ALGORITMI E STRUTTURE DATI

Esercitazioni

Andrea Orlandini

http://www.dia.uniroma3.it/~orlandin/asd/

e-mail: orlandin@dia.uniroma3.it

Orario di ricevimento: Martedì 14.00 - 16.00

File in C

StudentiDIA Forum

http://forum.studentidia.org/

Dove trovare i lucidi presentati a lezione

http://limongelli.dia.uniroma3.it/asd/

Input/Output da file

- ANSI C definisce un set completo di funzioni per leggere e scrivere qualsiasi tipo di dato da file (con il termine file si intende sia un file su disco che un terminale o un dispositivo)
- Un file (vero e proprio) su disco permette l'accesso casuale
- Indicatore di posizione nel file

- STDIO.H definisce il tipo FILE (Puntatore a file)

```
FILE *fp;
```

Operazioni su file

- Apertura di un file mediante *f*open

```
FILE *fopen(const char *nome_file,const char *modo);
```

- *nome_file* è una stringa che identifica il nome del file e *modo* indica il metodo di accesso
- Alcune modalità di accesso:

```
r   Apre un file di testo per la lettura
w   Crea un file di testo per la scrittura
a   Aggiunge a un file di testo alla fine
r+  Apre un file di testo per la lettura/scrittura
w+  Crea un file di testo per la lettura/scrittura
```

Gestione FILE

- Controllo di apertura corretta

```
if ((fp = fopen(‘‘prova’’,’’w’’))==NULL){
        printf(‘‘Impossibile aprire il file\n’’);
        exit(1);
}
```

- Chiusura di un file mediante *fclose*

```
int fclose(FILE *fp);
```

- *fp* è il puntatore al file che viene chiuso

- *feof* consente di verificare la condizione di fine file

```
int feof(FILE *fp);
```

- la *feof* restituisce TRUE se siamo giunti alla fine del file FALSE altrimenti

Lettura e Scrittura

- Di caratteri (getc/putc)

```
int getc(FILE *fp);
int putc(int car, FILE *fp);

ch = getc(fp);
...
putc(ch,fp);
```

- Di stringhe (fgets/fputs)

```
char *fgets(char *str, int lungh, FILE *fp);
int fputs(const char*str, FILE *fp);

fgets(str,79,fp);
...
fputs(str,fp);
```

- Generiche

```
int fscanf(FILE *fp, const char *stringa_di_controllo,...);
int fprintf(FILE *fp, const char *stringa_di_controllo,...);

fscanf(fp,''%s'',s);
fprintf(fp,''%s'',s);
```

Lettura di liste da file

- Lettura di un file e costruzione di una lista

```
void ScriviElemento(FILE *fi, TipoElemLista elem)
{
  putc(elem, fi);
}

int LeggiElemento(FILE *fi, TipoElemLista *elem)
{
  *elem = getc(fi);
  return *elem;
}
```

```
void LeggiLista(char *nomefile, TipoLista *lis)
{
  TipoLista paux;   TipoElemLista elem;   FILE *datafile;

  datafile = fopen(nomefile, "r");
  if (datafile == NULL) {
    fprintf(stderr,"Err: su %s in lettura\n",nomefile);
    *lis = NULL;
  }
  else { /* File aperto correttamente */
    *lis = malloc(sizeof(TipoNodoLista));
    paux = *lis;

    while (LeggiElemento(datafile, &elem) != EOF) {
      paux->next = malloc(sizeof(TipoNodoLista));
      paux = paux->next;
      paux->info = elem;
    }
    paux->next = NULL;

    paux = *lis;   *lis = (*lis)->next;  free(paux);

    fclose(datafile);
  }
}
```

Scrittura su file di una lista

```
void ScriviLista(char *nomefile, TipoLista lis)
{
  FILE *datafile;

  datafile = fopen(nomefile, "w");
  if (datafile == NULL)
    fprintf(stderr,
            "Err: su %s in scrittura\n",nomefile);
  else {
    while (lis != NULL) {
      ScriviElemento(datafile, lis->info);
      lis = lis->next;
    }
    fclose(datafile);
  }
}
```

Esempio di esecuzione

- Operiamo sulla lista creata dalla lettura di un file

```
main()
{
  char s[80]; TipoLista lista1;

  strcpy(s,"in.tab"); LeggiLista(s,&lista1);

  InvertiLista(&lista1);

  strcpy(s,"out.tab"); ScriviLista(s,lista1);
}
```

- il file di input è in.tab

```
algoritmi
e
strutture
dati
```

- il file di output è out.tab

```
itad
erutturts
e
imtirogla
```