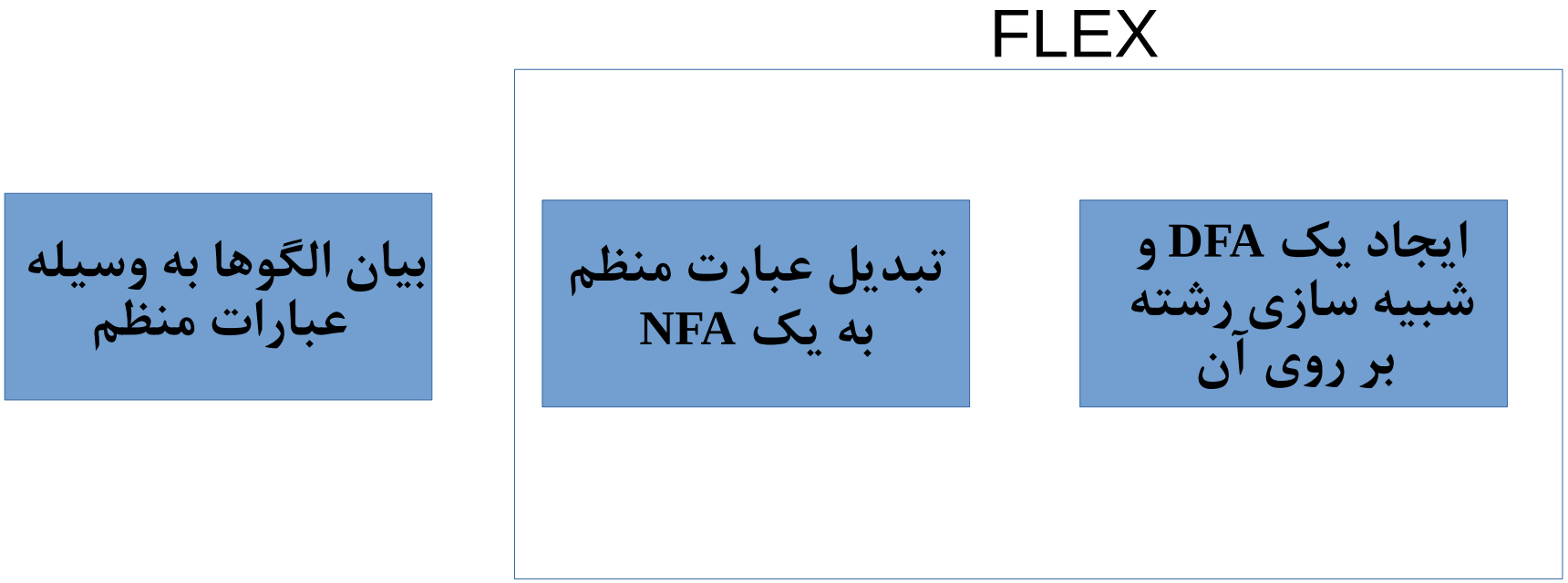
# طراحی اسکنرها به کمک FLEX

صادق اسکندری eskandari@guilan.ac.ir


FLEX
بیان الگوها به وسیله عبارات منظم
تبدیل عبارت منظم به یک NFA
ایجاد یک DFA و شبیه سازی رشته بر روی آن

```
%option noyywrap
%{
int chars = 0;
int words = 0;
int lines = 0;
%}

%%
[a-zA-Z]+       { words++; chars += strlen(yytext); }
\n              { chars++; lines++; }
.               { chars++; }
%%


int main(int argc, char **argv)
{
        if(argc > 1) {
                if(!(yyin = fopen(argv[1], "r"))) {
                        perror(argv[1]);
                        return (1);

                }
        }
        yylex();
        printf("%8d%8d%8d\n", lines, words, chars);

        return 0;
}
```

filename.l فایل

flex filename.l

Lex.yy.c

gcc -o outputfile lex.yy.c  -ll

outputfile
اسکنر (تحلیلگر لغوی)

# مثال

فایل username.l

```
%%
\"username\"    printf( "%s", getlogin() );
%%
```

```
flex username.l
```

```
gcc -o output lex.yy.c -ll
```

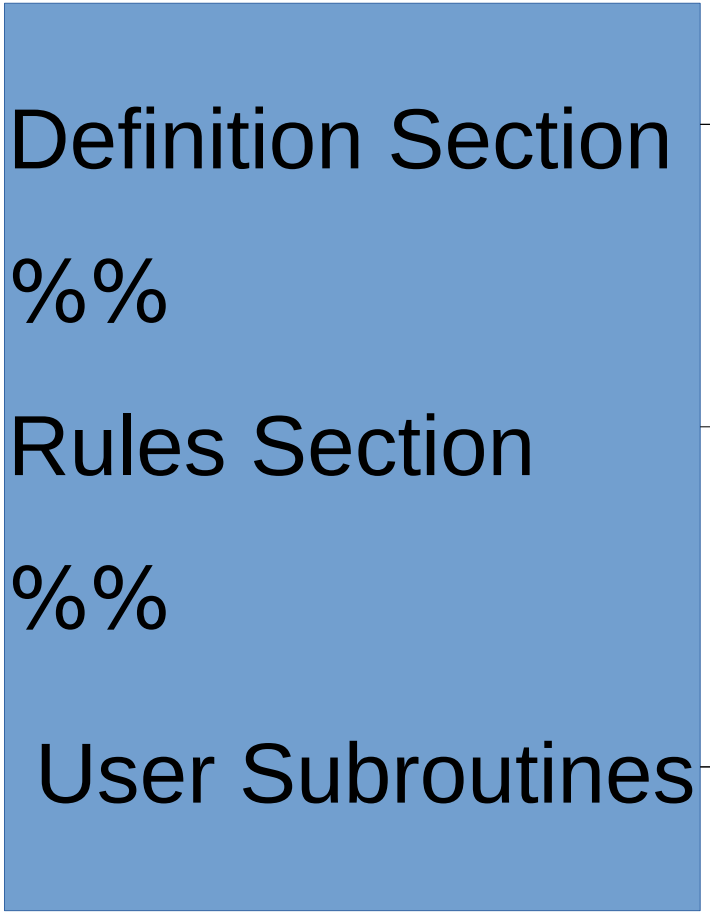
```
hello "username"
your name is "username"
username ====> "username"
```

```
./output <file1.txt >file2.txt
```

```
hello sadegh
your name is sadegh
username ====> sadegh
```

محتویات file1.txt

محتویات file2.txt




حاوی تعاریف، شروط و option ها

حاوی عبارات منظم و کدهای C

زمانی که الگو یافت شد،
کد C متناظر با آن اجرا می‌شود.

حاوی زیر روال هایی است که
توسط بخش قوانین فراخوانی
می‌شوند.

**ساختار یک فایل lex**

```
/* just like Unix wc */
%{
int chars = 0;
int words = 0;
int lines = 0;
%}
%%
[a-zA-Z]+       { words++; chars += strlen(yytext); }
\n              { chars++; lines++; }
.               { chars++; }
%%
main(int argc, char **argv)
{
yylex();
printf("%8d%8d%8d\n", lines, words, chars);
```

```
/* just like Unix wc */
%{
int chars = 0;
int words = 0;
int lines = 0;
%}
%%
[a-zA-Z]+       { words++; chars += strlen(yytext); }
\n              { chars++; lines++; }
.               { chars++; }
%%
main(int argc, char **argv)
{
yylex();
printf("%8d%8d%8d\n", lines, words, chars);
```

```
/* just like Unix wc */
%{
int chars = 0;
int words = 0;
int lines = 0;
%}
%%
[a-zA-Z]+       { words++; chars += strlen(yytext); }
\n              { chars++; lines++; }
.               { chars++; }
%%
main(int argc, char **argv)
{
yylex();
printf("%8d%8d%8d\n", lines, words, chars);
```

## سؤال: این اسکنر چه عملی را انجام میدهد؟

تعداد کاراکتر، کلمات و خطوط را در ورودی مشخص می‌کند.

# الگوها در FLEX

. (نقطه) ← هر چیزی غیر از خط جدید

[A-Z]

[0-9]

[^A-Z] ← هر کاراکتری غیر از A تا Z

a{1,3} ← یک تا سه رخداد a

0{5} ← دقیقاْ ۵ رخداد 0

{Ali} ← کلمه Ali

[ \t]*

[0-9]+

[0-9]|[a-z]

[-+]?[0-9]+

```
%%
[-+]?([0-9]*\.?[0-9]+|[0-9]+\.)(E([-+])?[0-9]+)?        {printf("FN");}
%%
```

الگوی اعداد در زبان Fortran