PROGETTO REALTA’

LABORATORIO DI INFORMATICA

Esercitazione:

Gestione Anagrafiche

Gruppo: Arancione

Componenti del gruppo : Braian Pilosio, Francesco Pezzutti, Abdullah Tariq.

Classe: 3 BIA

Testo della traccia:

Gestione Anagrafiche

1. Gestire l'anagrafica di una persona con una struct
Inserite in una struct almeno i seguenti campi:

- IdPersona (un progressivo numerico)
- Cognome
- Nome
- Indirizzo
- Località
- Sigla provincia
- Telefono

2. Gestire il caricamento da file in memoria di un elenco di anagrafiche. Le informazioni relative ad una anagrafica contenute nel file, corrispondono ai campi della struct.
Per generare il file di anagrafiche usate Excel (esportare il foglio Excel di anagrafiche in formato .csv)

3. . L'anagrafica persone deve essere sempre ordinata in modo crescente rispetto al campo IdPersona

4. Gestire l'inserimento da programma c++ di nuove anagrafiche (l'ordine deve essere mantenuto)

5. Gestione della ricerca per IdPersona

6. Gestire la cancellazione di una anagrafica (l'ordine deve essere mantenuto)

7. Visualizzazione del contenuto dell'anagrafica

8. Gestione del salvataggio su file dell'anagrafica

9. Realizzazione di un programma che attraverso un menù richiami ognuna di queste gestioni

Analisi del problema:

Il problema riguarda la gestione delle anagrafiche riguardanti una persona ovvero:
Il numero identificativo della persona (idPersona), il suo cognome, il suo nome, il suo indirizzo, la sua località, la sua provincia e il suo numero di telefono.
Per gestire questo problema abbiamo fatto uso di una struct dove tutti i suoi campi sono quelli richiesti dall'inserimento dei dati anagrafici.Poi per gestire più anagrafiche di più persone abbiamo usato un array/vector di struct.

1. Gestire l'anagrafica di una persona con una struct

Per gestire i dati anagrafici di ogni persona abbiamo creato un array di struct.
In questa struct verranno inseriti i dati riguardanti una persona.
Poi in seguito verrà creato un array di tipo "anagrafiche" che è appunto il nome della struct, che dopo che essa viene creata si crea un tipo di dato che appunto prende il nome di "anagrafiche" ovvero quella della struct

```
using namespace std;
//dichiarazione
    struct anagrafiche{
        int idPersona;
        string Cognome;
        string Nome;
        string Indirizzo;
        string Localita;
        string Sigla_provincia;
        string Telefono;
    };
```

2. Gestire il caricamento da file:

```
1;Corona;Matteo;via A;citt… A;PN;1234;
2;Rossi;Marco;via B;citt… B;PN;1234;
3;Verdi;Sara;via C;citt… C;PN;1234;
4;Blu;Anna;via D;citt… D;PN;1234;
5;Gialli;Giulia;via E;citt… E;PN;1234;
6;Marroni;Pietro;via F;citt… F;PN;1234;
7;Azzurri;Piero;via G;citt… G;PN;1234;
8;Bianchi;Mauro;via H;citt… H;PN;1234;
```

Come prima cosa bisognerà creare un file su Excel che rappresenti per ogni colonna ogni elemento della struct.In seguito il file verrà salvato con estensione ".csv" che sta per "comma separated values".
E successivamente che lo avremmo salvato in formato ".txt" noi tramite il nostro programma potremmo leggere da questo file i dati.
Nel programma dovremmo sia leggere i dati sia salvarli appunto nella nostra struct nei diversi campi

3. Gestire l’ordinamento crescente del campo IdPersona:

A questo fine abbiamo creato una funzione che utilizzi il bubble sort e che ha come parametri l’ array di struct e la sua dimensione massima.
Nonostante la funzione sia di tipo void l’array sarà ordinato in quanto gli array sono passati per riferimento in automatico.

```
void bubble_Sort(anagrafiche arr[], int dim){

    for(int j=0;j<=dim-1;j++)
        for(int i=0;i<=dim-1;i++){
            if(arr[i].idPersona > arr[i+1].idPersona)
                swap(arr[i].idPersona, arr[i+1].idPersona);
        }//for
}
```

4. Gestire l'inserimento di nuove anagrafiche

Per la gestione dell'inserimento di nuove anagrafiche si crea la funzione con il nome di "nuov_anagraf" con i parametri l'array di struct,la dimensione di base, che all'inizio sarà 0 e poi in base a quante anagrafiche verranno inserite la dimensione aumenterà (la dimensione massima all'inizio è impostata a 100 ma può essere cambiata) e infine la dimensione massima.
Per l'inserimento c'è un "cout" che ci fa visualizzare il quale campo noi metteremo i dati e di seguito un "cin" che appunto mette i dati nell'array di struct nella posizione usata, che all'inizio è 0.
E questo si ripete per tutti gli altri campi della struct.

```
void nuov_anagraf(anagrafiche arr[], int dim_usata, int dim_max){

        cout<<"inserire l'Id Persona:"<<endl;
    cin>>arr[dim_usata].idPersona;
        cout<<"inserire il cognome:"<<endl;
        cin>>arr[dim_usata].Cognome;
        cout<<"inserire il nome:"<<endl;
        cin>>arr[dim_usata].Nome;
        cout<<"inserire l' indirizzo:"<<endl;
        cin>>arr[dim_usata].Indirizzo;
        cout<<"inserire la localita':"<<endl;
        cin>>arr[dim_usata].Localita;
        cout<<"inserire la sigla della provincia':"<<endl;
        cin>>arr[dim_usata].Sigla_provincia;
        cout<<"inserire il numero di telefono':"<<endl;
        cin>>arr[dim_usata].Telefono;
        dim_usata++;
}
```

5. Gestione della ricerca per IdPesona:

Per gestire la ricerca di ogni persona abbiamo utilizzato un algoritmo di ricerca dicotomica che utilizza il campo IdPersona che è di tipo int.

```
int ricerca_bin(anagrafiche arr[], int cercato,int dim){

    int i = 0, j = dim-1, m, pos = -1;

    do {
    m = (i + j)/2;
    if(arr[m].idPersona == cercato)
        pos = m;
    else if (arr[m].idPersona < cercato)
        i = m + 1;
    else
        j = m - 1;
    } while(i <= j && pos == -1);

    return pos;
}
```

6. Cancellazione di un anagrafica:

Per cancellare il contenuto di un anagrafica abbiamo assegnato all'indice i l'ID dell'anagrafica da eliminare e questo indice andrà avanti scambiandosi con l'anagrafica successiva finché non raggiungerà la lunghezza del vettore-1 e quando arriverà lì con l'uso della funzione pop_back verrà "buttato fuori" dal vettore.

```
void deleteAna(std::vector <Anagrafica> &vectAna, int idxAna){
int len;
  len= vectAna.size();
  for(int i=idxAna; i <len-1;i++ ){
    std::swap(vectAna[i], vectAna[i+1]);
  } //for
  vectAna.pop_back();
}//deleteAna
```

7. Visualizzazione dell'anagrafica

Per la visualizzazione dell'anagrafica si crea una funzione con nome “showAnagrafica ” dove come parametri vengono passati il vettore di struct e l’ID dell’anagrafica.
E questa funzione consiste semplicemente nel fare il cout di ogni campo dell’anagrafica.

```
void showAnagrafica(std::vector<Anagrafica> &vectAna,int idAna){

    std::cout << "|" << vectAna[idAna].idPersona << "\t";
    std::cout << "|" << vectAna[idAna].cognome << "\t\t";
    std::cout << "|" << vectAna[idAna].nome << "\t";
    std::cout << "|" << vectAna[idAna].localita << "\t\t";
    std::cout << "|" << vectAna[idAna].sigla_provincia << "\t";
    std::cout << "|" << vectAna[idAna].telefono << "\t" << std::endl;

}//showAnagrafica
```

Vi è anche un’altra funzione di visualizzazione che a differenza della prima, questa fa la visualizzazione di tutte le anagrafiche presenti nel vettore di struct delle anagrafiche tramite un ciclo for.

```
void showAnagrafiche(std::vector <Anagrafica> &vectAna){
std::cout << "\n|ID PERSONA\t|COGNOME\t|NOME\t\t|INDIRIZZO\t|LOCALITA\t|PROV\t|TELEFONO\n" <<
          "-------------------------------------------------------------------------------------
for (int i=0;i<vectAna.size(); i++){
        std::cout << "|" << vectAna[i].idPersona << "\t";
        std::cout << "|" << vectAna[i].cognome << "\t\t";
        std::cout << "|" << vectAna[i].nome << "\t";
        std::cout << "|" << vectAna[i].localita << "\t\t";
        std::cout << "|" << vectAna[i].sigla_provincia << "\t";
        std::cout << "|" << vectAna[i].telefono << "\t" << std::endl;
}//for
}
```

8. Gestione del salvataggio su file dell'anagrafica

```
void saveAnagrafiche(std::ofstream &fl_DatiOut, std::vector <Anagrafica> &vectAna){

//intestazione
fl_DatiOut<<"idPersona;";
  fl_DatiOut<<"cognome;";
  fl_DatiOut<<"nome;";
  fl_DatiOut<<"indirizzo;";
  fl_DatiOut<<"localita;";
  fl_DatiOut<<"sigla_provincia;";
  fl_DatiOut<<"telefono"<<std::endl;

for (int i= 0;i< vectAna.size(); i++){
  fl_DatiOut<<vectAna[i].idPersona<<";";
  fl_DatiOut<<vectAna[i].cognome<<";";
  fl_DatiOut<<vectAna[i].nome<<";";
  fl_DatiOut<<vectAna[i].indirizzo<<";";
  fl_DatiOut<<vectAna[i].localita<<";";
  fl_DatiOut<<vectAna[i].sigla_provincia<<";";
  fl_DatiOut<<vectAna[i].telefono<<std::endl;
}
}//saveAnagrafiche
```

Per quanto riguarda il salvataggio delle anagrafiche si crea la funzione con nome "saveAnagrafiche" con parametri il file, passato by reference e il vettore di anagrafica.Infine con l'uso del ciclo for e tramite l'operatore (<<) i dati potranno essere salvati sul file.

9. Realizzazione di un menù:

Attraverso il menu possiamo avere un'interfaccia semplice con la quale far scegliere all'utente che azione vuole compiere riguardo le sue anagrafiche.

```
int showMenu(){
unsigned int sceltaMenu;
std::cout << "\n\n\t\t------ANAGRAFICHE ------\n";
        std::cout << "\n\tBenvenuto nella gestione Anagrafiche cosa vuoi fare?:\n";
        std::cout << "\t [1]  Inserire nuova Anagrafica\n";
        std::cout << "\t [2]  Ricercare una Anagrafica\n";
        std::cout << "\t [3]  Cancellare una Anagrafica\n";
        std::cout << "\t [4]  Stampare le Anagrafiche\n";
        std::cout << "\t [5]  Salvataggio modifiche\n";
        std::cout << "\t [6]  Exit\n";
        do {
            std::cout << "\n n scelta: ";
            std::cin >> sceltaMenu;
        } while (sceltaMenu < 1 || sceltaMenu > 7);

 return sceltaMenu;
}
```

Qui vengono indicate tutte le possibili modifiche che si possono apportare alle anagrafiche

```
switch(sceltaMenu){
        case 1:
            //inserimento nuova Anagrafica;
            insertAnagrafica(vectAna);
            break;
        case 2:

            std::cout << "Inserisci l'ID Anagrafica che vuoi cercare:  ";
            std::cin >> idAna;
            idxAnaTrovato = SeachAnaById(vectAna,idAna);
            if (idxAnaTrovato!=-1)
                showAnagrafica(vectAna,idxAnaTrovato);
            else
                std::cout << "[!] ID Anagrafica non presente\n";
            break;
        case 3:
            std::cout << "Inserisci l'ID della persona che vuoi eliminare:  ";
            std::cin >> idAna;
            idxAnaTrovato= SeachAnaById(vectAna, idAna);
            if (idxAnaTrovato==-1)
                std::cout << "[!] ID Persona non presente\n";
            else
             deleteAna(vectAna,idxAnaTrovato);
            break;
        case 4:
            showAnagrafiche(vectAna);
            break;
        case 5:
            //salvataggio modifiche;
            fl_Dati_Out.open("Anagrafiche.csv");
            if (fl_Dati_Out.fail()){
                std::cout <<"errore apertura file per scrittura"<<std::endl;
                exit(0);
            }
            saveAnagrafiche(fl_Dati_Out,vectAna);
            fl_Dati_Out.close();
            break;
        case 6:
            fineProgramma=true;
            break;
        }//switch
 return fineProgramma;
}
```

Qui viene mostrato il menù con tutte le varie opzioni dello switch, con tutte le possibilità dove ognuna corrisponde alle diverse funzioni illustrate in precedenza.