Курсовая работа по Системному программированию

1. Выбрать целевую платформу для языка программирования

в примере - язык Си под Юникс

2. Придумать язык программирования

в примере:

```
create id 0
create iid 0
create sch 3
create ssh 2
inc sch
loop sch
inc id
loop ssh
inc iid
pool
pool
print id
print iid
```

3. Вручную выполнить трансляцию под целевую платформу

в примере:

```
main()
{
 int id=0;
 int iid=0;
 int sch=3;
 int ssh=2;
 sch++;
 for(;sch>=0;sch--)
 {
  id++;
  for(;ssh>=0;ssh--)
  {
   iid++;
  }
 }
```

```
 printf("%d",id);
 printf("%d",iid);
}
```

4. Разработать лексический анализатор

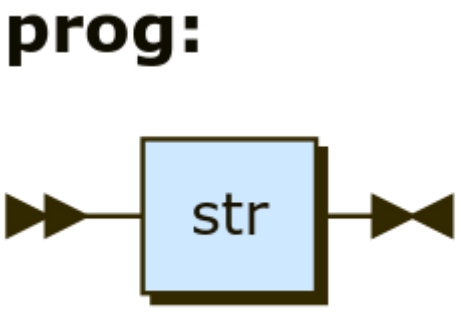
в примере файл forc.l:

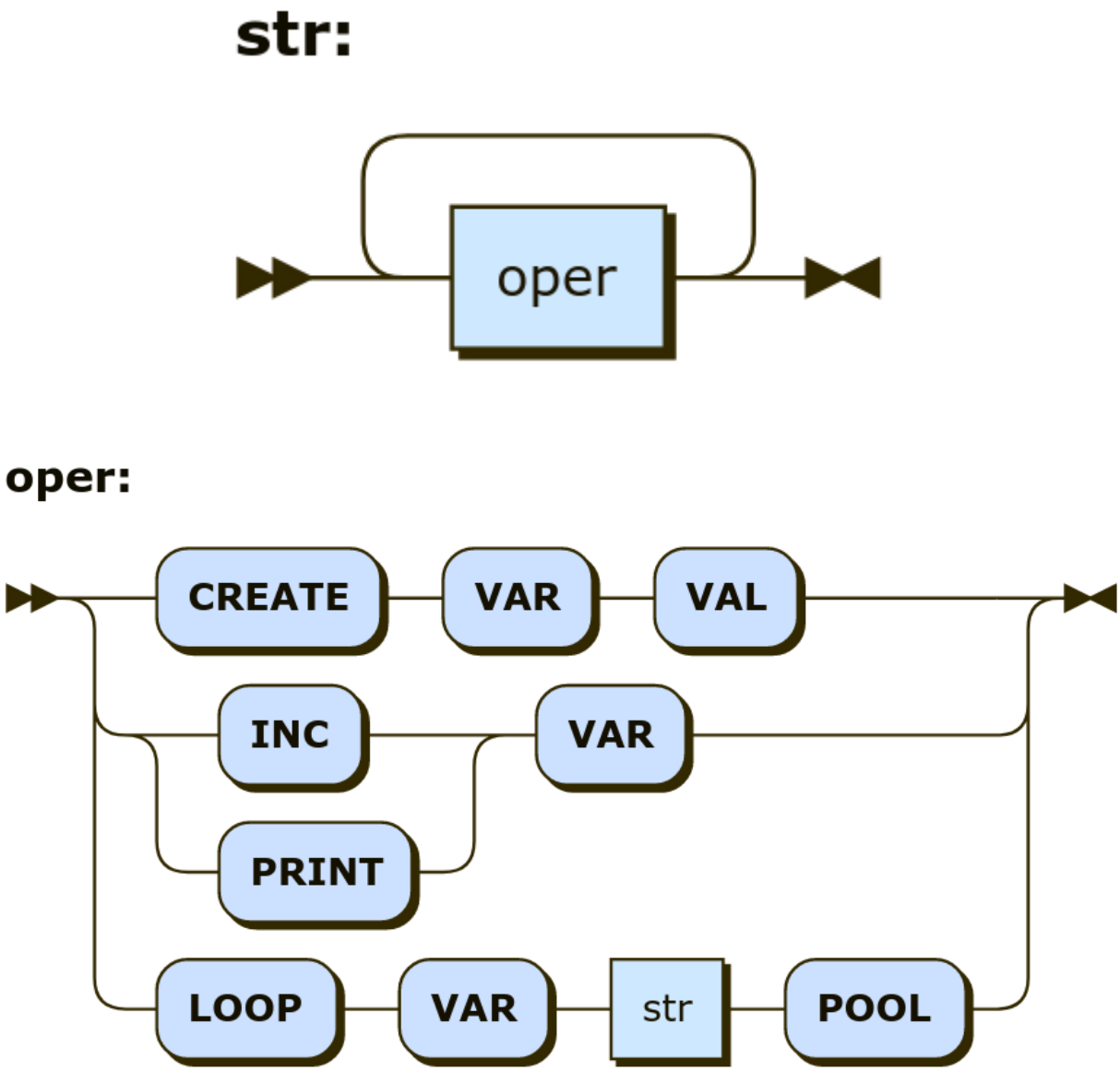
```
%{
#include <stdlib.h>
#include <string.h>
#include "forc_y.h"
%}
letter    [a-z]
digit     [0-9]
%%

create    { return( CREATE ); }
inc       { return( INC );    }
print     { return( PRINT );  }
loop      { return( LOOP );   }
pool      { return( POOL );   }
{letter}+ { strcpy( yylval.var, yytext ) ; return( VAR );   }
{digit}+  { sscanf( yytext, "%d", &yylval.val ) ; return( VAL );   }
[ \n]     {}
. { write(2,yytext,1); return( yytext[0] ); }
```

5. Разработать грамматику синтаксического разбора

в примере:





6. Разработать синтаксический анализатор

в примере файл forc.y:

```
%{
/* #define YYSTYPE double   */
char buf[2048];

%}
%union {
int val;
char var[2048];
}

%token <val> VAL
%token <var> VAR
```

```
%token CREATE INC PRINT LOOP POOL
%type <var> str
%type <var> oper



%% /* Grammar rules and actions follow */

prog:           str             { printf("main() { %s }", $1); }
        ;

str:            oper            {
                        sprintf($$, "%s", $1);

                          }
        |       oper str        {
                                  sprintf($$, "%s\n%s", $1, $2);

                                  }
        ;

oper:           CREATE VAR VAL   {
                        sprintf($$, "int %s=%d;", $<var>2, $<val>3);

                                  }
        |       INC VAR                 {
                                  sprintf($$, "%s++;", $<var>2);
                                  }
        |       LOOP VAR str POOL {
                             sprintf($$, "for(;%s>=0;%s--) {%s}",
                             $<var>2, $<var>2, $<var>3 );
                                   }
        |       PRINT VAR    {
                            sprintf($$, "printf(\"%s\",%s);",
                                   "%d", $<var>2);
                                   }
        ;

%%

yyerror (s)  /* Called by yyparse on error */
    char *s;
{
  printf ("err:%s\n", s);
}

main ()
{
 yyparse ();
}
```

6. Откомпилировать компилятор

В примере:
bison -d -o forc_y.c forc.y
flex -o forc_l.c forc.l
gcc -o forc forc_l.c forc_y.c -lfl
Может понадобиться опция gcc «-Wno-implicit»

7. Откомпилировать пример своего языка

В примере:
./forc <<EOT > forc_out.c
create id 0
create iid 0
create sch 3
create ssh 2
inc sch
loop sch
inc id
loop ssh
inc iid
pool
pool
print id
print iid
EOT

8. Выполнить программу своего языка

В примере:
gcc -o forc_out forc_out.c
./forc_out

9. Сделать Makefile и скрипт для автоматической сборки и проверки компилятора

В примере отсутствует.