24 oct 00 9:30 **exo1.lex** Page 1/1

```
/* remplace 0 par a si suivi par 0 par b sinon. */
/* remplace 1 par c si suivi par 1 par b sinon. */

%option noyywrap

%%

0/0 {printf("a");}
0 {printf("b");}
1/1 {printf("c");}
1 {printf("b");}

%%
main ()
{
  yylex();
}
```

30 oct 01 17:41 **exo2.lex** Page 1/1

```
/* décodage de phi(a)=0 phi(b)=01, phi(c)=11. */

%option noyywrap

C 11
PAIR {C}*

%%

0/{PAIR} {printf("a");}
01/{PAIR} {printf("b");}

{C} {printf("c");}
1 {printf("non déchiffrable");}

%%

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

31 oct 01 9:46 **exo3bis.lex** Page 1/1

```
/* les identificateurs d'un programme java */

%option noyywrap
%{
int niveauComment = 0;
%}

%x COMMENTLIGNE  COMMENTIMBRIQUE

KEYWORDS abstract|boolean|break|byte|case|catch|char|class|continue|default|do|d
ouble|else|extends|false|final|finally|float|for|if|implements|import|instanceof
|int|interface|long|native|new|null|package|private|protected|public|return|shor
t|static|super|switch|synchronized|this|throw|throws|transient|true|try|void|vol
atile|while

NEWLINE "\n"
IDENT [a-zA-Z$]([a-zA-Z$_0-9]*".")*[a-zA-Z$_0-9]+

COMMENTLIGNEDEBUT "//"
COMMENTIMBRIQUEDEBUT "/*"
COMMENTIMBRIQUEFIN "*/"

%%

<INITIAL>{KEYWORDS}|{NEWLINE} {}
<INITIAL>{IDENT} {printf("%s\n",yytext);}
<INITIAL>{COMMENTLIGNEDEBUT} {BEGIN(COMMENTLIGNE);}
<INITIAL>{COMMENTIMBRIQUEDEBUT} {niveauComment++;BEGIN(COMMENTIMBRIQUE);}

<COMMENTLIGNE>{NEWLINE} {BEGIN(INITIAL);}

<COMMENTIMBRIQUE>{COMMENTIMBRIQUEDEBUT} {niveauComment++;}
<COMMENTIMBRIQUE>{NEWLINE} {}
<COMMENTIMBRIQUE>{COMMENTIMBRIQUEFIN} {
  niveauComment--;
  if (niveauComment==0){
    BEGIN(INITIAL);
  }
}


<*>. {}

%%
main ()
{
  yylex();
}
```

31 oct 01 9:43 **exo3.lex** Page 1/1

```
/* les identificateurs d'un programme java */

%option noyywrap

KEYWORDS abstract|boolean|break|byte|case|catch|char|class|continue|default|do|d
ouble|else|extends|false|final|finally|float|for|if|implements|import|instanceof
|int|interface|long|native|new|null|package|private|protected|public|return|shor
t|static|super|switch|synchronized|this|throw|throws|transient|true|try|void|vol
atile|while

NEWLINE "\n"
IDENT [a-zA-Z$]([a-zA-Z$_0-9]*".")*[a-zA-Z$_0-9]+
COMMENTLIGNEDEBUT "//"
COMMENTLIGNE {COMMENTLIGNEDEBUT}.*{NEWLINE}
COMMENTDEBUT "/*"
COMMENTFIN "*/"|"**/"
MOTIF1 {COMMENTDEBUT}[^*]*{COMMENTFIN}
MOTIF2 {COMMENTDEBUT}([^*]*[*]+[^/])*[^*]*{COMMENTFIN}

COMMENT {MOTIF1}|{MOTIF2}
%%

{KEYWORDS}|{NEWLINE} {}
{IDENT} {printf("%s\n",yytext);}
{COMMENTLIGNE} {}
{COMMENT} {}

<*>. {}

%%
main ()
{
  yylex();
}
```

```
/* les identificateurs d'un programme java */

%option noyywrap
%{
int niveauComment = 0;
char **idents;
int nbIdents = 0;
%}

%x COMMENTLIGNE COMMENTIMBRIQUE

KEYWORDS abstract|boolean|break|byte|case|catch|char|class|continue|default|do|double|else|extends|false|final|finally|float|for|if|implements|import|instanceof|int|interface|long|native|new|null|package|private|protected|public|return|short|static|super|switch|synchronized|this|throw|throws|transient|true|try|void|volatile|while

NEWLINE "\n"
DIGIT [0-9]
ENTIER {DIGIT}+
DECIMALE "."{DIGIT}*
SUFFIXE E{ENTIER}
REEL {ENTIER}{DECIMALE}?{SUFFIXE}?
IDENT [a-zA-Z$]([a-zA-Z$_0-9]*("."))*[a-zA-Z$_0-9]*
COMMENTLIGNEDEBUT "//"
COMMENTIMBRIQUEDEBUT "/*"
COMMENTIMBRIQUEFIN "*/"

%%

<INITIAL>{KEYWORDS}|{NEWLINE}|{DIGIT}|{ENTIER}|{REEL} {}
<INITIAL>{IDENT} {
  nbIdents++;
  idents = realloc(idents,nbIdents*(sizeof (char *)));
  idents[nbIdents-1]=strcpy(malloc(strlen(yytext)),yytext);
}
<INITIAL>{COMMENTLIGNEDEBUT} {BEGIN(COMMENTLIGNE);}
<INITIAL>{COMMENTIMBRIQUEDEBUT} {niveauComment++;BEGIN(COMMENTIMBRIQUE);}

<COMMENTLIGNE>{NEWLINE} {BEGIN(INITIAL);}

<COMMENTIMBRIQUE>{COMMENTIMBRIQUEDEBUT} {niveauComment++;}
<COMMENTIMBRIQUE>{NEWLINE} {}
<COMMENTIMBRIQUE>{COMMENTIMBRIQUEFIN} {
  niveauComment--;
  if (niveauComment==0){
    BEGIN(INITIAL);
  }
}

<*>. {}

%%

void afficher(char **liste, int taille)
{
  char **aux;
  for (aux=liste; aux < liste + taille; aux++)
    printf("%s\n",*aux);
}

int compare(void **ch1, void **ch2){
```



```
  return strcmp(*ch1,*ch2);
}

main ()
{
  yylex();
  qsort(idents,
        nbIdents,
        sizeof (char *),
        (int (*)(const void *, const void *))compare);
  afficher(idents,nbIdents);
}
```