Departamento de Matemática da Universidade de Coimbra		
2004/2005	**Compiladores**	**Folha 2** (v1.5)

1. Escreva expressões regulares para as seguintes linguagens e implemente autómatos finitos em *C* para as mesmas:
   (a) palavras com $V = \{a, b, c\}$ em que o primeiro "a" precede o primeiro "b".
   (b) palavras com $V = \{a, b, c\}$ com número par de "a"s.
   (c) números binários múltiplos de 4.
   (d) números binários maiores que 101001.
   (e) palavras em $V = \{a, b, c\}$ que não contêm a sub-string "baa".

2. Para as seguintes expressões regulares, implemente em *C* um autómato finito determinístico que as reconheça:

   ```
   if  then  else  [a-z][a-z0-9]*  [0-9]+  ("--"[a-z]*"\n")|(" "|"\n"|"\t")
   ```

3. Especifique em *flex* autómatos que reconheçam as diferentes expressões dadas nas várias alíneas das perguntas anteriores.

4. Implementar um programa *wc* ("word count") que retorne o número de linhas, palavras e caracteres escritas no *stdin*. Altere o programa para poder fazer o mesmo para um ficheiro dado na linha de comando.

5. Re-escrever o exemplo anterior por forma a poder processar vários ficheiros em sucessão (sugestão, re-escrever `yywrap()`).

6. Implementar um programa que elimine de um programa em *C* todos os comentários e caracteres de espaço.

7. Implementar um pré-processador de *C*, que expanda macros definidas por `#define`, ficheiros incluídos com `#include` e elimine os comentários.

8. Implementar um analisador léxico para um sub-conjunto da linguagem *C*, com os construtores usuais:

   ```
   for while break if then else ...
   ```

   e a pontuação usual:

   ```
   , : ; ( ) { } [ ] < > <= >= == != ...
   ```

   Teste o analisador obtido com programas em *C*.

9. implementar um analisador léxico para a linguagem *Tiger*, cujos construtores são:

   ```
   while for to break let in end function var type array if then
   else do of nil
   ```

e os operadores são:

```
, : ; ( ) { } [ ] . + - * / = <> < > <= >= == != & | :=
```

os comentários são idênticos ao *C*. Experimente nos programas exemplo:

`merge.tig` e `queens.tig` (ver no fim da folha).

merge.tig

```
let

 type any = {any : int}
 var buffer := getchar()

function readint(any: any) : int =
 let var i := 0
     function isdigit(s : string) : int =
                 ord(buffer)>=ord("0") & ord(buffer)<=ord("9")
     function skipto() =
       while buffer=" " | buffer="\n"
         do buffer := getchar()
  in skipto();
     any.any := isdigit(buffer);
     while isdigit(buffer)
       do (i := i*10+ord(buffer)-ord("0"); buffer := getchar());
     i
 end

 type list = {first: int, rest: list}

 function readlist() : list =
    let var any := any{any=0}
        var i := readint(any)
     in if any.any
         then list{first=i,rest=readlist()}
         else nil
    end

 function merge(a: list, b: list) : list =
   if a=nil then b
   else if b=nil then a
   else if a.first < b.first
      then list{first=a.first,rest=merge(a.rest,b)}
      else list{first=b.first,rest=merge(a,b.rest)}

 function printint(i: int) =
  let function f(i:int) = if i>0
            then (f(i/10); print(chr(i-i/10*10+ord("0"))))
   in if i<0 then (print("-"); f(-i))
      else if i>0 then f(i)
      else print("0")
 end

 function printlist(l: list) =
   if l=nil then print("\n")
   else (printint(l.first); print(" "); printlist(l.rest))
```

queens.tig

```
/* A program to solve the 8-queens problem */

let
    var N := 8

    type intArray = array of int

    var row := intArray [ N ] of 0
    var col := intArray [ N ] of 0
    var diag1 := intArray [N+N-1] of 0
    var diag2 := intArray [N+N-1] of 0

    function printboard() =
       (for i := 0 to N-1
         do (for j := 0 to N-1
              do print(if col[i]=j then " O" else " .");
            print("\n"));
        print("\n"))

    function try(c:int) =
( /*  for i:= 0 to c do print("."); print("\n"); flush();*/
     if c=N
     then printboard()
     else for r := 0 to N-1
           do if row[r]=0 & diag1[r+c]=0 & diag2[r+7-c]=0
                 then (row[r]:=1; diag1[r+c]:=1; diag2[r+7-c]:=1;
                       col[c]:=r;
                       try(c+1);
                       row[r]:=0; diag1[r+c]:=0; diag2[r+7-c]:=0)

)
 in try(0)
end
```