I file in C (II)

Laboratorio di Linguaggi di Programmazione
a.a. 2001/2002

dott.ssa Francesca A. Lisi
lisi@di.uniba.it

Sommario

- I file di testo (ancora)
  - realizzazione di estrattori di *token*
  - realizzazione di analizzatori lessicali

Riferimenti

- capp. 11-12 di Deitel & Deitel
- par. 7.5 di Kernighan & Ritchie

Estrazione di *token* da file di testo (I)

```
#include <iostream.h>
#include <stdlib.h>
#include <stdio.h>
#include <ctype.h>

#define MAXTOKEN 256

struct token {
   char name[MAXTOKEN+1];
   int cat;};

int gettoken(FILE *src, struct token *tkn,
  int max);
```

Estrazione di *token* da file di testo (II)

```
int main() {
   FILE *source;
   char *categories[] = {"3n caratteri",
     "3n+1 caratteri","3n+2 caratteri"};
   struct token next;

   source = fopen("sorgente","r");
   while (gettoken(source, &next, MAXTOKEN))
      printf("Token: %s -> Category: %s\n",
          next.name,categories[next.cat]);
   fclose(source);
   system("PAUSE");
   return 0;}
```

Estrazione di *token* da file di testo (III)

```
int gettoken(FILE *src, struct token *tkn, int max)
  {
   int car, i=0;
   if ((car = getc(src)) != EOF) {
      while (isspace(car)) car = getc(src);
      while (car != EOF && !isspace(car) && i <=
  max) {
         tkn->name[i] = car; i++;
         car = getc(src);
      }
      tkn->name[i] = '\0';
      tkn->cat = i % 3;
      return 1;}
   else return 0;}
```

Realizzazione di analizzatori lessicali

- Implementazione di un automa
  - Strutture a scelta multipla nidificate
  - Output: token + classe
- Necessità di *lookahead*
  - Conservazione di caratteri letti ma non usati

Realizzazione di un analizzatore lessicale: esercizio

Si consideri l'alfabeto A del linguaggio **miniP (**un *subset* del Pascal)

parole chiave BEGIN, BOOLEAN, CHAR, DO, ELSE, END, FALSE, IF, INTEGER, PROGRAM, REAL, THEN, TRUE, VAR, WHILE

operatori e separatori /, *, +, -, >, >=, <, <=, <>, =, (,), :, :=, ;, . (fine programma), fine file, spazi bianchi, commenti

altri token <identificatore>, <carattere>, <intero>, <reale>

Realizzazione di un analizzatore lessicale: esercizio (II)

- Assegnare un codice ad ogni token di A
- Disegnare l'automa R che riconosce A
- Scrivere un programma in **C** che implementa R
  - Dato un file di testo F, deve stabilire se F contiene solo e soltanto simboli di A o no
- Scrivere un programma in **C** che costruisca la tabella dei simboli (inserimento degli identificatori) mentre fa l'analisi lessicale di F