**Regulární výrazy v pravidlech mohou mít podobu:**

**je-li c jeden znak**

**r reg. výraz**

**s řetězec**

**i identifikátor**

**pak**

**výrazu odpovídá např. pozn.**

**--------------------------------------------------**

**c libov. neoperátorový znak c a**

**\c znak c literálně \\* ne u ANSIC \n \t**

**"s" řetězec s literálně "\*\*"**

**. libov. znak mimo nový řádek a.\*b**

**^ začátek řádky ^abc**

**$ konec řádky abc$**

**[s] libov. znak z s [abc]**

**[x-z] znaky x, y, . . . z. [0-9]**

**[^s] " " " není-li z s [^abc]**

**r\* nula nebo více r a\***

**r+ jeden nebo více r a+**

**r? nula nebo jeden r a?**

**r{m,n} m až n výskytů r a{1,5}**

**r1r2 r1 pak r2 ab**

**r1|r2 r1 nebo r2 a|b**

**(r) r (a|b)**

**r1/r2 r1 je-li n sledováno r2 abc/123**

**{i} překlad i z definiční sekce {PISMENO}**

**--------------------------------------------------**

**yylval proměnná pro předání tokenu do Yacc (ten provádí synt. analýzu)**

**yytext proměnná  obsahující text odpovídajícího reg.výrazu**

**yyleng " " počet znaků "**

**yyless(n) ubere n znaků z yytext[]**

**yymore() přidá  k obsahu yytext[] další koresp. část textu**

**REJECT přejde na další pravidlo bez změny obsahu yytext[]**

**Př.**

**[a-z A-Z][0-9 a-z A-Z]\* { printf(“identifikator”);**

**printf(“jeho tvar je v yytext”);**

**\n pocetradek ++; /\*promenna zavedena v definicni sekci\*/**

**\\ /\* rezezna \ \*/**

**“\” /\* dtto \*/**

**[1-9]+ |**

**+?[1-9]+ printf(“kladne cele cislo”);**

**-[0-9]+ printf(“zaporne cele cislo”);**

**Příklad**

**%{**

**/\* definice manifestovych konstant**

**LT, LE, EQ, NE, GT, GE, IF, THEN, ELSE,**

**ID, NUMBER, RELOP \*/**

**%}**

**/\* regularni definice \*/**

**delim [ \t\n]**

**ws {delim}\***

**letter [A-Za-z]**

**digit [0-9]**

**id {letter}({letter}|{digit})\***

**number {digit}+(\.{digit}+)?(E[+\-]?{digit}+)?**

**%%**

**{ws} {/\* zadna akce ani navrat \*/}**

**if {return(IF);}**

**then {return(THEN);}**

**else {return(ELSE);}**

**{id} {yylval=install\_id(); return(ID);}**

**{number} {yylval=install\_num(); return(NUMBER);}**

**"<=" {yylval=LE; return(RELOP);}**

**"=" {yylval=EQ; return(RELOP);}**

**"<>" {yylval=NE; return(RELOP);}**

**">=" {yylval=GE; return(RELOP):}**

**“<” {yylval=LT; return(RELOP);}**

**“>” {yylval=GT; return(RELOP);}**

**%%**

**install\_id() {**

**/\* vlozi do tabulky symbolu lex.elem.,jehoz prvy**

**znak je urceny v yytext a delka je v yyleng.**

**Vracenou hodnotou je ukazatel do tab.sym. NEROZEPSANA**

**\*/**

**}**

**install\_num() {**

**/\*podobne, pro instalaci cisla\*/**

**}**

**Prostředky pro regulární výrazy jiných programovacích jazyků jsou z Lex**

Př.1) [a-zA-Z][0-9a-zA-Z]\*

|  |  |
| --- | --- |
| JavaScript | "nejaky text".match(/[a-zA-Z][0-9a-zA-Z]\*/g) |
|  | re.exec("nejaky text") |
| PHP | preg\_match\_all('/[a-zA-Z][0-9a-zA-Z]\*/', "nejaky text", $match) |
| Python | re.findall(r'[a-zA-Z][0-9a-zA-Z]\*', "nejaky text") |
| Ruby | "nejaky text".scan(/[a-zA-Z][0-9a-zA-Z]\*/) |

Použito v Python

>>> re.findall(r'[a-zA-Z][0-9a-zA-Z]\*', 'ab1 nic 44')

['ab1', 'nic']

Př. 2) ([+-]?\d\*\.\d+([eE][+-]?\d+)?)

|  |  |
| --- | --- |
| JavaScript | "-2.33e-2alfa11beta12e3?.12E3".replace**(/([+-]?\d\*\.\d+([eE][+-]?\d+)?)**/g, "'expcislo' ") |
| PHP | preg\_replace('**/([+-]?\\d\*\.\d+([eE][+-]?\d+)?)**/', "'expcislo' ", "-2.33e-2alfa11beta12e3?.12E3") |
| Python | re.sub(r'**([+-]?\\d\*\.\d+([eE][+-]?\d+)?)**', "'expcislo' ", "-2.33e-2alfa11beta12e3?.12E3") |
| Ruby | "-2.33e-2alfa11beta12e3?.12E3".gsub(/**([+-]?\\d\*\.\d+([eE][+-]?\d+)?)**/, "'expcislo' ") |

Použito v Python

>>> re.sub(r'([+-]?\d\*\.\d+([eE][+-]?\d+)?)', "'expcislo' ", "-2.33e-2alfa11beta12e3?.12E3")

"'expcislo' alfa11beta12e3?'expcislo' "

>>> re.findall(r'([+-]?\d\*\.\d+([eE][+-]?\d+)?)', "-2.33e-2alfa11beta12e3?.12E3")

[('-2.33e-2', 'e-2'), ('.12E3', 'E3')]

Př. Použití Python modulu re

Spustit Python

>>> import re

>>> re.sub(r'([+-]?\\d\*\.\d+([eE][+-]?\d+)?)', "'expcislo' ", "-2.33e-2alfa11beta12e3?.12E3")

'-2.33e-2alfa11beta12e3?.12E3'

>>> re.findall(r'([+-]?\d\*\.\d+([eE][+-]?\d+)?)', "-2.33e-2alfa11beta12e3?.12E3")

[('-2.33e-2', 'e-2'), ('.12E3', 'E3')]

>>> re.findall(r'[a-zA-Z][0-9a-zA-Z]\*', 'ab1 nic 44')

['ab1', 'nic']

>>> re.sub(r'([+-]?\\d\*\.\d+([eE][+-]?\d+)?)', "'expcislo' ", "-2.33e-2alfa11beta12e3?.12E3")

'-2.33e-2alfa11beta12e3?.12E3'

>>>

Př. Použití Flex

Otevřít DOS okno v adresáři s Flex .exe a příp. i souborem s flex souborem priklad.l

Je to v D:\ProgramFiles(X86)\GnuWin32\bin\

Spustit Flex++ priklad.l nebo Flex priklad.l

Vznikne soubor lex.yy.cc nebo lex.yy.c

Přeložit Dev-C++

…?