Liczby

```
DIGIT    [0-9]
[\-+]?(DIGIT+(\.DIGIT*)?|DIGIT*\.DIGIT+)([eE][\-+]?DIGIT+)?

(pozwala na dane typu:
12
12.
.54
12.12
12.12E12
12.12E+12
12.12E-12
itp.
)
```

prosty skaner dla jezyka pascalo-podobnego

```
%{
#include <math.h>
%}

DIGIT    [0-9]
ID       [a-z][a-z0-9]*

%%
```

prosty skaner dla jezyka pascalo-podobnego

```
 %%
{DIGIT}+    {
            printf( "An integer: %s (%d)\n", yytext,
                    atoi( yytext ) );
            }

{DIGIT}+"."{DIGIT}*         {
            printf( "A float: %s (%g)\n", yytext,
                    atof( yytext ) );
            }

if|then|begin|end|procedure|function        {
            printf( "A keyword: %s\n", yytext );
            }
```

```
{ID}        printf( "An identifier: %s\n", yytext );

"+"|"-"|"*"|"/"   printf( "An operator: %s\n", yytext );

"{"[\^{$\;$}}\n]*"}"     /* komentarze */

[ \t\n]+          /* biale znaki */

.           printf( "Unrecognized character: %s\n", yytext );

%%
```

```
int main( void )
{
  yylex();
}
```

Przyklad kalulatorka

```
%{
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
#include "token.h" // deklaracje token'ow
%}

%option noyywrap
```

```
%%
\n      {
            return END;
         }
[0-9]+  {
            yylval.rval = atof( yytext );
            return NUMBER;
         }
"+"     return PLUS;
"-"     return MINUS;
"*"     return MULTIPLY;
"/"     return DIVIDE;
"("     return LP;
")"     return RP;
%%
```

Prosty skanner dla wyrazen boolowskich

```
%{
#include <stdio.h>
%}

%option noyywrap
```

```
%%

[ \t]       ;

[0-9]+    { yylval.liczba = atoi(yytext); return NUM; } ;

true      { return TRUE; }

false     { return FALSE;} ;

or        { return OR; } ;

and   { return AND; } ;
[a-zA-Z]+ { yylval.zmienna = malloc(yyleng);
```

```
            strcpy(yylval.zmienna, yytext);
            return VAR; };
.           { return yytext[0];};
\n          { return yytext[0]; }
%%
```