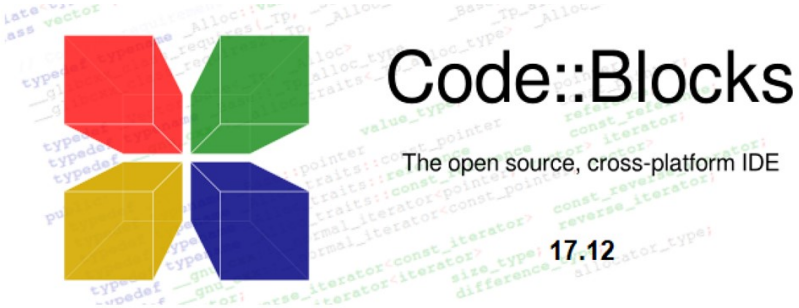
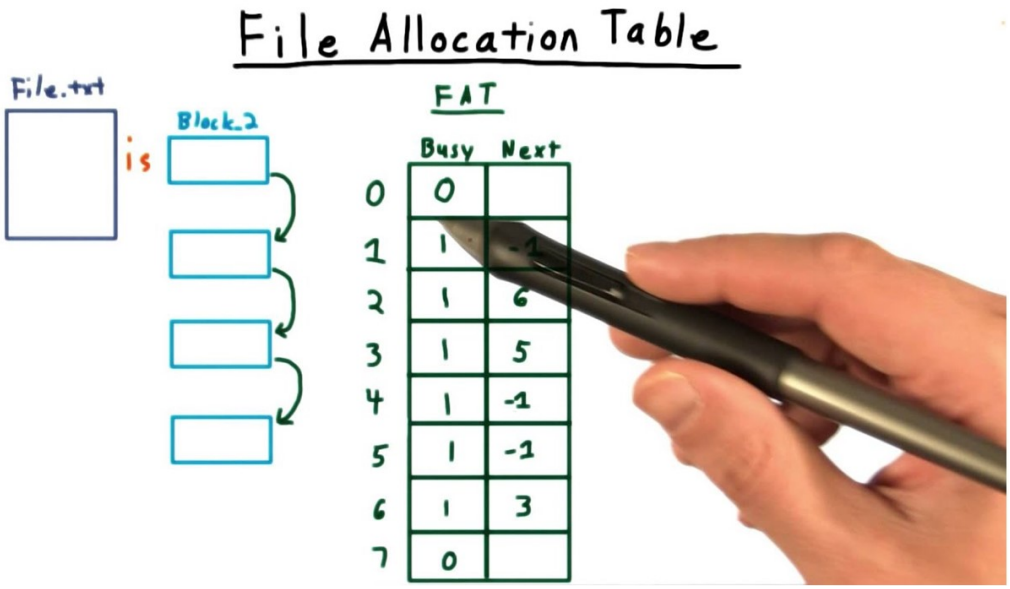
ITIS-LS "Francesco Giordani" Caserta

prof. Ennio Ranucci

a.s. 2019-2020

Semplici esercitazioni FILE C++





```
/*
ITIS-LS F.Giordani Caserta
Anno scolastico 2019/2020
Classe 4^ sez.B spec. Informatica
Data: 17/09/2019
Numero es: 1
Versione:1.0
Programmatore/i: Lezione collettiva
Sistema Operativo:Windows 10
Compilatore/Interprete: Code::Blocks Release 17.12 rev 11256
Obiettivo didattico: L'alunno e' in grado di caricare e leggere un file;
Obiettivo del programma: Caricare e leggere un file di caratteri;

TABELLA DELLE VARIABILI
IDENTIF.        I/O/LAVORO   DESC.   TIPO    DIMENSIONE
caratteriFile   I/O          FILE    1 BYTE
Car     I/O             CARATTERE   1 BYTE
Risp    INPUT           CARATTERE   1 BYTE
*/

#include<iostream>
#include<fstream>
#include <cstdlib>

using namespace std;
fstream caratteriFile; //dichiarazione variabile di tipo file (nome logico)
char car;

//Prototipi
void CaricaFile();
void LeggiFile();
```

```
int main()
{
  CaricaFile();
  LeggiFile();
  system("pause");
}


void CaricaFile()
{
  char risp;
  caratteriFile.open ("caratteri.dat", ios::out | ios::binary); //apertura in scrittura
  do
  {
    cout<<"Inserire un carattere: ";
    cin>>car;
    caratteriFile.write((char *) &car, sizeof(car)); //scrittura su file
    cout<<"Vuoi inserire altri caratteri? (s/n) ";
    cin>>risp;
  }
  while(risp=='s' || risp=='S');
  caratteriFile.close(); //chiusura buffer
}

void LeggiFile()
{
  caratteriFile.open ("caratteri.dat", ios::in | ios::binary); //apertura in lettura
  if(!caratteriFile) cout<<"Errore di apertura del file! ";
  else
  {
    cout<<"I caratteri sono: "<<endl;
    caratteriFile.read((char *) &car, sizeof(car)); //lettura da file
```

```
  while(!caratteriFile.eof())
  {
    cout<<car<<endl;
    caratteriFile.read((char *) &car, sizeof(car));
  }
 }
 caratteriFile.close(); //chiusura buffer
}
```

```
/*
ITIS-LS F.Giordani Caserta
Anno scolastico 2019/2020
Classe 4^ sez.B spec. Informatica
Data: 17/09/2019
Numero es: 2
Versione:1.0
Programmatore/i: Lezione collettiva
Sistema Operativo:Windows 10
Compilatore/Interprete: Code::Blocks Release 17.12 rev 11256
Obiettivo didattico: L'alunno e' in grado di caricare e leggere un file;
Obiettivo del programma: Caricare e leggere un file di numeri interi;

TABELLA DELLE VARIABILI
IDENTIF.        I/O/LAVORO   DESC.   TIPO    DIMENSIONE
caratteriFile   I/O          FILE    1 BYTE
num     I/O             int     2 BYTE
Risp    INPUT           CARATTERE   1 BYTE
*/

#include<iostream>
#include<fstream>
#include <cstdlib>

using namespace std;
fstream intFile; //dichiarazione variabile di tipo file - Nome logico del file
int num;

//Prototipi
void CaricaFile();
void LeggiFile();
```

```
int main()
{
  CaricaFile();
  LeggiFile();
  system("pause");
}


void CaricaFile()
{
  char risp;
  intFile.open ("num.dat", ios::out | ios::binary); //apertura in scrittura
  do
  {
    cout<<"Inserire un numero: ";
    cin>>num;
    intFile.write((char *) &num, sizeof(num));  //scrittura su file
    cout<<"Vuoi inserire altri numeri? (s/n) ";
    cin>>risp;
  }
  while(risp=='s' || risp=='S');
  intFile.close(); //chiusura buffer
}

void LeggiFile()
{
  intFile.open ("num.dat", ios::in | ios::binary); //apertura in lettura
  if(!intFile) cout<<"Errore di apertura del file! ";
  else
  {
    cout<<"I numeri sono: "<<endl;
    intFile.read((char *) &num, sizeof(num));
```

```
   while(!intFile.eof())
   {
     cout<<num<<endl;
     intFile.read((char *) &num, sizeof(num));  //lettura da file
   }
  }
  intFile.close();
}
```

```
/*
ITIS-LS F.Giordani Caserta
Anno scolastico 2019/2020
Classe 4^ sez.B spec. Informatica
Data: 17/09/2019
Numero es: 3
Versione:1.0
Programmatore/i: Lezione collettiva
Sistema Operativo:Windows 10
Compilatore/Interprete: Code::Blocks Release 17.12 rev 11256
Obiettivo didattico: L'alunno e' in grado di caricare e leggere un file;
Obiettivo del programma: Caricare,leggere ed effettuare la somma di un file di interi;
TABELLA DELLE VARIABILI
IDENTIF.        I/O/LAVORO   DESC.   TIPO     DIMENSIONE
intFile  I/O                 FILE     1 BYTE
num      I/O                 INTERO   1 BYTE
risp     INPUT               CARATTERE 1 BYTE
somma           LAVORO                INTERO   2 BYTE

*/

#include<iostream>
#include<fstream>
#include <cstdlib>

using namespace std;

//Prototipi
void CaricaFile();
void LeggiFile();
int SommaFile();
```

```
int main()
{
  CaricaFile();
  LeggiFile();
  cout<<"La somma dei numeri del file e': "<<SommaFile()<<endl;
  system("pause");
}


void CaricaFile()
{
  fstream intFile;
  int num;
  char risp;
  intFile.open ("somma.dat", ios::out | ios::binary); //apertura in scrittura
  do
  {
    cout<<"Inserire un numero: ";
    cin>>num;
    intFile.write((char *) &num, sizeof(num));  //scrittura su file
    cout<<"Vuoi inserire altri numeri? (s/n) ";
    cin>>risp;
  }
  while(risp=='s' || risp=='S');
  intFile.close();
}

void LeggiFile()
{
  fstream intFile;
  int num;
  intFile.open ("somma.dat", ios::in | ios::binary); //apertura in lettura
```

```
 if(!intFile) cout<<"Errore di apertura del file! ";
 else
 {
  cout<<"I numeri sono: "<<endl;
  intFile.read((char *) &num, sizeof(num));
  while(!intFile.eof())
  {
   cout<<num<<endl;
   intFile.read((char *) &num, sizeof(num));  //lettura da file
  }
 }
 intFile.close();
}

int SommaFile()
{
 fstream intFile;
 int num,somma=0;
 intFile.open ("somma.dat", ios::in | ios::binary);
 if(!intFile) cout<<"Errore di apertura del file! ";
 else
 {
  intFile.read((char *) &num, sizeof(num));
  while(!intFile.eof())
  {
   somma=somma+num;
   intFile.read((char *) &num, sizeof(num));
  }
 }
 return somma;
 intFile.close();
}
```

```
/*
ITIS-LS F.Giordani Caserta
Anno scolastico 2019/2020
Classe 4^ sez.B spec. Informatica
Data: 17/09/2019
Numero es: 4
Versione:1.0
Programmatore/i: Lezione collettiva
Sistema Operativo:Windows 10
Compilatore/Interprete: Code::Blocks Release 17.12 rev 11256
Obiettivo didattico: L'alunno e' in grado di caricare e leggere un file;
Obiettivo del programma: Caricare e leggere un file di caratteri contando il numero di caratteri;
TABELLA DELLE VARIABILI
IDENTIF.        I/O/LAVORO   DESC.   TIPO    DIMENSIONE
caratteriFile   I/O          FILE    1 BYTE
car     I/O             CARATTERE    1 BYTE
risp    INPUT           CARATTERE    1 BYTE
cont    LAVORO          INTERO       2 BYTE
*/

#include<iostream>
#include<fstream>
#include <cstdlib>
using namespace std;

//Prototipi
void CaricaFile();
void LeggiFile();
int ContaCarFile();

int main()
{
```

```
 CaricaFile();
 LeggiFile();
 cout<<"Il numero di caratteri e': "<<ContaCarFile()<<endl;
 system("pause");
}


void CaricaFile()
{
 fstream caratteriFile;
 char car,risp;
 caratteriFile.open ("contacar.dat", ios::out | ios::binary);
 do
 {
  cout<<"Inserire un carattere: ";
  cin>>car;
  caratteriFile.write((char *) &car, sizeof(car));
  cout<<"Vuoi inserire altri caratteri? (s/n) ";
  cin>>risp;
 }
 while(risp=='s' || risp=='S');
 caratteriFile.close();
}

void LeggiFile()
{
 fstream caratteriFile;
 char car;
 caratteriFile.open ("contacar.dat", ios::in | ios::binary);
 if(!caratteriFile) cout<<"Errore di apertura del file! ";
 else
 {
```

```
   cout<<"I caratteri sono: "<<endl;
   caratteriFile.read((char *) &car, sizeof(car));
   while(!caratteriFile.eof())
   {
     cout<<car<<endl;
     caratteriFile.read((char *) &car, sizeof(car));
   }
  }
  caratteriFile.close();
}

int ContaCarFile()
{
  fstream caratteriFile;
  char car;
  int cont=0;
  caratteriFile.open("contacar.dat", ios::in | ios::binary);
  if(!caratteriFile) cout<<"Errore di apertura del file! ";
  else
  {
   caratteriFile.read((char *) &car, sizeof(car));
   while(!caratteriFile.eof())
   {
     cont++;
     caratteriFile.read((char *) &car, sizeof(car));
   }
  }
  return cont;
  caratteriFile.close();
}
```

```
/*
ITIS-LS F.Giordani Caserta
Anno scolastico 2019/2020
Classe 4^ sez.B spec. Informatica
Data: 17/09/2019
Numero es: 5
Versione:1.0
Programmatore/i: Lezione collettiva
Sistema Operativo:Windows 10
Compilatore/Interprete: Code::Blocks Release 17.12 rev 11256
Obiettivo didattico: L'alunno e' in grado di caricare e leggere un file di records;
Obiettivo del programma: Caricare e leggere un file di records;

TABELLA DELLE VARIABILI
IDENTIF.        I/O/LAVORO   DESC.   TIPO     DIMENSIONE
articolifile    I/O                  FILE
articolirec     I/O                  ARTICOLOTYPE
risp     INPUT           CARATTERE     1 BYTE
*/

#include<iostream>
#include<fstream>
#include <cstdlib>
using namespace std;
struct articolitype
 {
   char cod[15];
   char desc[15];
   float prezzo;
 };

//Prototipi
```

```
void CaricaFile();
void LeggiFile();

int main()
{
  CaricaFile();
  LeggiFile();
  system("pause");
}

void CaricaFile()
{
  fstream articolifile;
  char risp;
  articolitype articolirec;
  articolifile.open("articoli.dat", ios::out | ios::binary);
  do
  {
    cout<<"Inserire il codice del prodotto: ";
    cin>>articolirec.cod;
    cout<<"Inserire la descrizione del prodotto: ";
    cin>>articolirec.desc;
    cout<<"Inserire il prezzo del prodotto: ";
    cin>>articolirec.prezzo;
    articolifile.write((char *) &articolirec, sizeof(articolirec));
    cout<<"Vuoi continuare? s/n ";
    cin>>risp;
  }
  while(risp=='s' || risp=='S');
  articolifile.close();
}
```

```
void LeggiFile()
{
 fstream articolifile;
 articolitype articolirec;
 articolifile.open("articoli.dat", ios::in | ios::binary);
 if(!articolifile) cout<<"Errore di apertura del file! ";
 else
 {
  cout<<"Gli articoli presenti nel file sono: "<<endl;
  cout<<"Codice \t Descrizione \t Prezzo"<<endl;
  articolifile.read((char *) &articolirec, sizeof(articolirec));
  while(!articolifile.eof())
  {
   cout<<articolirec.cod<<"\t"<<articolirec.desc<<"\t"<<articolirec.prezzo<<endl;
   articolifile.read((char *) &articolirec, sizeof(articolirec));
  }
 }
 articolifile.close();
}
```

```
/*
ITIS-LS F.Giordani Caserta
Anno scolastico 2019/2020
Classe 4^ sez.B spec. Informatica
Data: 17/09/2019
Numero es: 6
Versione:1.0
Programmatore/i: Lezione collettiva
Sistema Operativo:Windows 10
Compilatore/Interprete: Code::Blocks Release 17.12 rev 11256
Obiettivo didattico: L'alunno e' in grado di caricare e leggere un file di records ad accesso diretto;
Obiettivo del programma: Caricare e leggere un file di records con ricerca ad accesso diretto;


#include<iostream>
#include<fstream>
#include<iomanip>
#include <cstdlib>
using namespace std;
struct studenteType
 {
   int matricola;
   char nome[50];
   int eta;
   char citta[15];
 };
int K;


//Prototipi
int hash(int matrPar);
void CaricaFile();
```

```
void LeggiFile();
void RicercaDiretta(int kPar);

int main()
{
  CaricaFile();
  cout<<endl;
  LeggiFile();
  cout<<endl;
  cout<<"Inserire la matricola dello studente da ricercare: ";
  cin>>K;
  RicercaDiretta(K);
  system("pause");
}

int hash(int matrPar)
{
  return matrPar;
}
void CaricaFile()
{
  fstream studentiFile;
  studenteType studenteRec;
  char risp;int indirizzo;
  studentiFile.open("studenti.dat", ios::out | ios::binary);
  do
  {
    cout<<"Inserire la matricola: ";
    cin>>studenteRec.matricola;
    cout<<"Inserire il nome: ";
    cin>>studenteRec.nome;
    cout<<"Inserire l'eta: ";
```

```
   cin>>studenteRec.eta;
   cout<<"Inserire la citta: ";
   cin>>studenteRec.citta;
   indirizzo=hash(studenteRec.matricola);
   studentiFile.seekg((indirizzo)*sizeof(studenteRec));
   studentiFile.write((char *) &studenteRec, sizeof(studenteRec));
   cout<<"Vuoi inserire un altro record? s/n ";
   cin>>risp;
  }
  while(risp=='s' || risp=='S');
  studentiFile.close();
}

void LeggiFile()
{
  fstream studentiFile;
  studenteType studenteRec;
  studentiFile.open("studenti.dat", ios::in | ios::binary);
  if(!studentiFile) cout<<"Errore di apertura del file! ";
  else
  {
   cout<<"Gli studenti registrati nel file sono: "<<endl;

cout<<setw(16)<<"Matricola"<<setw(50)<<"Nome"<<setw(16)<<"Eta'"<<setw(15)<<"Città"<<endl;
   studentiFile.read((char *) &studenteRec, sizeof(studenteRec));
   while(!studentiFile.eof())
   {

cout<<setw(16)<<studenteRec.matricola<<setw(50)<<studenteRec.nome<<setw(16)<<studenteRec
.eta<<setw(15)<<studenteRec.citta<<endl;
     studentiFile.read((char *) &studenteRec, sizeof(studenteRec));
   }
  }
  studentiFile.close();
```

```
}
void RicercaDiretta(int KPar)
{
 fstream studentiFile; int indirizzo;
 studenteType studenteRec;
 studentiFile.open("studenti.dat", ios::in | ios::binary);
 if(!studentiFile) cout<<"Errore di apertura del file! ";
 else
 {

cout<<setw(16)<<"Matricola"<<setw(50)<<"Nome"<<setw(16)<<"Eta'"<<setw(15)<<"Città"<<endl;
   indirizzo=hash(KPar);
   studentiFile.seekg((indirizzo)*sizeof(studenteRec));
   studentiFile.read((char *) &studenteRec, sizeof(studenteRec));

cout<<setw(16)<<studenteRec.matricola<<setw(50)<<studenteRec.nome<<setw(16)<<studenteRec
.eta<<setw(15)<<studenteRec.citta<<endl;


 }
 studentiFile.close();
}
```

Anno scolastico 2019/2020

Classe 4^ sez.B spec. Informatica

Data: 17/09/2019

Numero es: 7

Versione:1.0

Programmatore/i: Lezione collettiva

Sistema Operativo:Windows 10

Compilatore/Interprete: Code::Blocks Release 17.12 rev 11256

Obiettivo didattico: L'alunno e' in grado di caricare, leggere e modificare un file di records ad accesso sequenziale;

Obiettivo del programma: Caricare, leggere e modificare un file di records con accesso sequenziale;

```
#include<iostream>
#include<fstream>
#include <cstdlib>

using namespace std;
char nomefisico[10];
//Prototipi
void CaricaFile(char nomefisicopar[10]);
void LeggiFile(char nomefisicopar[10]);
void AggiungiInCoda (char nomefisicopar[10]);
void ModificaFile(char nomefisicopar[10]);
int SommaFile(char nomefisicopar[10]);

int main()
{
 cout<<"Inserire il nome del file: ";
 cin>>nomefisico;
 fstream intFile;
 intFile.open (nomefisico, ios::in | ios::binary); //apertura in lettura
 if(!intFile)
  {
```

```
    CaricaFile(nomefisico);
    LeggiFile(nomefisico);
    AggiungiInCoda(nomefisico);
   cout<<"La somma dei numeri del file e': "<<SommaFile(nomefisico)<<endl;
   ModificaFile(nomefisico);
   LeggiFile(nomefisico);
   system("pause");
  }
 else
 {
   LeggiFile(nomefisico);
   AggiungiInCoda(nomefisico);
   LeggiFile(nomefisico);
   cout<<"La somma dei numeri del file e': "<<SommaFile(nomefisico)<<endl;
   ModificaFile(nomefisico);
   LeggiFile(nomefisico);
   system("pause");
 }
}


void CaricaFile(char nomefisicopar[10])
{
 fstream intFile;
 int num;
 char risp;
 intFile.open (nomefisico, ios::out | ios::binary); //apertura in scrittura
 do
 {
  cout<<"Inserire un numero: ";
  cin>>num;
  intFile.write((char *) &num, sizeof(num));  //scrittura su file
```

```
   cout<<"Vuoi inserire altri numeri? (s/n) ";
   cin>>risp;
  }
  while(risp=='s' || risp=='S');
  intFile.close();
}

void AggiungiInCoda (char nomefisicopar[10])
{
  fstream intFile;
  int numnew;
  intFile.open (nomefisico, ios::app | ios::binary); //apertura per scrittura e lettura
  cout<<"Inserire il nuovo numero: ";
  cin>>numnew;
  intFile.write((char *) &numnew, sizeof(numnew));  //scrittura su file
  intFile.close();
}

void ModificaFile(char nomefisicopar[10])
{
  fstream intFile;
  int num, numnew, pos;
  intFile.open (nomefisico, ios::in | ios::out | ios::binary); //apertura per scrittura e lettura
  cout<<"Inserire il nuovo numero: ";
  cin>>numnew;
  cout<<"Inserire la sua posizione: ";
  cin>>pos;
  for (int i=0; i<pos; i++) intFile.read((char *) &numnew, sizeof(numnew));  //lettura da file
  intFile.write((char *) &numnew, sizeof(numnew));  //scrittura su file
  intFile.close();
}
void LeggiFile(char nomefisicopar[10])
```

```
{
 fstream intFile;
 int num;
 intFile.open (nomefisico, ios::in | ios::binary); //apertura in lettura
 if(!intFile) cout<<"Errore di apertura del file! ";
 else
 {
  cout<<"I numeri sono: "<<endl;
  intFile.read((char *) &num, sizeof(num));
  while(!intFile.eof())
  {
   cout<<num<<endl;
   intFile.read((char *) &num, sizeof(num));  //lettura da file
  }
 }
 intFile.close();
}

int SommaFile(char nomefisicopar[10])
{
 fstream intFile;
 int num,somma=0;
 intFile.open (nomefisico, ios::in | ios::binary);
 if(!intFile) cout<<"Errore di apertura del file! ";
 else
 {
  intFile.read((char *) &num, sizeof(num));
  while(!intFile.eof())
  {
   somma=somma+num;
   intFile.read((char *) &num, sizeof(num));
  }
```

```
  }
  return somma;
  intFile.close();
}
```

```
Anno scolastico 2019/2020
Classe 4^ sez.B spec. Informatica
Data: 17/09/2019
Numero es: 8
Versione:1.0
Programmatore/i: Lezione collettiva
Sistema Operativo:Windows 10
Compilatore/Interprete: Code::Blocks Release 17.12 rev 11256
Obiettivo didattico: Cancellazione logica e ripristino
Obiettivo del programma: Cancellazione logica e ripristino
#include<iostream>
#include<fstream>
#include<iomanip>
#include<string.h>
#include<stdlib.h>
using namespace std;
int opz;
int matricolaK, etaNew;
struct studentetype
{
   int  matricola;
   char nome[15];
   char citta[15];
   int eta;
   int tagCanc;
};
fstream studenteFile;
studentetype studente, studenteApp;
char nomeFile[50]="studenti.dat";
void CaricaFile(char nomeFilePar[50]);
void LeggiFile(char nomeFilePar[50]);
void LeggiFileTutti(char nomeFilePar[50]);
```

```
void ripristinaRec(char nomeFilePar[50],int matricolaPar);
void ModificaEta(char nomeFilePar[50],int matricolaPar, int etaPar);
int  ricercaPos(char nomeFilePar[50],int matricolaPar);
void cancellaLogRec(char nomeFilePar[50],int matricolaPar);
void aggiungiInCoda(char nomeFilePar[50]);
int  menu();
int main()
{
   do
   {
     system("cls");
     opz=menu();
     switch(opz)
     {
       case 1:
       CaricaFile(nomeFile);
       break;
       case 2:
       LeggiFile(nomeFile);
       LeggiFileTutti(nomeFile);
       break;
       case 3:
       cout<<"inserisci la matricola dello studente da cancellare logicamente: ";
       cin>>matricolaK;
       cancellaLogRec(nomeFile,matricolaK);
       break;
       case 4:
       cout<<"inserisci la matricola dello studente da modificare: ";
       cin>>matricolaK;
       cout<<"inserisci il nuovo valore del campo eta': ";
       cin>>etaNew;
       ModificaEta(nomeFile,matricolaK,etaNew);
```

```
      break;
      case 5:
      aggiungiInCoda(nomeFile);
      break;
      case 6:
      cout<<"inserisci la matricola dello studente da ripristinare: ";
      cin>>matricolaK;
      ripristinaRec(nomeFile,matricolaK);
      break;
      case 7:
      cout<<"inserisci la matricola dello studente da ricercare: ";
      cin>>matricolaK;
      if (ricercaPos(nomeFile,matricolaK)==-1) cout<<"Ricerca senza successo"<<endl;
      else  cout<<ricercaPos(nomeFile,matricolaK)<<endl;
      system("Pause");
      break;
      case 8:
      cout<<"Fine";
      break;
      default:
      cout<<"hai inserito un numero non compreso tra 1 e 8";
      break;

     }
   }
   while(opz!=8);
   return 0;
}


int menu()
{
```

```
   int scelta;
   cout<<"=============| MENU'|============================"<<endl;
   cout<<"|1.carica il file                        |"<<endl;
   cout<<"|2.leggi  il file                        |"<<endl;
   cout<<"|3.cancella logicamente un elemento del file    |"<<endl;
   cout<<"|4.modifica un elemento del file             |"<<endl;
   cout<<"|5.aggiungi  un elemento del file            |"<<endl;
   cout<<"|6.ripristina un elemento del file           |"<<endl;
   cout<<"|7.ricerca la posizione di un elemento del file |"<<endl;
   cout<<"|8.esci                                  |"<<endl;
   cout<<"================================================="<<endl;
   cout <<"inserisci il numero della opzione: ";
   cin>>scelta;
   return scelta;
}


void CaricaFile(char nomeFilePar[50])
{
   char risp,risp2;
   if(studenteFile)
   {
    cout<<"file gia' creato, intendi sostituirlo?(s/n)";
    cin>>risp;
    if(risp =='s' || risp=='S')
     {
      studenteFile.open(nomeFilePar, ios::out | ios::binary);
     do
      {
       cout<<"inserisci la matricola: ";
       cin>>studente.matricola;
       cout<<"inserisci il nome: ";
```

```
    cin>>studente.nome;
    cout<<"inserisci la citta: ";
    cin>>studente.citta;
    cout<<"inserisci l'eta: ";
    cin>>studente.eta;
    studente.tagCanc=1;
    studenteFile.write((char*) &studente ,sizeof(studente));
    cout<<"Vuoi inserire altri studenti  ? (s/n) ";
    cin>>risp2;

   }
   while(risp2=='s' || risp2=='S');
   studenteFile.close();
  }
}
else
  {
   studenteFile.open(nomeFilePar, ios::out | ios::binary);
  do
   {
    cout<<"inserisci la matricola: ";
    cin>>studente.matricola;
    cout<<"inserisci il nome: ";
    cin>>studente.nome;
    cout<<"inserisci la citta: ";
    cin>>studente.citta;
    cout<<"inserisci l'eta: ";
    cin>>studente.eta;
    studente.tagCanc=1;
    studenteFile.write((char*) &studente ,sizeof(studente));
    cout<<"Vuoi inserire altri studenti  ? (s/n) ";
    cin>>risp2;
```

```
        }
        while(risp2=='s' || risp2=='S');
        studenteFile.close();
       };
 }


void LeggiFile(char nomeFilePar[50])
{
    studenteFile.open(nomeFilePar, ios::in | ios::binary);

cout<<setw(16)<<"matricola"<<setw(16)<<"nome"<<setw(16)<<"citta'"<<setw(16)<<"eta'"<<endl;
   if (!studenteFile)cout<<"Errore di apertura del file! ";
   else
   {

      studenteFile.read((char *) &studente, sizeof(studente)); //lettura da file
       while(!studenteFile.eof())
       { if(studente.tagCanc==1)
         {

cout<<setw(16)<<studente.matricola<<setw(16)<<studente.nome<<setw(16)<<studente.citta<<set
w(16)<<studente.eta<<endl;
         }
         studenteFile.read((char *) &studente, sizeof(studente));


       }
   }
  studenteFile.close();
  system("Pause");
}

void LeggiFileTutti(char nomeFilePar[50])
```

```
{
    studenteFile.open(nomeFilePar, ios::in | ios::binary);

cout<<setw(16)<<"matricola"<<setw(16)<<"nome"<<setw(16)<<"citta'"<<setw(16)<<"eta'"<<endl;
   if (!studenteFile)cout<<"Errore di apertura del file! ";
   else
   {

      studenteFile.read((char *) &studente, sizeof(studente)); //lettura da file
       while(!studenteFile.eof())
        {


cout<<setw(16)<<studente.matricola<<setw(16)<<studente.nome<<setw(16)<<studente.citta<<
          setw(16)<<studente.eta<<setw(16)<<studente.tagCanc<<endl;
          studenteFile.read((char *) &studente, sizeof(studente));
        }



   }
  studenteFile.close();
  system("Pause");
}

void ModificaEta(char nomeFilePar[50],int matricolaPar, int etaPar)
{
int pos,i;
   pos=ricercaPos(nomeFile,matricolaPar);
   if (pos==-1) cout<<"Ricerca senza successo, modifica impossibile";
   else
     { studenteFile.open(nomeFilePar, ios::in | ios::out | ios::binary);
       if (!studenteFile)cout<<"Errore di apertura del file! ";
       else
```

```
        {
         for ( i=0;i<pos+1;i++) studenteFile.read((char *) &studente, sizeof(studente));
         studente.eta=etaPar;
         studenteApp=studente;
         studenteFile.close();
         studenteFile.open(nomeFilePar, ios::in | ios::out | ios::binary);
         for ( i=0;i<pos;i++) studenteFile.read((char *) &studente, sizeof(studente));
         studenteFile.write((char *) &studenteApp, sizeof(studenteApp));
         cout<<"Modifica del campo eta'  effettuata con successo "<<endl;
         system("Pause");
        }
     }
   studenteFile.close();
}

void aggiungiInCoda(char nomeFilePar[50])
{
   fstream studenteFile;
   char risp;
   studentetype studente;

        studenteFile.open(nomeFilePar, ios::app | ios::binary);
        cout<<"inserisci la matricola: ";
        cin>>studente.matricola;
        cout<<"inserisci il nome: ";
        cin>>studente.nome;
        cout<<"inserisci la citta: ";
        cin>>studente.citta;
        cout<<"inserisci l'eta: ";
        cin>>studente.eta;
        studente.tagCanc=1;
        studenteFile.write((char*) &studente ,sizeof(studente));
```

```
        studenteFile.close();
}




int  ricercaPos(char nomeFilePar[50],int matricolaPar)
{
   int pos=0;
   studenteFile.open(nomeFilePar, ios::in | ios::binary);
   if (!studenteFile)cout<<"Errore di apertura del file! ";
   else
   {
      studenteFile.read((char *) &studente, sizeof(studente));
      while(studente.matricola!=matricolaPar && !studenteFile.eof())
      {
         pos++;
         studenteFile.read((char *) &studente, sizeof(studente));
      }

   }
  studenteFile.close();
  if (studente.matricola==matricolaPar)  return pos; else return -1;

 }
void cancellaLogRec(char nomeFilePar[50],int matricolaPar)
{
   int pos,i;
   pos=ricercaPos(nomeFile,matricolaPar);
   if (pos==-1) cout<<"Ricerca senza successo, cancellazione logica impossibile";
   else
     { studenteFile.open(nomeFilePar, ios::in | ios::out | ios::binary);
```

```
        if (!studenteFile)cout<<"Errore di apertura del file! ";
        else
         {
          for ( i=0;i<pos+1;i++) studenteFile.read((char *) &studente, sizeof(studente));
          studente.tagCanc=0;
          studenteApp=studente;
          studenteFile.close();
          studenteFile.open(nomeFilePar, ios::in | ios::out | ios::binary);
          for ( i=0;i<pos;i++) studenteFile.read((char *) &studente, sizeof(studente));
          studenteFile.write((char *) &studenteApp, sizeof(studenteApp));
          cout<<"Cancellazione logica effettuata con successo "<<endl;
          system("Pause");
        }
     }
   studenteFile.close();

}
void ripristinaRec(char nomeFilePar[50],int matricolaPar)
{
  int pos,i;
    pos=ricercaPos(nomeFile,matricolaPar);
    if (pos==-1) cout<<"Ricerca senza successo, cancellazione logica impossibile";
    else
      { studenteFile.open(nomeFilePar, ios::in | ios::out | ios::binary);
        if (!studenteFile)cout<<"Errore di apertura del file! ";
        else
         {
          for ( i=0;i<pos+1;i++) studenteFile.read((char *) &studente, sizeof(studente));
          studente.tagCanc=1;
          studenteApp=studente;
          studenteFile.close();
          studenteFile.open(nomeFilePar, ios::in | ios::out | ios::binary);
```

```
        for ( i=0;i<pos;i++) studenteFile.read((char *) &studente, sizeof(studente));
        studenteFile.write((char *) &studenteApp, sizeof(studenteApp));
        cout<<"Ripristino effettuato con successo "<<endl;
        system("Pause");
      }
    }
  studenteFile.close();

}
```

Anno scolastico 2020/2021

Classe 4^ sez.C spec. Informatica

Data: 29/03/2021

Numero es: 9

Versione:1.0

Programmatore/i: Lezione collettiva

Sistema Operativo:Windows 10

Compilatore/Interprete: Code::Blocks Release 17.12 rev 11256

Obiettivo didattico: Accesso diretto Cancellazione logica e ripristino

Obiettivo del programma: Accesso diretto Cancellazione logica e ripristino

```
#include<iostream>
#include<fstream>
#include<iomanip>
#include<string.h>
#include<stdlib.h>
using namespace std;
int opz;
int matricolaK;
struct studentetype
{
 int matricola;
 char nome[15];
 int tagCanc;
};
fstream studenteFile;
studentetype studente;
char nomeFile[50]="studenti.dat";
void creaFile(char nomeFilePar[50]);
void leggiFile(char nomeFilePar[50]);
void ripristina(char nomeFilePar[50],int matricolaPar);
int ricercaPos(char nomeFilePar[50],int matricolaPar);
```

```
void cancellaLog(char nomeFilePar[50],int matricolaPar);
void inserisciNelFile(char nomeFilePar[50]);
void creaInserisci(char nomeFilePar[50]);
void leggiFileAncheCancellatiLog(char nomeFilePar[50]);
void cancellaFis(char nomeFilePar[50]);
int menu();


int main()
{
   do
   {
     system("cls");
     opz=menu();
     switch(opz)
      {
       case 1:
         creaFile(nomeFile);
       break;

       case 2:
         inserisciNelFile(nomeFile);
       break;

       case 3:
         leggiFile(nomeFile);
       break;

       case 4:
         cout<<"inserisci la matricola dello studente da cancellare logicamente: ";
         cin>>matricolaK;
         cancellaLog(nomeFile,matricolaK);
```

```
break;

case 5:
cout<<"inserisci la matricola dello studente da ripristinare: ";
cin>>matricolaK;
ripristina(nomeFile,matricolaK);
break;

case 6:
cout<<"inserisci la matricola dello studente da ricercare: ";
cin>>matricolaK;
if (ricercaPos(nomeFile,matricolaK)==-1) cout<<"Ricerca senza successo"<<endl;
 else cout<<ricercaPos(nomeFile,matricolaK)<<endl;
system("Pause");
break;

case 7:
  leggiFileAncheCancellatiLog(nomeFile);
break;

case 8:
  cancellaFis(nomeFile);
break;

case 9:
cout<<"  "<<endl;
cout<<"----Fine----";
cout<<"  "<<endl;
break;
default:
cout<<"hai inserito un numero non compreso tra 1 e 9";
break;
```

```
      }
  }
 while(opz!=9);
 return 0;
}
int menu()
{
 int scelta;
 cout<<"| MENU'"<<endl;
 cout<<"|1.creazione file "<<endl;
 cout<<"|2.inserimento file "<<endl;
 cout<<"|3.lettura file "<<endl;
 cout<<"|4.cancellazione logica di un elemento del file "<<endl;
 cout<<"|5.ripristino di un elemento del file cancellato logicamente "<<endl;
 cout<<"|6.ricerca la posizione di un elemento del file "<<endl;
 cout<<"|7.Lettura del file (anche i record cancellati logicamente) "<<endl;
 cout<<"|8.cancellazione fisica "<<endl;
 cout<<"|9.esci "<<endl;
 cout<<"  "<<endl;
 cout <<"inserisci il numero della opzione: ";
 cin>>scelta;
 return scelta;
}
void creaFile(char nomeFilePar[50])
{
 char risp,risp2;
 studenteFile.open(nomeFilePar, ios::in| ios::binary);
 if(studenteFile)
 {
   studenteFile.close();
   cout<<"Attenzione il file risulta gia' creato, una nuova creazione cancella i dati precedenti.
Procedo?(s/n)";
   cin>>risp;
```

```
   if(risp =='s' || risp=='S') creaInserisci(nomeFilePar);
   else
      {
         cout<<"Procedura annullata"<<endl;
         system("Pause");
      }
 }
 else creaInserisci(nomeFilePar);

}
void leggiFile(char nomeFilePar[50])
{
 studenteFile.open(nomeFilePar, ios::in | ios::binary);
 cout<<setw(16)<<"matricola"<<setw(16)<<"nome"<<endl;
 if (!studenteFile)cout<<"Errore di apertura del file! ";
   else
      {
        studenteFile.read((char *) &studente, sizeof(studente)); //lettura da file
        while(!studenteFile.eof())
         {
           if(studente.tagCanc==1)
            {
               cout<<setw(16)<<studente.matricola<<setw(16)<<studente.nome<<endl;
            }
           studenteFile.read((char *) &studente, sizeof(studente));
         }
      }
  studenteFile.close();
  system("Pause");
}

void inserisciNelFile(char nomeFilePar[50])
```

```
{
 char risp;
 studenteFile.open(nomeFilePar, ios::out |ios::in| ios::binary);
 if (!studenteFile)cout<<"Errore di apertura del file! ";
  else
     {
     do
        {
           cout<<"inserisci la matricola: ";
           cin>>studente.matricola;
           cout<<"inserisci il nome: ";
           cin>>studente.nome;
           studente.tagCanc=1;
           studenteFile.seekg((studente.matricola)*sizeof(studente));
           studenteFile.write((char*) &studente ,sizeof(studente));
           cout<<"Vuoi inserire altri studenti ? (s/n) ";
           cin>>risp;
        }
        while(risp=='s' || risp=='S');
        studenteFile.close();
     };
}



int ricercaPos(char nomeFilePar[50],int matricolaPar)
{
 int pos=0;
 studenteFile.open(nomeFilePar, ios::in | ios::binary);
 if (!studenteFile)cout<<"Errore di apertura del file! ";
 else
  {
```

```
      studenteFile.read((char *) &studente, sizeof(studente));
      while(studente.matricola!=matricolaPar && !studenteFile.eof())
      {
         pos++;
         studenteFile.read((char *) &studente, sizeof(studente));
      }
   }
 studenteFile.close();
 if (studente.matricola==matricolaPar) return pos; else return -1;
}


void cancellaLog(char nomeFilePar[50],int matricolaPar)
{
 studentetype studenteApp;
 int pos,i;
 pos=ricercaPos(nomeFile,matricolaPar);
 if (pos==-1)
     {
      cout<<"Ricerca senza successo, cancellazione logica impossibile"<<endl;
      system("Pause");
     }
   else
   { studenteFile.open(nomeFilePar, ios::in | ios::out | ios::binary);
   if (!studenteFile)cout<<"Errore di apertura del file! ";
   else
      {
      for ( i=0;i<pos+1;i++) studenteFile.read((char *) &studente, sizeof(studente));
      studente.tagCanc=0;
      studenteApp=studente;
      studenteFile.close();
      studenteFile.open(nomeFilePar, ios::in | ios::out | ios::binary);
```

```
      for ( i=0;i<pos;i++) studenteFile.read((char *) &studente, sizeof(studente));
      studenteFile.write((char *) &studenteApp, sizeof(studenteApp));
      cout<<"Cancellazione logica effettuata con successo "<<endl;
      system("Pause");
      }
   }
 studenteFile.close();
}


void ripristina(char nomeFilePar[50],int matricolaPar)
{
 studentetype studenteApp;
 int pos,i;
 pos=ricercaPos(nomeFile,matricolaPar);
   if (pos==-1)
    {
      cout<<"Ricerca senza successo, cancellazione logica impossibile"<<endl;
      system("Pause");
    }
   else
      { studenteFile.open(nomeFilePar, ios::in | ios::out | ios::binary);
      if (!studenteFile)cout<<"Errore di apertura del file! ";
      else
         {
            for ( i=0;i<pos+1;i++) studenteFile.read((char *) &studente, sizeof(studente));
            studente.tagCanc=1;
            studenteApp=studente;
            studenteFile.close();
            studenteFile.open(nomeFilePar, ios::in | ios::out | ios::binary);
            for ( i=0;i<pos;i++) studenteFile.read((char *) &studente, sizeof(studente));
            studenteFile.write((char *) &studenteApp, sizeof(studenteApp));
```

```
            cout<<"Ripristino effettuato con successo "<<endl;
            system("Pause");
          }
       }
 studenteFile.close();
}
void creaInserisci(char nomeFilePar[50])
 {
   char risp;
   studenteFile.open(nomeFilePar, ios::out | ios::binary);
     do
      {
        cout<<"inserisci la matricola: ";
        cin>>studente.matricola;
        cout<<"inserisci il nome: ";
        cin>>studente.nome;
        studente.tagCanc=1;
        studenteFile.seekg((studente.matricola)*sizeof(studente));
        studenteFile.write((char*) &studente ,sizeof(studente));
        cout<<"Vuoi inserire altri studenti ? (s/n) ";
        cin>>risp;
      }
     while(risp=='s' || risp=='S');
   studenteFile.close();
 }
void leggiFileAncheCancellatiLog(char nomeFilePar[50])
{
 studenteFile.open(nomeFilePar, ios::in | ios::binary);
 cout<<setw(16)<<"matricola"<<setw(16)<<"nome"<<endl;
 if (!studenteFile)cout<<"Errore di apertura del file! ";
   else
      {
```

```
        studenteFile.read((char *) &studente, sizeof(studente)); //lettura da file
        while(!studenteFile.eof())
         {

cout<<setw(16)<<studente.matricola<<setw(16)<<studente.nome<<setw(16)<<studente.tagCanc<
<endl;
           studenteFile.read((char *) &studente, sizeof(studente));
         }
      }
  studenteFile.close();
  system("Pause");
}
void cancellaFis(char nomeFilePar[50])
{
 fstream studenteApp;
 studenteFile.open(nomeFilePar, ios::in | ios::binary);
 if (!studenteFile)cout<<"Errore di apertura del file! ";
   else
      {
        studenteApp.open("app", ios::out | ios::binary);
        studenteFile.read((char *) &studente, sizeof(studente)); //lettura da file
        while(!studenteFile.eof())
         {
           if(studente.tagCanc==1)
            {
              studenteApp.seekg((studente.matricola)*sizeof(studente));
              studenteApp.write((char *) &studente, sizeof(studente));
            }
           studenteFile.read((char *) &studente, sizeof(studente));
         }
      }
  studenteFile.close();
  studenteApp.close();
```

```
  remove(nomeFilePar);
  rename("app", nomeFilePar);
}
```