

Slovenská technická univerzita v Bratislave
Fakulta informatiky a informačných technológií

Syntaktický analyzátor
Simple URL

Autori: Michal Zajic, Tímea Nagy

Predmet: Softvérové jazyky

Cvičenie: Streda 8:00

Cvičiaci: doc. Ing Michal Kompan, PhD.

Obsah

1.	Analýza a návrh.....	1
1.1.	Príklady viet z jazyka	1
1.1.1.	Korektné príklady.....	1
1.1.2.	Nekorektné príklady	1
1.2.	Prepis gramatiky z BNF do gramatických pravidiel s alternatívami.....	2
1.2.1.	Gramatika v BNF	2
1.2.2.	Pravidlá s alternatívami	2
1.3.	Transformácia gramatiky do LL(1).....	4
1.4.	Overenie LL(1).....	7
1.4.1.	First ₁	7
1.4.2.	Follow ₁	7
1.5.	Tabuľka prechodov.....	9
2.	Implementácia	11
2.1.	Používateľské rozhranie	11
2.2.	Testy	13
2.3.	Tabuľka a pravidlá.....	14
2.4.	Postup implementácie.....	14
2.5.	Zotavenie z chýb	15
2.6.	Návod na spustenie.....	15

1. Analýza a návrh

Podiel práce:

- Tímea Nagy – 50%
- Michal Zajic – 50%

1.1. Príklady viet z jazyka

Pre analyzátor sme vytvorili testy a v nich sme snažili pokryť všetky typy URL adries, ktoré patria do jazyka a aj niekoľko chybných. V tejto časti sme vybrali niektoré z nich.

1.1.1. Korektné príklady

Príklad 1: `http://hostname:port/path?search`

Napr. `http://Youtube.com.foo:48484/Watch/Video?123456+foo+bar`

Príklad 2: `ftp://user:password@hostname:port/path`

Napr. `ftp://foo:fooBAR123456@foo:123456/foo/bar/foo/`

Príklad 3: `telnet://user:password@hostname:port`

Napr. `telnet://foo:123456@Foo:123456`

Príklad 4: `mailto::xalphas@xalphas.xalphas`

Napr. `mailto::foo@foo.BAR.foo.123`

1.1.2. Nekorektné príklady

Príklad 1: nesprávne poradie search a path

Napr. `http://foo.com?foo/bar`

Príklad 2: nesprávna ftp adresa

Napr. fpt://a@b/

Príklad 3: nesprávny port (port môže byť len číselný)

Napr. telnet://foo@foo:foo

Príklad 4: chýbajúci xalphas

Napr. mailto::

1.2. Prepis gramatiky z BNF do gramatických pravidiel s alternatívami

1.2.1. Gramatika v BNF

```
url := httpaddress | ftpaddress | telnetaddress | mailtoaddress
httpaddress := „http://“ hostport [ / path ] [ ? search ]
ftpaddress := „ftp://“ login / path
telnetaddress := „telnet://“ login
mailtoaddress := „mailto:“ xalphas @ hostname
login := [ user [ : password ] @ ] hostport
hostport := hostname [ : port ]
hostname := xalphas { . xalphas }
port := digits
path := segment { / segment }
search := xalphas { + xalphas }
user := xalphas
password := xalphas
segment := { xalpha }
xalphas := xalpha { xalpha }
xalpha := alpha | digit
digits := digit { digit }
alpha := „A“ | .. | „Z“ | „a“ | .. | „z“
digit := „0“ | .. | „9“
```

1.2.2. Pravidlá s alternatívami

Zadefinovanie neterminálov:

A = url
B = httpaddress
C = ftpaddress
D = telnetaddress
E = mailtoaddress
F = login

G = hostport
 H = hostname
 I = port
 J = path
 K = search
 L = user
 M = password
 N = segment
 O = xalphas
 P = xalpha
 R = digits
 S = alpha
 T = digit

Prepis gramatiky v BNF pomocou neterminálov definovaných vyššie:

$A \rightarrow B \mid C \mid D \mid E$
 $B \rightarrow \text{„http://“ } G \mid \text{„http://“ } G / J \mid \text{„http://“ } G / J ? K \mid \text{„http://“ } G ? K$
 $C \rightarrow \text{„ftp://“ } F / J$
 $D \rightarrow \text{„telnet://“ } F$
 $E \rightarrow \text{„mailto::“ } O @ H$
 $F \rightarrow L : M @ G \mid L @ G \mid G$
 $G \rightarrow H \mid H : I$
 $H \rightarrow O H^`$
 $H^` \rightarrow . O H^` \mid \varepsilon$
 $I \rightarrow R$
 $J \rightarrow N J^`$
 $J^` \rightarrow / N J^` \mid \varepsilon$
 $K \rightarrow O K^`$
 $K^` \rightarrow + O K^` \mid \varepsilon$
 $L \rightarrow O$
 $M \rightarrow O$
 $N \rightarrow P N \mid \varepsilon$
 $O \rightarrow P O^`$
 $O^` \rightarrow P O^` \mid \varepsilon$
 $P \rightarrow S \mid T$
 $R \rightarrow T R^`$
 $R^` \rightarrow T R^` \mid \varepsilon$
 $S \rightarrow [A-Za-z]$
 $T \rightarrow [0-9]$

Červenou farbou sú vyznačené miesta, kde gramatika nesĺpňa podmienky LL(1). Pri vyznačených miestach nastáva First-First konflikt.

1.3. Transformácia gramatiky do LL(1)

Na odstránenie First-First konfliktov, ktoré sme opísali vyššie sme využili faktorizáciu:

$A \rightarrow B \mid C \mid D \mid E$
 $B \rightarrow \text{„http://“ } G B'$
 $B' \rightarrow / J B'' \mid ? K \mid \varepsilon$
 $B'' \rightarrow ? K \mid \varepsilon$
 $C \rightarrow \text{„ftp://“ } F / J$
 $D \rightarrow \text{„telnet://“ } F$
 $E \rightarrow \text{„mailto::“ } O @ H$
 $F \rightarrow F' G$
 $F' \rightarrow L F'' \mid \varepsilon$
 $F'' \rightarrow @ \mid : M @$
 $G \rightarrow H G'$
 $G' \rightarrow : I \mid \varepsilon$
 $H \rightarrow O H'$
 $H' \rightarrow . O H' \mid \varepsilon$
 $I \rightarrow R$
 $J \rightarrow N J'$
 $J' \rightarrow / N J' \mid \varepsilon$
 $K \rightarrow O K'$
 $K' \rightarrow + O K' \mid \varepsilon$
 $L \rightarrow O$
 $M \rightarrow O$
 $N \rightarrow P N \mid \varepsilon$
 $O \rightarrow P O'$
 $O' \rightarrow P O' \mid \varepsilon$
 $P \rightarrow S \mid T$
 $R \rightarrow T R'$
 $R' \rightarrow T R' \mid \varepsilon$
 $S \rightarrow [A-Za-z]$
 $T \rightarrow [0-9]$

Červenou farbou je vyznačené miesto, kde gramatika naďalej nesĺpňa podmienky LL(1). Pri vyznačenom mieste nastáva First-First konflikt kvôli číslam a veľkým a malým písmenám. Výpočet First pre vybrané neterminály pre znázornenie First-First konfliktu:

$\text{First}(F) = \text{First}(F') \setminus \varepsilon \cup \text{First}(G) = \{[0-9], [A-Za-z]\} \cup \{[0-9], [A-Za-z]\}$
 $\text{First}(F') = \text{First}(L) \cup \{\varepsilon\} = \{[0-9], [A-Za-z]\} \cup \{\varepsilon\} = \{[0-9], [A-Za-z], \varepsilon\}$
 $\text{First}(G) = \text{First}(H) = \{[0-9], [A-Za-z]\}$
 $\text{First}(L) = \text{First}(O) = \{[0-9], [A-Za-z]\}$
 $\text{First}(H) = \text{First}(O) = \{[0-9], [A-Za-z]\}$
 $\text{First}(O) = \text{First}(P) = \{[0-9], [A-Za-z]\}$
 $\text{First}(P) = \text{First}(S) \setminus \varepsilon \cup \text{First}(T) = \{[0-9], [A-Za-z]\}$
 $\text{First}(T) = \{[0-9]\}$
 $\text{First}(S) = \{[A-Za-z]\}$

Po viacerých pokusoch odstránenia First-First konfliktu, po lepšej analýze daného konfliktu a po konzultácii s cvičiacim sa prišlo na to, že hore uvedenú gramatiku nie je možné transformovať na typ LL(1) prostriedkami obsiahnutými v predmete.

Problém nastáva pri riadkoch:

```
login := [ user [ : password ] @ ] hostport
hostport := hostname [ : port ]
```

Ako príklad si zoberieme vstup, ktorý obsahuje „usera“ a vstup, ktorý neobsahuje „usera“, ale len „hostname“. „User“ aj „hostname“ sa môžu skladať z čísel a z malých a z veľkých písmen. Gramatika nešpecifikuje spôsob, ako zistiť, že aktuálne spracovávaný znak patrí pre „usera“ alebo pre „hostname“.

Na riešenie tohto problému sme zaviedli pravidlo, že sa „user“ musí nachádzať v URL adrese práve raz.

Problémové riadky po zavedení pravidla vyzerajú nasledovne:

```
login := user [ : password ] @ hostport
hostport := hostname [ : port ]
```

Prepis zmenenej gramatiky v BNF pomocou neterminálov definovaných vyššie (zmeny oproti predošlej gramatike sú vyznačené farebne):

```
A → B | C | D | E
B → „http://“ G B`
B` → / J B`` | ? K | ε
B`` → ? K | ε
C → „ftp://“ F / J
D → „telnet://“ F
E → „mailto::“ O @ H
F → L F` @ G
F` → : M | ε
G → H G`
G` → : I | ε
H → O H`
H` → . O H` | ε
I → R
J → N J`
J` → / N J` | ε
K → O K`
K` → + O K` | ε
L → O
M → O
N → PN | ε
O → P O`
O` → P O` | ε
P → S | T
R → T R`
R` → T R` | ε
S → [A-Za-z]
```

$T \rightarrow [0-9]$

V tejto gramatike už neboli nájdené žiadne miesta, ktoré by nespĺňali podmienky LL(1).

Rozhodli sme sa však gramatiku ďalej upraviť, zjednodušiť. Ako prvé sme odstránili zbytočné neterminály s jedným pravidlom, ktorý obsahuje jeden neterminál. To sú neterminály I, L a M. Všetky výskyty týchto neterminálov sme nahradili ich pravidlom. Gramatika po tejto zmene vyzerá nasledovne (zmeny oproti predošlej gramatike sú vyznačené farebne):

$A \rightarrow B \mid C \mid D \mid E$
 $B \rightarrow \text{„http://“ } G B'$
 $B' \rightarrow / J B'' \mid ? K \mid \varepsilon$
 $B'' \rightarrow ? K \mid \varepsilon$
 $C \rightarrow \text{„ftp://“ } F / J$
 $D \rightarrow \text{„telnet://“ } F$
 $E \rightarrow \text{„mailto::“ } O @ H$
 $F \rightarrow O F' @ G$
 $F' \rightarrow : O \mid \varepsilon$
 $G \rightarrow H G'$
 $G' \rightarrow : R \mid \varepsilon$
 $H \rightarrow O H'$
 $H' \rightarrow . O H' \mid \varepsilon$
 $J \rightarrow N J'$
 $J' \rightarrow / N J' \mid \varepsilon$
 $K \rightarrow O K'$
 $K' \rightarrow + O K' \mid \varepsilon$
 $N \rightarrow P N \mid \varepsilon$
 $O \rightarrow P O'$
 $O' \rightarrow P O' \mid \varepsilon$
 $P \rightarrow S \mid T$
 $R \rightarrow T R'$
 $R' \rightarrow T R' \mid \varepsilon$
 $S \rightarrow [A-Za-z]$
 $T \rightarrow [0-9]$

Následne sme gramatiku prepísali na krajší tvar, kde sme zmenili len pomenovanie neterminálov tak, aby išli v abecednom poradí a neobsahovali apostrof:

$A \rightarrow \text{„http://“ } F B \mid \text{„ftp://“ } D / J \mid \text{„telnet://“ } D \mid \text{„mailto::“ } O @ H$
 $B \rightarrow / J C \mid ? L \mid \varepsilon$
 $C \rightarrow ? L \mid \varepsilon$
 $D \rightarrow O E @ F$
 $E \rightarrow : O \mid \varepsilon$
 $F \rightarrow H G$
 $G \rightarrow : R \mid \varepsilon$
 $H \rightarrow O I$
 $I \rightarrow . O I \mid \varepsilon$
 $J \rightarrow N K$
 $K \rightarrow / N K \mid \varepsilon$
 $L \rightarrow O M$

$M \rightarrow + O M \mid \varepsilon$
 $N \rightarrow Q N \mid \varepsilon$
 $O \rightarrow Q P$
 $P \rightarrow Q P \mid \varepsilon$
 $Q \rightarrow U \mid T$
 $R \rightarrow T S$
 $S \rightarrow T S \mid \varepsilon$
 $T \rightarrow [0-9]$
 $U \rightarrow [A-Za-z]$

1.4. Overenie LL(1)

1.4.1. First₁

$\text{First}(U) = \{[A-Za-z]\}$
 $\text{First}(T) = \{[0-9]\}$
 $\text{First}(S) = \text{First}(T) \cup \{\varepsilon\} = \{[0-9]\} \cup \{\varepsilon\} = \{[0-9], \varepsilon\}$
 $\text{First}(R) = \text{First}(T) = \{[0-9]\}$
 $\text{First}(Q) = \text{First}(T) \cup \text{First}(U) = \{[0-9]\} \cup \{[A-Za-z]\} = \{[0-9], [A-Za-z]\}$
 $\text{First}(P) = \text{First}(Q) \cup \{\varepsilon\} = \{[0-9], [A-Za-z]\} \cup \{\varepsilon\} = \{[0-9], [A-Za-z], \varepsilon\}$
 $\text{First}(O) = \text{First}(Q) = \{[0-9], [A-Za-z]\}$
 $\text{First}(N) = \text{First}(Q) \cup \{\varepsilon\} = \{[0-9], [A-Za-z]\} \cup \{\varepsilon\} = \{[0-9], [A-Za-z], \varepsilon\}$
 $\text{First}(M) = \{+, \varepsilon\}$
 $\text{First}(L) = \text{First}(O) = \{[0-9], [A-Za-z]\}$
 $\text{First}(K) = \{/, \varepsilon\}$
 $\text{First}(J) = \text{First}(N) \setminus \varepsilon \cup \text{First}(K) = \{[0-9], [A-Za-z]\} \cup \{/, \varepsilon\} = \{[0-9], [A-Za-z], /, \varepsilon\}$
 $\text{First}(I) = \{., \varepsilon\}$
 $\text{First}(H) = \text{First}(O) = \{[0-9], [A-Za-z]\}$
 $\text{First}(G) = \{:, \varepsilon\}$
 $\text{First}(F) = \text{First}(H) = \{[0-9], [A-Za-z]\}$
 $\text{First}(E) = \{:, \varepsilon\}$
 $\text{First}(D) = \text{First}(O) = \{[0-9], [A-Za-z]\}$
 $\text{First}(C) = \{?, \varepsilon\}$
 $\text{First}(B) = \{/, ?, \varepsilon\}$
 $\text{First}(A) = \{, , \text{http://}, \text{ftp://}, \text{telnet://}, \text{mailto::}\}$

1.4.2. Follow₁

$\text{Follow}(A) = \{\$\}$
 $\text{Follow}(B) = \text{Follow}(A) = \{\$\}$
 $\text{Follow}(C) = \text{Follow}(B) = \{\$\}$
 $\text{Follow}(D) = \text{Follow}(A) \cup \{/ \} = \{\$\} \cup \{/ \} = \{\$, /\}$
 $\text{Follow}(E) = \{ @ \}$
 $\text{Follow}(F) = \text{First}(B) \setminus \varepsilon \cup \text{Follow}(A) \cup \text{Follow}(D) = \{/, ? \} \cup \{\$\} \cup \{\$\} = \{/, ?, \$\}$
 $\text{Follow}(G) = \text{Follow}(F) = \{/, ?, \$\}$

$\text{Follow(H)} = \text{Follow(A)} \cup \text{First(G)} \setminus \epsilon \cup \text{Follow(F)} = \{\$, \} \cup \{:\} \cup \{/, ?, \$\} = \{\$, :, /, ?\}$

$\text{Follow(I)} = \text{Follow(H)} \cup \text{Follow(I)} = \{\$, :, /, ?\}$

$\text{Follow(J)} = \text{Follow(A)} \cup \text{First(C)} \setminus \epsilon \cup \text{Follow(B)} = \{\$, \} \cup \{?\} \cup \{\$\} = \{\$, ?\}$

$\text{Follow(K)} = \text{Follow(J)} \cup \text{Follow(K)} = \{\$, ?\}$

$\text{Follow(L)} = \text{Follow(B)} \cup \text{Follow(C)} = \{\$, \} \cup \{\$\} = \{\$\}$

$\text{Follow(M)} = \text{Follow(L)} \cup \text{Follow(M)} = \{\$\}$

$\text{Follow(N)} = \text{First(K)} \setminus \epsilon \cup \text{Follow(J)} \cup \text{Follow(K)} \cup \text{Follow(N)} = \{ /\} \cup \{\$, ?\} \cup \{\$, ?\} = \{/, \$, ?\}$

$\text{Follow(O)} = \{ @ \} \cup \text{First(E)} \setminus \epsilon \cup \text{Follow(D)} \cup \text{Follow(E)} \cup \text{First(I)} \setminus \epsilon \cup \text{Follow(H)} \cup \text{Follow(I)} \cup \text{First(M)} \setminus \epsilon \cup \text{Follow(L)} \cup \text{Follow(M)}$
 $= \{ @ \} \cup \{ :\} \cup \{\$, /\} \cup \{ @ \} \cup \{ . \} \cup \{\$, :, /, ?\} \cup \{\$, :, /, ?\} \cup \{ + \} \cup \{\$ \} \cup \{\$ \} = \{ @, :, \$, /, ., ?, + \}$

$\text{Follow(P)} = \text{Follow(O)} \cup \text{Follow(P)} = \{ @, :, \$, /, ., ?, + \}$

$\text{Follow(Q)} = \text{First(N)} \setminus \epsilon \cup \text{Follow(N)} \cup \text{First(P)} \setminus \epsilon \cup \text{Follow(O)} \cup \text{Follow(P)}$
 $= \{ [0-9], [A-Za-z] \} \cup \{ /\, \$, ? \} \cup \{ [0-9], [A-Za-z] \} \cup \{ @, :, \$, /, ., ?, + \} \cup \{ @, :, \$, /, ., ?, + \}$
 $= \{ [0-9], [A-Za-z], /, \$, ?, @, :, ., + \}$

$\text{Follow(R)} = \text{Follow(G)} = \{ /\, ?, \$ \}$

$\text{Follow(S)} = \text{Follow(R)} \cup \text{Follow(S)} = \{ /\, ?, \$ \}$

$\text{Follow(T)} = \text{Follow(Q)} \cup \text{First(S)} \setminus \epsilon \cup \text{Follow(R)} \cup \text{Follow(S)} = \{ [0-9], [A-Za-z], /, \$, ?, @, :, ., + \} \cup \{ [0-9] \} \cup \{ /\, ?, \$ \} \cup \{ /\, ?, \$ \}$
 $= \{ [0-9], [A-Za-z], /, \$, ?, @, :, ., + \}$

$\text{Follow(U)} = \text{Follow(Q)} = \{ [0-9], [A-Za-z], /, \$, ?, @, :, ., + \}$

Po analýze množín FIRST_1 a FOLLOW_1 sme sa presvedčili, že gramatika je LL(1).

1.5. Tabuľka prechodov

Pravidlo A

1 : http://FB
2 : ftp://D/J
3 : telnet://D
4 : mailto::O@H

Pravidlo B

5 : /JC
6 : ?L
7 : epsilon

Pravidlo C

8 : ?L
9 : epsilon

Pravidlo D

10 : OE@F

Pravidlo E

11 : :O
12 : epsilon

Pravidlo F

13 : HG

Pravidlo G

14 : :R
15 : epsilon

Pravidlo H

16 : OI

Pravidlo I

17 : .OI
18 : epsilon

Pravidlo J

19 : NK

Pravidlo K

20 : /NK
21 : epsilon

Pravidlo L

22 : OM

Pravidlo M

23 : +OM
24 : epsilon

Pravidlo N

25 : QN
26 : epsilon

Pravidlo O

27 : QP

Pravidlo P

28 : QP
29 : epsilon

Pravidlo Q

30 : U
31 : T

Pravidlo R

32 : TS

Pravidlo S

33 : TS
34 : epsilon

Pravidlo T

35 : [0-9]

Pravidlo U

36 : [A-Za-z]

	[A-Za-z]	[0-9]	+	/	.	:	@	http://	ftp://	telnet://	mailto::	?	\$
A	error	error	error	error	error	error	error	1	2	3	4	error	error
B	error	error	error	5	error	error	error	error	error	error	error	6	7
C	error	error	error	error	error	error	error	error	error	error	error	8	9
D	10	10	error	error	error	error	error	error	error	error	error	error	error
E	error	error	error	error	error	11	12	error	error	error	error	error	error
F	13	13	error	error	error	error	error	error	error	error	error	error	error
G	error	error	error	15	error	14	error	error	error	error	error	15	15
H	16	16	error	error	error	error	error	error	error	error	error	error	error
I	error	error	error	18	17	18	error	error	error	error	error	18	18
J	19	19	error	19	error	error	error	error	error	error	error	19	19
K	error	error	error	20	error	error	error	error	error	error	error	21	21
L	22	22	error	error	error	error	error	error	error	error	error	error	error
M	error	error	23	error	error	error	error	error	error	error	error	error	24
N	25	25	error	26	error	error	error	error	error	error	error	26	26
O	27	27	error	error	error	error	error	error	error	error	error	error	error
P	28	28	29	29	29	29	29	error	error	error	error	29	29
Q	30	31	error	error	error	error	error	error	error	error	error	error	error
R	error	32	error	error	error	error	error	error	error	error	error	error	error
S	error	33	error	34	error	error	error	error	error	error	error	34	34
T	error	35	error	error	error	error	error	error	error	error	error	error	error
U	36	error	error	error	error	error	error	error	error	error	error	error	error

2. Implementácia

Podiel práce:

- Tímea Nagy – 50%
- Michal Zajic – 50%

Implementovali sme tabuľkou riadený syntaktický analyzátor v jazyku JavaScript spolu s používateľským rozhraním a testami. Analyzátor rozpoznáva vety v jazyku simple URL.

2.1. Používateľské rozhranie

Analyzátor je možné spustiť spolu s používateľským rozhraním, ktorého základ je „index.html“ súbor. Používateľ môže zadať ľubovoľný počet URL adresies oddelených pomocou znaku pipe „|“. Po zadaní URL adresy analyzátor vypíše či je adresa platná alebo nie a či sa využilo zotavenie z chýb alebo nie. Ponúka aj možnosť zobrazenia krokov, ako bol vstup spracovaný.

Pri správnom vstupe a bez využitia zotavenia z chýb vyzerá výstup nasledovne:

The screenshot shows a web application titled "Simple URL - Syntactic Analyzer". It has a dark grey background. At the top, it says "Enter URLs below separated by pipe" with an example: "e.g. -> http://test.url/path/to/dir?syntactic+analyzer|ftp://test@url/path/". Below this is a white rounded rectangle containing the input "ftp://test@url/path/". Underneath the input is a purple button labeled "CHECK URL". At the bottom, there is a table with the following data:

Url	Valid	Trace	Recovered
ftp://test@url/path/	true	Show Trace	false

At the very bottom, it says "@copyright Michal Zajic and Tímea Nagy".

Rules	Url
A	ftp://test@url/path/\$
ftp://D/J	ftp://test@url/path/\$
OE@F/J	test@url/path/\$
QPE@F/J	test@url/path/\$
UPE@F/J	test@url/path/\$
[A-Za-z]PE@F/J	test@url/path/\$
QPE@F/J	est@url/path/\$
UPE@F/J	est@url/path/\$

Pri využití zotavenia chýb vyzerá výstup nasledovne:

Simple URL - Syntactic Analyzer

Enter URLs below separated by pipe

e.g. -> http://test.url/path/to/dir?syntactic+analyzer|ftp://test@url/path/

hptt://foo

CHECK URL

Url	Valid	Trace	Recovered
hptt://foo	true	Show Trace	true

@copyright Michal Zajic and Timea Nagy

Rules	Url
A	hptt://foo\$
http://FB	http://foo\$
HGB	foo\$
OIGB	foo\$
QPIGB	foo\$
UPIGB	foo\$
[A-Za-z]PIGB	foo\$
QPIGB	oo\$

2.2. Testy

Pre analyzátor sme napísali testy tak, aby sme pokryli všetky možné URL adresy, ktoré sú platné a niekoľko neplatných adries. Nachádzajú sa v súbore „analyzer.test.js“. Testy sme rozdelili na štyri časti podľa štyroch pravidiel, ktoré sa nachádzajú pri štartovacom neterminály – URL adresy začínajúce na „http://“, „ftp://“, „telnet://“ a „mailto:“. V rámci všetkých týchto typov sme zadefinovali platné URL adresy, adresy, pri ktorých sa využilo zotavenie z chýb a chybné URL adresy. Spustiť ich je možné pomocou príkazu „npm test“.

K URL adresám máme informáciu či je adresa platná alebo nie a či bolo využité zotavenie z chýb alebo nie.

Ukážka platných URL adries:

```
[ "http://a", true, false],
[ "http://1", true, false],
[ "http://Z", true, false],
[ "http://5", true, false],
[ "http://foobarfoo", true, false],
[ "http://1234567", true, false],
[ "http://F00", true, false],
[ "http://fooBARfoo123", true, false],
```

Ukážka URL adries, pri ktorých sa využilo zotavenie:

```
[ "telnet://foo:123456@foo:123456", true, true], // incorrect telnet address
[ "telnet://foo@foo:foo", true, true], // incorrect port
```

Ukážka neplatných URL adries:

```
["mailto:", false, false], // missing xalphas
["mailto:a@", false, false], // missing hostname
["mailto:a", false, false], // missing hostname and @ character
["mailto:a@b.", false, false], // incorrect hostname
["mailto:a.foo.bar@b", false, false], // incorrect xalphas
```

2.3. Tabuľka a pravidlá

Reprezentácia prechodovej tabuľky a pravidiel sa nachádza v súbore „table_and_rules.js“. Má formu *dictionary*. Pre každý neterminál si pamätáme terminály, ktoré vieme vygenerovať pomocou daného neterminálu a k terminálom pravidlo, ktorým musíme pokračovať, aby sme dostali daný terminál. Ak neterminál môže vygenerovať hocijaké písmená alebo čísla, v *dictionary* máme pre tieto terminály špeciálne označenie. Na základe tohto označenia analyzátor na začiatku vygeneruje *key-value* hodnoty do *dictionary* pre každé písmeno alebo číslo. Kód pre generovanie týchto hodnôt sa nachádza v súbore „table_transform.js“.

Ukážka reprezentácia tabuľky pre neterminál S:

```
"S" : {
  "number" : 33,
  "/" : 34,
  "?" : 34,
  "$" : 34
}
```

Ukážka reprezentácie pravidiel:

```
1: "http://FB",
2: "ftp://D/J",
3: "telnet://D",
4: "mailto::O@H",
5: "/JC",
6: "?L",
7: "'",
8: "?L",
```

2.4. Postup implementácie

Samotný proces spracovávania vstupu sa nachádza v súbore „simple_url_syntactic_analyzer.js“. Algoritmus začína predspracovaním vstupu, v rámci ktorého pridáme na koniec vstupu znak „\$“.

Po predspracovaní inicializujeme zásobník, t. j. pridáme štartovací neterminál.

Samotný algoritmus odvodenia prebieha ako rekurzia. Má nasledovný priebeh:

1. Vyberieme prvý znak zo zásobníka, ktorý môže byť terminál alebo neterminál a vyberieme aj prvý znak z nespracovanej časti URL adresy.
2. Ak sú tieto znaky rovnaké a nejedná sa o neterminál, tak odstraňujeme tento znak zo zásobníka a aj z URL adresy. Pri štartovacom pravidle odstraňujeme celý terminál napr. „http://“.
3. Ak sme rekurziu zavolali s regulárnym výrazom napr. [A-Za-z] alebo [0-9], tak skontrolujeme či platí na prvý znak z nespracovanej časti URL adresy. Ak áno, odstránime daný znak.
4. Skontrolujeme či máme epsilonové pravidlo. Ak áno, zavoláme rekurziu znova so zvyšnými pravidlami.
5. Znova vyberieme prvý znak zo zásobníka a vyberieme aj prvý znak z nespracovanej časti URL adresy.
6. Ak narazíme na znak „\$“, teda spracovali sme celú URL adresu a nemáme žiadne ďalšie pravidlá v zásobníku, adresa je validná a rekurziu skončíme.
7. Ak pre aktuálny neterminál nemáme žiadne pravidlo, s ktorým by sme sa dostali k terminálu na začiatku URL adresy, adresa nie je validná a končíme rekurziu.
8. Aplikujeme pravidlo (uložíme do zásobníka) pre aktuálny neterminál na základe terminálu, ktorý sa nachádza na začiatku nespracovanej časti URL adresy.
9. Ak pravidlo je regulárny výraz, musíme ju poslať do rekurzie.
10. Zavoláme rekurziu a celý proces začíname od znova.

2.5. Zotavenie z chýb

Zotavenie chýb nastane, ak sa používateľ pomýli na začiatku URL adresy alebo pri porte.

URL adresa môže začínať len so štyrmi rôznymi reťazcami znakov podľa typu URL adresy - „http://“, „ftp://“, „telnet://“ a „mailto:“. Ak sa používateľ pomýli v tejto časti URL adresy, tak sa pomocou edit distance algoritmu zistí, na ktorú z tých štyroch typov sa URL adresa podobá a na základe toho zmeníme začiatok URL adresy.

Port sa môže skladať len z čísel. Chceme však uznať aj URL adresy, v ktorých sa používateľ pomýlil pri porte a namiesto čísel tam napísal aj písmená. Pridali sme nové pravidlá do tabuľky:

$$\begin{aligned} R &\rightarrow U S \\ S &\rightarrow U S \end{aligned}$$

Kde R a S sú terminály, ktoré generujú port a U generuje veľké a malé písmená.

2.6. Návod na spustenie

Program je vytvorený v jazyku JavaScript. Vyžaduje nainštalovaný Node.js.

Ako prvé je potrebné nainštalovať všetky potrebné moduly, to zabezpečí príkaz „npm install“ spustený v hlavnom priečinku programu. Na spustenie testov slúži príkaz „npm test“. Program obsahuje aj používateľské rozhranie, ktoré je možné spustiť otvorením „index.html“ súboru v prehliadači. Pred tým však odporúčame zadať príkaz „npm run build“.