Video Lezione n.18
Linguaggio C/C++
I files di testo esempi e casi d’uso

a cura del prof. Giuseppe Sportelli

Che cos’è un file

- Un insieme di informazioni memorizzati in un supporto di memoria di massa (ad es. Disco Rigido)
- In ogni Sistema Operativo i files sono identificati da un nome e da un’estensione
- Il nome ad esempio in Windows è CASE Insensitive in Linux/Mac OS no
- L’estensione generalmente indica l’applicazione che ha generato il file non è obbligatoria ma è consigliabile

Tipoolgia di File

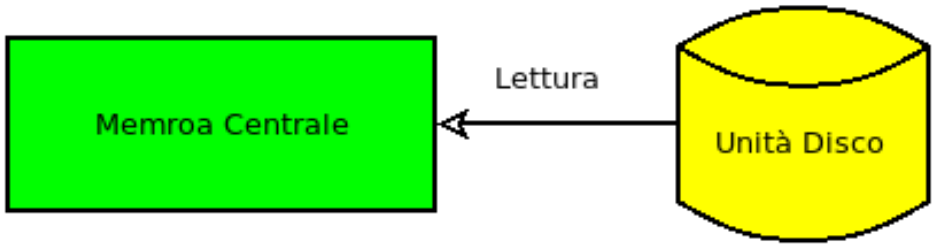
- In tutti i linguaggi di programmazione i files si possono suddividere in:
  - File di testo
    - senza formato (le informazioni sono solo testuali)
    - Con formato (le informazioni sono da interpretare)
  - File binari
    - File costituiti da sequenza di byte
    - La lettura e la scrittura prevede sempre l’interpretazione attraverso il programma ne fa uso

Modalità di Accesso

- Un’altra distinzione forte è la differenza nella modalità di accesso che può essere:
  - Sequenziale (i dati sono letti e scritti in modo consecutivo)
  - Casuale e/o Diretto (la lettura e la scrittura avviene mediante il posizionamento all’indirizzo desiderato)
  - Relative o Calcolato (l’indirizzo è calcolato sulla base di un criterio relativo alle informazioni presenti nel file

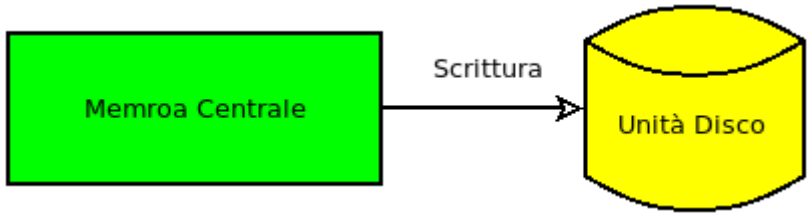
Scrittura e Lettura

La lettura di un file prevede il trasferimento dalla memoria di massa alla memoria centrale

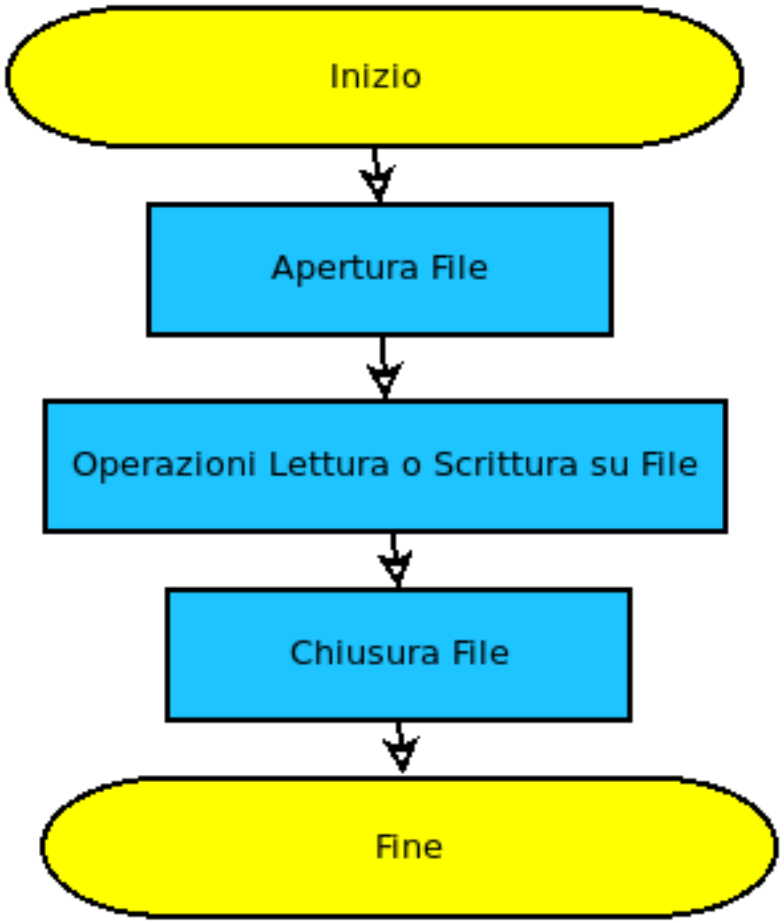


Scrittura e Lettura

La scrittura di un file prevede il trasferimento dalla memoria centrale alla memoria di massa



Sequenza Operativa per i file



- Apertura file collegare il file fisico (presente sul supporto) al descrittore di file nel programma (identificativo file logico)
- Effettuare mediante dei cicli di lettura o scrittura le operazioni
- Chiusura del file e de-allocazione risorsa

Istruzioni e Funzioni in C per i file

- Includere la libreria stdio.h
- Dichiarare un file mediante la parola chiave: FILE *descrittore file; ad esempio FILE *fp;
  - fopen(nome_file,”permessi”,descrittore file) ad esempio fopen(“prova.txt”,”r”,fp);
    - Apre il file in lettura se permesso “r, in scrittura “w”, “a” aggiunta, “w+” lettura scrittura per i file di testo
    - In modalità binaria occorre aggiungere ai permessi il flag “b”; ad esempio “rb” apre un file binario in lettura

Lettura e scrittura di caratteri

- Le istruzioni fgets e e fputs sono per la lettura e la scrittura di sequenze di caratteri
- Tutti i dati letti e scritti sono trattati come stringhe
- L'istruzione fgets(nome buffer,lunghezza,descrittore file) ad esempio
  char buffer[10];
  fgets(buffer,10,fp); legge in buffer 10 caratteri del file aperto in lettura

Lettura e scrittura di caratteri

- Per la scrittura occorre ovviamente aprire il file in scrittura e poi è possibile utilizzare l'istruzione fputs(buffer,descrittore file);
- Ad esempio:

```
char buffer[20];
printf("\n Inserisci la stringa");
scanf("%s",buffer);
fputs(buffer,fp);
```

Annotazione sull'input

- Per inserire stringhe con spaziature non conviene utilizzare l'istruzione scanf ma gets già per altro vista in precedenza;
- Un richiamo ulteriore può venire dall'uso di fgets(buffer, lungheza, stdin);
  Ad esempio il codice previene problemi di buffer:

```
#include <stdio.h>
#include <string.h>
char buffer[20];
int main (int argc, char *argv[])
{
        printf("\n Digita la strigna:");
        fgets(buffer,20,stdin);
        printf("\n Quello che hai digitato:");
        printf("%s",buffer);
        return 0;
}
```

Esempio lettura e scrittura numeri

- Attraverso le funzioni fgets, ftputs, fopen, fclose (chiusura file fclose(descrittore del file) scrivere un file di n numeri e ristamparlo a video dopo averlo letto.

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
FILE *fp;
int main (int argc, char *argv[])
{int n,k,num;
char s[10];
        printf("\n Apertura file di testo in scrittura/aggiunta");
        fp=fopen("dati.txt","wa");
        printf("\n Quanti numeri vuoi memorizzare ? ");
        scanf("%d",&n);
        for (k=0;k<n;k++)
                { printf("\n Inserisci un numero ");
                  scanf("%d",&num);
                sprintf(s,"%d",num);
                strcat(s,"\n");
                 fputs(s,fp);}
        fclose(fp);
        printf("\n File Salvato correttamente\n");
        printf("\n Apertura in lettura del file ....");
        fp=fopen("dati.txt","r");
        while(fgets(s,10,fp)>0)
                { num=atoi(s);
                printf("\nNumero Letti: %d ",num);}
        fclose(fp);
        printf("\n Arrivederci\n");
        return 0;
}
```

Commenti sul codice

- Sono include string.h per la strcat che concatena due stringhe, la stdlib.h per la funzione atoi che converte una stringa di cifre in numero ad esempio num=atoi(s);
- E' usta l'istruzione sprintf che converte un qualunque formato in stringa nell'esempio
  sprintf(s,"%d",num); occorre specificare il formato di input all'istruzione che è intero e la stringha di uscita
- Nell'input dei dati è fissato il numero delle iterazioni (n è richiesto in input)
- Nella lettura dei dati è utilizzato il ciclo while che termina quanto l'istruzione fputs restituisce -1 end of file

File di testo con formato

- Sono file che memorizzano i dati secondo il loro formato sono distinti dati numerici da quelli alfanumerici
- Le istruzioni per la lettura fscanf e per la scrittura fprintf
- fscanf(descrittore file,formato, variabili)
  ad esempio fscanf(fp,”%d”,&num); legge dati in formato numerico intero
- fprintf(descrittore file, formato, variabili);
  ad esempio fprintf(fp,”%d”,num);

Esempio lettura e scrittura

Scrivere e leggere un file di numeri interi. I dati sono letti da tastiera e poi scritti nel file. La lettura dei dati avviene da file la stampa a video.

```c
#include <stdio.h>
FILE *fp;
int main (int argc, char *argv[])
{int n,k,num;
        printf("\n Apertura file di testo in scrittura/aggiunta");
        fp=fopen("dati.txt","wa");
        printf("\n Quanti numeri vuoi memorizzare ? ");
        scanf("%d",&n);
        for (k=0;k<n;k++)
                { printf("\n Inserisci un numero ");
                  scanf("%d",&num);
                 fprintf(fp,"%d\n",num);}
        fclose(fp);
        printf("\n File Salvato correttamente\n");
        printf("\n Apertura in lettura del file ....");
        fp=fopen("dati.txt","r");
        while(fscanf(fp,"%d",&num)>0)
                printf("\nNumero Letto: %d ",num);
        fclose(fp);
        printf("\n Arrivederci\n");
        return 0;
}
```

Commenti sul codice

- Le funzioni fscanf e fprintf consentono di leggere e scrivere dati da e verso il file
- Per le stringhe non server la “&” come nella funzione scanf in fscanf; infatti una stringa è una indirizzo di memoria ove allocare le celle contigue

Nel C++ ?

- Occorre utilizzare la libreria fstream
- I file come descrittori possono essere solo di input solo di output o entrambi con le seguenti dichiarazioni:

```
ifstream nome_descrittore_file;
ofstream nome_descrittore_file;
fstream nome_descrittore;
```

Lettura e Scrittura

- La scrittura avviene con la sintassi descrittore_file << varibile1<<...<<variailen;
- E’ opportuno separare i dati da separatori come “\n” se sono dati con formato
- La lettura avviene con l’istruzione descrittore_file>>variabile1>>...>>variabilen

Caso di studio

```
#include <iostream>
#include <fstream>
#include <string>
using namespace std;
fstream fp;
int n,k;
int age;
string cognome, nome;
int main()
        {
        fp.open("dati.txt",ios::out);
        cout<< "Quanti alunni\n"<<endl;
        cin >> n;
        for (k=0;k<n;k++)
        { cout <<"\n Inserisci nome alunno:";
         getchar();
          getline(cin,nome);
          cout << "\n Inserisci cognome alunno:";
          getline(cin,cognome);
          cout << "\n Inserisci l'età:";
          cin >> age;
          fp << nome << "\n" <<  cognome <<"\n"<< age;}
         fp.close();
        fp.open("dati.txt",ios::in);
        cout << "Dati alunni:\n";
        while (!fp.eof())
        {
                fp >> nome >> cognome >> age;
           cout << "\nNome Alunno:"<<nome;
           cout << "\nCognome Alunno:"<<cognome;
           cout << "\nEtà alunno:"<<age;
        }
        cout << "\nGrazie per aver utilizzato il programma\n";
        return 0;
}
```

Questo esercizio chiede all'utente di inserire dei dati all'interno di un file di testo. Una volta inseriti i dati occorrre ristamparli. Nell'esempio posto è essenziale l'uso del seperatore "\n" per appunto distinguere all'interno del file i dati.

Commenti

- E’ utilizzata la funzione getline() per prevedere l’input di stringhe con spazi
- Il ciclo di lettura deve essere sempre prevedere una struttura del tipo

```
leggi_dati_da_file
mentre(non finito il file)
    leggi_dati_da_file
```

per evitare che il ciclo un loop infinito vale in generale

Fine Video Lezione

- Argomenti trattati:
  - File Sequenziali
  - File di testo in C
  - File di testo in C++
  - Esempi pratici
- Nelle prossime video lezioni
  - File binari in C e C++
  - Uso di file ad accesso diretto

Fine Video Lezione

Grazie per l’attenzione
Prof. Giuseppe Sportelli
Sito web https://www.giuseppesportelli.it
Revisione Febbraio 2020

Apertura del file in C++

- nome_file.open(nome_file_fisico,ios:accesso);
  ios:in sola lettura
  ios:out sola scrittura
  ios:app aggiunta
  ios:binary apertura del file in binario
- E’ possibile combinare più flag con l’operatore “|”
- Ad esempio nome_file.open(“dati.txt”,ios:out);