a light gray title bar. The title bar contains the text "oren@oren: ~/GIT/COMPILATION_FOR_STUDENTS/FOLDER_1_SLIDES/SLIDES_01_Lexical__Analysis/FOLDER_0_XMPL..." and standard window control buttons. The terminal has a menu bar with "File", "Edit", "View", "Search", "Terminal", and "Help". The command prompt shows "\$ cat example_02.c" and the output is "int main(){ 6d; }". A white cursor is positioned at the end of the line.

```
oren@oren: ~/GIT/COMPILATION_FOR_STUDENTS/FOLDER_1_SLIDES/SLIDES_01_Lexical__Analysis/FOLDER_0_XMPL...  
File Edit View Search Terminal Help  
$ cat example_02.c  
int main(){ 6d; }
```

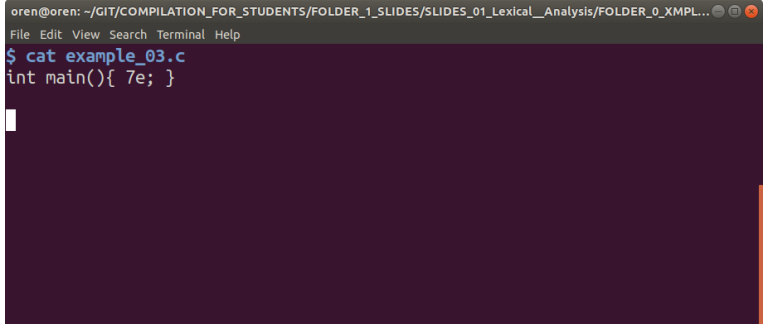
Warm up examples

► Numbers

```
oren@oren: ~/GIT/COMPILATION_FOR_STUDENTS/FOLDER_1_SLIDES/SLIDES_01_Lexical_Analysis/FOLDER_0_XMPL...  
File Edit View Search Terminal Help  
$ cat example_02.c  
int main(){ 6d; }  
  
$ gcc example_02.c  
/home/oren/GIT/COMPILATION_FOR_STUDENTS/FOLDER_1_SLIDES/SLIDES_01_Lexical_Analysis/FOLDER_0_XMPLFILES/example_02.c: In function 'main':  
/home/oren/GIT/COMPILATION_FOR_STUDENTS/FOLDER_1_SLIDES/SLIDES_01_Lexical_Analysis/FOLDER_0_XMPLFILES/example_02.c:1:13: error: invalid suffix "d" on  
integer constant  
int main(){ 6d; }  
            ^~
```

Warm up examples

► Numbers



```
oren@oren: ~/GIT/COMPILATION_FOR_STUDENTS/FOLDER_1_SLIDES/SLIDES_01_Lexical__Analysis/FOLDER_0_XMPL...  
File Edit View Search Terminal Help  
$ cat example_03.c  
int main(){ 7e; }
```

A terminal window with a dark purple background. The title bar shows the path: `oren@oren: ~/GIT/COMPILATION_FOR_STUDENTS/FOLDER_1_SLIDES/SLIDES_01_Lexical__Analysis/FOLDER_0_XMPL...`. The menu bar includes `File Edit View Search Terminal Help`. The command prompt shows `$ cat example_03.c` and the file content is `int main(){ 7e; }`. A white cursor is positioned at the end of the line.

Warm up examples

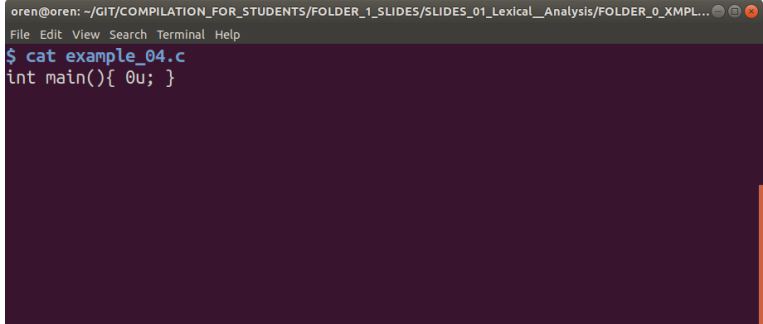
► Numbers

```
oren@oren: ~/GIT/COMPILATION_FOR_STUDENTS/FOLDER_1_SLIDES/SLIDES_01_Lexical_Analysis/FOLDER_0_XMPL...
File Edit View Search Terminal Help
$ cat example_03.c
int main(){ 7e; }

$ gcc example_03.c
/home/oren/GIT/COMPILATION_FOR_STUDENTS/FOLDER_1_SLIDES/SLIDES_01_Lexical_Analysis/FOLDER_0_XMPLFILES/example_03.c: In function 'main':
/home/oren/GIT/COMPILATION_FOR_STUDENTS/FOLDER_1_SLIDES/SLIDES_01_Lexical_Analysis/FOLDER_0_XMPLFILES/example_03.c:1:13: error: exponent has no digit
s
int main(){ 7e; }
            ^~
```

Warm up examples

► Numbers

A terminal window with a dark background and light text. The title bar at the top reads 'oren@oren: ~/GIT/COMPILATION_FOR_STUDENTS/FOLDER_1_SLIDES/SLIDES_01_Lexical__Analysis/FOLDER_0_XMPL...'. Below the title bar is a menu bar with 'File', 'Edit', 'View', 'Search', 'Terminal', and 'Help'. The main area of the terminal shows a command prompt '\$' followed by 'cat example_04.c'. Below this, the first line of a C program is visible: 'int main(){ 0u; }'.

```
oren@oren: ~/GIT/COMPILATION_FOR_STUDENTS/FOLDER_1_SLIDES/SLIDES_01_Lexical__Analysis/FOLDER_0_XMPL...  
File Edit View Search Terminal Help  
$ cat example_04.c  
int main(){ 0u; }
```

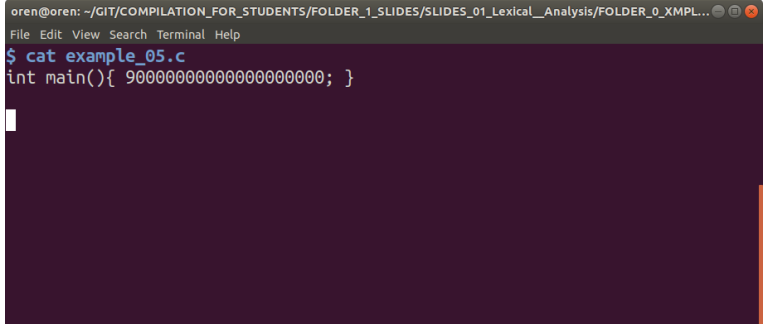

Warm up examples

► Numbers

```
oren@oren: ~/GIT/COMPILATION_FOR_STUDENTS/FOLDER_1_SLIDES/SLIDES_01_Lexical_Analysis/FOLDER_0_XMPL...  
File Edit View Search Terminal Help  
$ cat example_04.c  
int main(){ 0u; }  
  
$ gcc example_04.c  
/home/oren/GIT/COMPILATION_FOR_STUDENTS/FOLDER_1_SLIDES/SLIDES_01_Lexical_Analysis/FOLDER_0_XMPLFILES/example_04.c: In function 'main':  
/home/oren/GIT/COMPILATION_FOR_STUDENTS/FOLDER_1_SLIDES/SLIDES_01_Lexical_Analysis/FOLDER_0_XMPLFILES/example_04.c:1:13: warning: statement with no effect [-Wunused-value]  
int main(){ 0u; }  
             ^~
```

Warm up examples

► Numbers



```
oren@oren: ~/GIT/COMPILATION_FOR_STUDENTS/FOLDER_1_SLIDES/SLIDES_01_Lexical__Analysis/FOLDER_0_XMPL...  
File Edit View Search Terminal Help  
$ cat example_05.c  
int main(){ 9000000000000000000; }
```

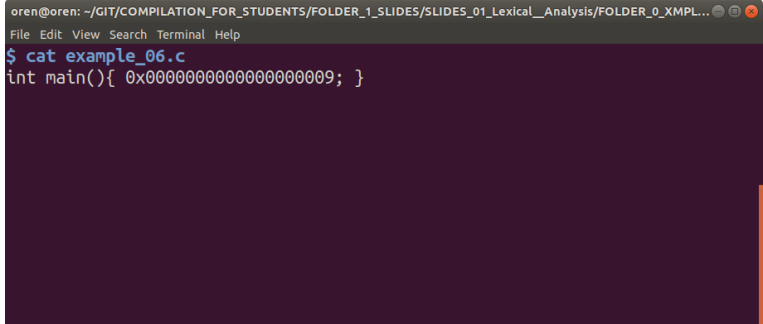
Warm up examples

- ▶ Numbers

```
oren@oren: ~/GIT/COMPILATION_FOR_STUDENTS/FOLDER_1_SLIDES/SLIDES_01_Lexical__Analysis/FOLDER_0_XMPL...  
File Edit View Search Terminal Help  
  
$ gcc example_05.c  
/home/oren/GIT/COMPILATION_FOR_STUDENTS/FOLDER_1_SLIDES/SLIDES_01_Lexical__Analysis/FOLDER_0_XMPLFILES/example_05.c: In function 'main':  
/home/oren/GIT/COMPILATION_FOR_STUDENTS/FOLDER_1_SLIDES/SLIDES_01_Lexical__Analysis/FOLDER_0_XMPLFILES/example_05.c:1:13: warning: integer constant is too large for its type  
int main(){ 9000000000000000000000000000; }  
            ^~  
  
/home/oren/GIT/COMPILATION_FOR_STUDENTS/FOLDER_1_SLIDES/SLIDES_01_Lexical__Analysis/FOLDER_0_XMPLFILES/example_05.c:1:13: warning: statement with no effect [-Wunused-value]
```

Warm up examples

► Numbers

A terminal window with a dark background and light text. The title bar at the top reads 'oren@oren: ~/GIT/COMPILATION_FOR_STUDENTS/FOLDER_1_SLIDES/SLIDES_01_Lexical__Analysis/FOLDER_0_XMPL...'. Below the title bar is a menu bar with 'File', 'Edit', 'View', 'Search', 'Terminal', and 'Help'. The main content area shows a command prompt '\$' followed by 'cat example_06.c'. The output of the command is a single line of C code: 'int main(){ 0x000000000000000009; }'.

```
oren@oren: ~/GIT/COMPILATION_FOR_STUDENTS/FOLDER_1_SLIDES/SLIDES_01_Lexical__Analysis/FOLDER_0_XMPL...
File Edit View Search Terminal Help
$ cat example_06.c
int main(){ 0x000000000000000009; }
```

Warm up examples

► Numbers

```
oren@oren: ~/GIT/COMPILATION_FOR_STUDENTS/FOLDER_1_SLIDES/SLIDES_01_Lexical_Analysis/FOLDER_0_XMPL...
File Edit View Search Terminal Help
$ cat example_06.c
int main(){ 0x0000000000000000009; }

$ gcc example_06.c
/home/oren/GIT/COMPILATION_FOR_STUDENTS/FOLDER_1_SLIDES/SLIDES_01_Lexical_Analysis/FOLDER_0_XMPLFILES/example_06.c: In function 'main':
/home/oren/GIT/COMPILATION_FOR_STUDENTS/FOLDER_1_SLIDES/SLIDES_01_Lexical_Analysis/FOLDER_0_XMPLFILES/example_06.c:1:13: warning: statement with no effect [-Wunused-value]
int main(){ 0x0000000000000000009; }
            ^~~~~~
```

Lexical Analyzer

- ▶ The input program is actually a long sentence
- ▶ The lexical analyzer verifies each word is legal
- ▶ What are the words of programming languages?

| | |
|-------------------|--|
| Constants | 123, 19.7, -700, 13e+8, 0x80, ... |
| Identifiers | numPoints, doSomething, ... |
| Reserved Keywords | while, for, int, virtual, class, auto, ... |
| Parentheses | (,), [,], {, }, ... |
| Binary Operators | +, -, *, /, ... |
| Unary Operators | !, ., *, →, ... |
| Comments | /* bla */ , # bla , ; bla , ... |

- ▶ Once an illegal word is encountered, compilation stops

Lexical Analyzer

- ▶ Recall the Google Docs lexical analyzer example [here](#)
- ▶ Why not do the same thing? simply keep a dictionary
- ▶ How many legal words are there (in C for instance)?
- ▶ Is keeping a dictionary feasible?
- ▶ So how should we specify legal words?
- ▶ Specification must be clear for programmers to understand
- ▶ Specification must enable a fast validity test for each word
- ▶ The answer is regular expressions

Regular Expressions (regex)

- ▶ All PLs use regular expressions to specify legal words
 - ▶ Python lexical specification [here](#)
 - ▶ ANSI C lexical specification [here](#)
 - ▶ JAVA lexical specification [here](#)
 - ▶ Haskell lexical specification [here](#)
 - ▶ Scala lexical specification [here](#)
- ▶ Here is a reminder about regular expressions: [wikipedia](#)
- ▶ Indeed, it is rather intuitive for human eyes
- ▶ How can we test membership fast with regular expressions?
- ▶ The answer is regex \rightarrow NFA \rightarrow DFA
- ▶ And remember that DFAs are easily programmable

Regex \rightarrow NFA

- ▶ Here is a reminder about DFA: [wikipedia](#)
- ▶ Here is a reminder about NFA: [wikipedia](#)
 - ▶ Actually, we only need a DFA + ϵ transitions
- ▶ Regex to NFA is a simple recursive algorithm
 - ▶ Suppose M_i is an NFA representing regex r_i
 - ▶ Which NFA represents $r_1 r_2$?
 - ▶ Which NFA represents $r_1 | r_2$?
 - ▶ Which NFA represents r_1^* ?
- ▶ Regex to NFA base cases
 - ▶ If $r_1 = a$, what is M_1 ?
 - ▶ If r_1 is the empty set, what is M_1 ?
 - ▶ If r_1 is the empty string, what is M_1 ?

NFA \rightarrow DFA

- ▶ NFA to DFA is a fairly simple iterative algorithm
- ▶ Described formally [here](#)
- ▶ Regex to NFA base cases