Variante esercizio p.332 n.11 – ATLAS – Lorenzi,Moriggia – Informatica per Ist.Tecnici Tecnologici – vol.A

Leggere dal file di testo **"alunni_INP.txt"** i nominativi (cognome e nome) dei nuovi alunni della scuola; **per ogni alunno** esporre a video il nome dell'alunno, chiedere la digitazione della classe da assegnare all'alunno e registrare nel **file binary "alunni_OUT.dat"** un record di tipo Studente (22 char per il nome, 6 char per la classe).
Completata l'elaborazione del file di input, chiudere i due file, aprire **in input** il **file binary "alunni_OUT.dat"** appena creato ed esporre a video l'elenco dei nuovi alunni.

alunni_INP.txt

```
Baldi Luigi
Carli Franco
Franchi Anna Maria
Sesti Carlo
```

Output a video

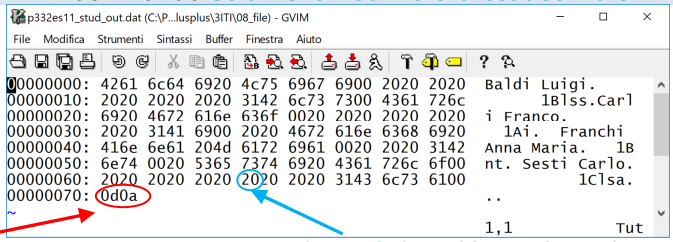
```
per ogni studente, assegnare
la classe:
Baldi Luigi      classe? 1Blss
Carli Franco     classe? 1Ai
Franchi Anna Maria classe? 1Bnt
Sesti Carlo      classe? 1Clsa

 Elenco studenti registrati
Baldi Luigi              1Blss
Carli Franco             1Ai
Franchi Anna Maria       1Bnt
Sesti Carlo              1Clsa
```

i record NON sono separati: c'è solo 1 ACAPO (0d0a) a fine file

alunni_OUT.dat

```
Baldi Luigi          1Blss Carli Franco          1Ai
Franchi Anna Maria   1Bnt  Sesti Carlo           1Clsa
```

alunni_OUT.dat visto con ***VIM***
utilizzando *Strumenti>Converti a esadecimale*



Nota: lo **spazio** ha codifica **20** (in esadecimale)

pseudocodifica

```
INIZIO
  APRI (finp, "alunni_INP.txt", INPUT)
  SE finp.fail()
  ALLORA
    SCRIVI ("il file non esiste")
    RITORNO-RESTITUISCI -1
  FINE SE
  APRI (fout, "alunni_OUT.dat", OUTPUT, BINARY)
  LEGGI (finp, alunno.nome)
  ESEGUI MENTRE NOT finp.eof()
        SCRIVI (alunno.nome, "classe? ")
        LEGGI (alunno.classe)
        SCRIVI (fout, alunno)
        LEGGI (finp, alunno.nome)
  RIPETI
  CHIUDI (finp, fout)
  APRI (finp, "alunni_OUT.dat", INPUT, BINARY)
  LEGGI (finp, alunno)
  ESEGUI MENTRE NOT finp.eof()
        SCRIVI (alunno.nome, alunno.classe)
        LEGGI (finp, alunno)
  RIPETI
  CHIUDI (finp)
FINE
```

codifica (parte 1)

```
#include <iostream>
using namespace std;
#include <fstream>
#include <string.h>
#include <iomanip>
#define LN 22

struct Studente {
  char        nome[LN];
  char        classe[6];
};

void clean(Studente& al){
     for (int i=0; i<LN; i++)
          al.nome[i]  = ' '; //spazio
     for (int i=0; i<6; i++)
          al.classe[i]= ' '; //spazio
}
```

codifica (parte 2)

```
int main(){
  Studente alunno;
  fstream finp, fout;   //--- dichiarazione generica nomi file
                        //--- apertura esplicita in INPUT di finp
   finp.open("alunni_INP.txt", ios::in);
   if (finp.fail()) {
      cout << "il file non esiste\n";
      return -1;
   }
   fout.open("alunni_OUT.dat", ios::out | ios::binary);
   cout<<"\nper ogni studente, assegnare la classe:\n";

   clean(alunno);                    //--- spazi in nome e classe
   finp.getline(alunno.nome, LN);    //--- lettura fuori ciclo

   while ( ! finp.eof() )  {         //--- scrittura a video
        cout<< alunno.nome << '\t' << "classe? ";
        cin >> alunno.classe;        //--- lettura da tastiera
        fout.write( (char *) &alunno, sizeof(alunno) );
        clean(alunno);
        finp.getline(alunno.nome, LN); //--- lettura in ciclo
   }
   finp.close();
   fout.close();

   finp.open("alunni_OUT.dat", ios::in | ios::binary);
   if ( finp.fail() )
      cout<<"\n Errore nell'apertura dell'archivio\n";

   else {
      cout<<"\n Elenco studenti registrati\n";
      finp.read( (char *) &alunno, sizeof(alunno) );
                                    //--- lettura fuori ciclo
      while ( ! finp.eof() )  {

         cout<<left <<setw(LN)<< alunno.nome << '\t';
         cout<<left <<setw(6) << alunno.classe << '\n';
         finp.read( (char *) &alunno, sizeof(alunno) );
      }                             //--- lettura in ciclo

      cout<<endl;
      finp.close();
   }
   return 0;
}
```

NOTA: sia nel DAB (Diagramma A Blocchi) che nella pseudocodifica per la istruzione **APRI** vengono indicati:

- nome **logico** del file (esempio: **fout**)
- nome **fisico** del file (esempio: "alunni_OUT.**dat**")
- **modalità di apertura** (INPUT, OUTPUT oppure APPEND)
- **tipologia file** (ASCII oppure BINARY, obbligatorio solo se BINARY; se non indicato si intende file di Testo/ASCII)

per le istruzioni **LEGGI** e **SCRIVI** vengono indicati:

- **nome logico del file**, tranne nel caso di lettura/scrittura da standard input(tastiera) / std output(video)
- elenco di variabili **elementari** da leggere/scrivere (**SOLO per std input/output e per file di Testo/ASCII**)
- la **variabile strutturata** (esempio: **alunno**) da leggere con **read** o da scrivere con **write** (per file BINARY)

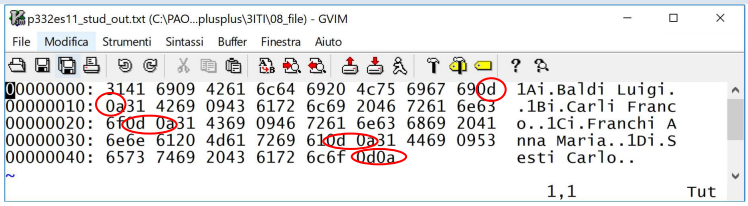
Lo stesso esercizio può essere risolto utilizzando **un file ASCII per l'output**, purchè nel file si registri prima la classe (che è una sequenza di caratteri senza spazi) e poi il nome (che invece contiene gli spazi di separazione tra cognome e nome e per i nomi composti), e si vada a capo (\n) **per ogni alunno** (file ASCII "alunni_OUT.txt").

alunni_OUT.txt

```
1Blss   Baldi Luigi
1Ai     Carli Franco
1Bnt    Franchi Anna Maria
1Clsa   Sesti Carlo
```

In questo caso ogni alunno è separato dal successivo da un ACAPO (0d0a)

alunni_OUT.txt visto con ***VIM***
utilizzando *Strumenti>Converti a esadecimale*



pseudocodifica

```
INIZIO
  APRI (finp, "alunni_INP.txt", INPUT)
  SE  finp.fail()
  ALLORA
    SCRIVI ("il file non esiste")
    RITORNO-RESTITUISCI -1
  FINE SE
  APRI (fout, "alunni_OUT.txt", OUTPUT)
  LEGGI (finp, nome)
  ESEGUI MENTRE NOT finp.eof()
       SCRIVI (nome, "classe? ")
       LEGGI (classe)
       SCRIVI (fout, classe, TAB, nome, ACAPO)
       LEGGI (finp, nome)
  RIPETI
  CHIUDI (finp, fout)
  APRI (finp, "alunni_OUT.txt", INPUT)
  LEGGI (finp, classe)
  ESEGUI MENTRE NOT finp.eof()
         LEGGI (finp, nome)
         SCRIVI (nome, TAB, classe, ACAPO)
         LEGGI (finp, classe)
  RIPETI
  CHIUDI (finp)
FINE
```

codifica (parte 1)

```
#include <iostream>
using namespace std;
#include <fstream>
#include <string.h>
#include <iomanip>

#define LN 22
#define TAB    '\t'
#define ACAPO '\n'

int main(){
char        nome[LN];
char        classe[6];
```

codifica (parte 2)

```
fstream  finp, fout;     //--- dichiarazione generica nomi file

 finp.open("alunni_INP.txt", ios::in);
 if (finp.fail()) {
    cout << "il file non esiste\n";
    return -1;
 }
 fout.open("alunni_OUT.txt", ios::out);
 cout<<"\nper ogni studente, assegnare la classe:\n";


 finp.getline(nome, LN);              //--- lettura fuori ciclo

 while ( ! finp.eof() )  {
       cout<< nome << TAB << "classe? ";
       cin >> classe;
       fout << classe << TAB << nome << ACAPO;
       finp.getline(nome, LN);        //--- lettura in ciclo
 }
 finp.close();
 fout.close();

 finp.open("alunni_OUT.txt", ios::in);
 if ( finp.fail() )
    cout<<"\n Errore nell'apertura dell'archivio\n";

 else {
    cout<<"\n Elenco studenti registrati\n";

    finp >> classe;                   //--- lettura fuori ciclo
    while ( ! finp.eof() )  {
       finp.getline(nome, LN);
       cout<<left <<setw(LN)<< nome << TAB;
       cout<<left <<setw(6) << classe << ACAPO;
       finp >> classe;                //--- lettura in ciclo
    cout<<endl;
    finp.close();
 }

 return 0;
}
```