%{

  #include<stdio.h>

  #include<string.h>

  #include<ctype.h>

  char lexema[255];

  void yyerror(char \*);

%}

// Especificamos los tokens

%token OTRO IF ELSE LPAR RPAR O I

// Especificamos la gramatica

%%

sentencia: sent\_if | OTRO;

sent\_if: IF LPAR exp RPAR sentencia parte\_else;

parte\_else: ELSE sentencia |  ;

exp: O | I;

%%

void yyerror(char \*msg) {

  printf("error: %s", msg);

}

// Especificamos las reglas de los tokens

int yylex() {

  char c;

  while(1) {

    c = getchar();

    if(c == '\n') continue;

    if(isspace(c)) continue;

    if(c == '(') return LPAR;

    if(c == ')') return RPAR;

    if(c == '0') return O;

    if(c == '1') return I;

    char CADENA\_IF[] = "if";

    if(c == CADENA\_IF[0]) {

      int i = 0, j = 0;

      do {

        lexema[i++] = c;

        c = getchar();

        j++;

      } while (c == CADENA\_IF[j] && j < 2);

      if(j == 2) {

        ungetc(c, stdin);

        lexema[i] == 0;

        return IF;

      }

    }

    char CADENA\_ELSE[] = "else";

    if(c == CADENA\_ELSE[0]) {

      int i = 0, j = 0;

      do {

        lexema[i++] = c;

        c = getchar();

        j++;

      } while (c == CADENA\_ELSE[j] && j < 4);

      if(j == 4) {

        ungetc(c, stdin);

        lexema[i] == 0;

        return ELSE;

      }

    }

    char CADENA\_OTRO[] = "otro";

    if(c == CADENA\_OTRO[0]) {

      int i = 0, j = 0;

      do {

        lexema[i++] = c;

        c = getchar();

        j++;

      } while (c == CADENA\_OTRO[j] && j < 4);

      if(j == 4) {

        ungetc(c, stdin);

        lexema[i] == 0;

        return OTRO;

      }

    }

    return c;

  }

}

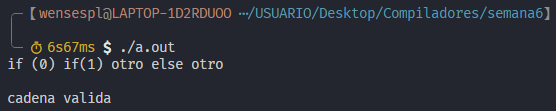
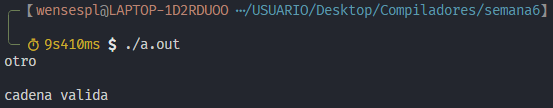
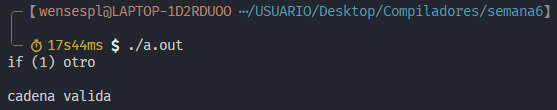
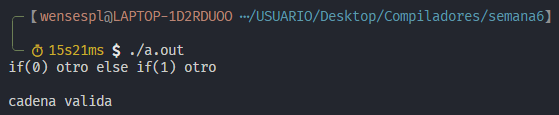
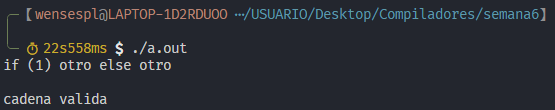
int main() {

  if(!yyparse()) printf("\ncadena valida\n");

  else printf("cadena invalida\n");

  return 0;

}



Penadillo Lazares Wenses Johan