```
// Soluzione_Simulazione_Compito_Fatture.cpp: creazione archivio Fatture
/*
Si vuole gestire un elenco le cui voci che memorizzano i seguenti dati:

Fattura: CodFattura, CodCliente, Data, Importo, Pagato.
L'attributo Pagato avrà valori possibili: SI/NO.
Formato  Date: aaaa, mm. gg es.: "2016-05-24"

Scrivere un programma in C++ che definisca la struttura "voce",
composta dagli attributi elencati sopra.
Si leggano poi da  input standard n voci, con n definito dall'utente,
e si inseriscano in un file.
L'utente inserisca poi il valore di un attributo del record da cercare
e si stampi il contenuto del record, se il record è presente.
 */
 /*
 Svolgere i seguenti compiti:
1) Creazione del programma, comprendente:
   la definizione delle strutture dati e le variabili necessarie
   ad archiviare i dati in un file, le librerie necessarie,
   il main e la chiamata, nel main, alle funzioni per la lettura dei dati
   e per la ricerca del record.
   Solo i Prototipi delle funzioni!!!
   Vedi diagramma a blocchi del programma principale. (P.ti 2.5)

2.1) Implementare la funzione per la la lettura dei dati che deve
     terminare quando l'utente lo desidera.
     La funzione deve acquisire i dati da input standard
     e li salva in un file. (P.ti 3.5)

2.2) Implementare la funzione per la ricerca:
   la funzione cerca il record e se lo trova stampa il contenuto. (P.ti 4)

Attenzione: Lo svolgimento del compito senza l'uso di funzioni comporta
           la perdita di due punti sul punteggio complessivo.
*/
/* comandi da console per compilare ed eseguire il codice:
ls
cd 4CodiciSorgenti_FlussiDati/Soluzione_Simulazione_Compito_FATTURE_C++
g++ Soluzione_Simulazione_Compito_Fatture.cpp -o Archivio_Fatture
./Archivio_Fatture
...
Inserisci un nome (* = fine): studenti
*/
```

```cpp
#include <iostream>
#include <fstream>
#include <iomanip>
#include <string>
//#include <cstring>
//#define NOMEARCH = "fatture.dat";// mi darebbe errore.
using namespace std;
const string NOMEARCH = "fatture.dat";

struct Fattura{
    char CodFattura[8], CodCliente[8],
    Data[11];
   float Importo;
   int Pagato; // SI=1; NO = 2; oppure sospesa = 3;
};

Fattura fattura;
//prototipi Funzioni
int Leggi_Dati();
int Stampa_Report();// non richiesta nel compito ma la aggiungo per completezza
void Ricerca();
int main()
{
    Leggi_Dati();
    Stampa_Report();
    Ricerca();
    return 0;
}
int Leggi_Dati()
{   string Stop= "stop";
    ofstream fout;
// aggiunge altri dati
//  fout.open(NOMEARCH,  ios::out | ios::app | ios::binary); // da errore per sintassi errata di NOMEARCH
//  fout.open("anagrafe.dat",  ios::out | ios::app | ios::binary); // Funziona anche se passo il nome file come "anagrafe.dat".
                                        //apre il file binario in append se eesiste altrimenti lo crea!
    fout.open(NOMEARCH.c_str(),  ios::out | ios::app | ios::binary); // apre il file binario in append se eesiste altrimenti lo crea!
    if (!fout) {
        cout << "Errore nell'apertura dell'archivio" << endl;
    } else {
        cout << "Codice Fattura (stop per terminare): ";
        cin >> fattura.CodFattura;
        while (fattura.CodFattura != Stop) {
```

```
87              cin.ignore(80, '\n');
88              cout << "Codice Cliente: ";
89              cin.getline(fattura.CodCliente, 8);
90              cout << "\nInserire Data (aaaa, mm. gg es.: '2022-04-08': ";
91              //getline(cin, fattura.Data); // Non questa sintassi che richiede un tipo string
92              cin.getline(fattura.Data, 11);
93
94              cout << "\nInserire Importo: ";
95              cin >> fattura.Importo;
96
97              /*cout << "\nInserire Pagato (si/no): ";
98              cin >> fattura.Pagato;*/
99              do {
100                     cout<<"\nInserire Pagato (SI=1; NO = 2; oppure sospesa = 3): ";
101                     cin>>fattura.Pagato;
102                     cout<<"\nPagato inserito = "<< fattura.Pagato<<endl;
103                 } while(!(fattura.Pagato==1 || fattura.Pagato ==2 ||  fattura.Pagato ==3));
104
105
106             fout.write((char *) &fattura, sizeof(fattura));
107             cout << "Inserisci Codice Fattura (stop per terminare): ";
108             cin >> fattura.CodFattura;
109         }
110         fout.close();
111     }
112     return 0;
113 }
114
115
116 int Stampa_Report()
117 {
118 // visualizza i dipendenti registrati
119     ifstream fin;
120     fin.open(NOMEARCH.c_str(), ios::in | ios::binary);
121     if (!fin) {
122         cout << "Errore nell'apertura dell'archivio" << endl;
123     } else {
124         cout << "Elenco fatture registrate:" << endl;
125         while (fin.read((char *) &fattura, sizeof(fattura))) {
126             cout << fattura.CodFattura << ": "
127                  << fattura.CodCliente << '\t'
128                  << fattura.Data << '\t'
129                  << fattura.Importo << " euro"<< '\t';
130               //  << fattura.Pagato << endl; // devo decodificare!!!
```

```
131                  switch (fattura.Pagato){
132                    case 1: cout<< setw(8)<<"Si"<<endl;
133                      break;
134                    case 2: cout<< setw(8)<<"NO"<<endl;
135                      break;
136                    case 3: cout<< setw(8)<<"Sospesa"<<endl;
137                     break;
138                  } // fine switch
139          }
140          fin.close();
141      }
142      return 0;
143  }
144
145  void Ricerca() { // Funzione per cercare una voce
146
147    string nomeDaCercare;
148    Fattura fatturaInLettura;
149    ifstream ilMioFile;
150
151    cout << "Inserisci il campo da cercare: ";
152
153    cin.ignore(80, '\n');
154    getline(cin, nomeDaCercare);
155
156    ilMioFile.open(NOMEARCH.c_str(), ios_base::binary); // Apro il file
157
158     string attributo;
159    cout << endl;
160    while (ilMioFile.read((char *) &fatturaInLettura, sizeof(fatturaInLettura))) {
161     attributo = (string)fatturaInLettura.CodCliente;// converto il tipo char in string
162    if (attributo.find(nomeDaCercare) != string::npos) { // Controllo se nella stringa codCliente è presente il
       campo di ricerca
163
164      cout << "\nTROVATO\n" << "\nCodice Cliente: " << fatturaInLettura.CodCliente << "\nCodice Fattura: " <<
         fatturaInLettura.CodFattura << "\nData: " << fatturaInLettura.Data << "\nImporto: " <<
         fatturaInLettura.Importo << "\nPagato: " << fatturaInLettura.Pagato << endl;
165
166    }
167      attributo = (string)fatturaInLettura.CodFattura;// converto il tipo char in string
168    if (attributo.find(nomeDaCercare) != string::npos) { // Stessa cosa di prima solo con codFattura
169
170      cout << "\nTROVATO\n" << "\nCodice Cliente: " << fatturaInLettura.CodCliente << "\nCodice Fattura: " <<
         fatturaInLettura.CodFattura << "\nData: " << fatturaInLettura.Data << "\nImporto: " <<
```

```
      fatturaInLettura.Importo << "\nPagato: "
171   << fatturaInLettura.Pagato << endl;
172
173   }
174
175    attributo = (string)fatturaInLettura.Data;// converto il tipo char in string
176 if (attributo.find(nomeDaCercare) != string::npos) {
177
178    cout << "\nTROVATO\n" << "\nCodice Cliente: " << fatturaInLettura.CodCliente << "\nCodice Fattura: " <<
       fatturaInLettura.CodFattura << "\nData: " << fatturaInLettura.Data << "\nImporto: " <<
       fatturaInLettura.Importo << "\nPagato: " << fatturaInLettura.Pagato << endl;
179
180   }
181
182   }
183
184   cout << endl;
185
186   ilMioFile.close(); // Chiudo ed esco
187
188 }
189
190
```