**##Note secondo progetto##**

**Definizione dati:**

**Prenotazioni**:

Lista di liste presente all'interno dei campi del nodo utente

la principale avrà come suoi campi la città di partenza, la prossima prenotazione e

il puntatore alla lista delle destinazioni

Le destinazioni sono definite da Nome della città, il costo economico per arrivarci,

la distanza e un puntatore alla successiva ove ci sia.

**typedef struct lista\_prenotazioni** {

char[maxstring] cittàPartenza;

struct lista\_prenotazioni \*next;

struct lista\_destinazioni \*dest;

} **prenotazione**;

**typedef struct lista\_destinazioni**{

char[maxstring] città;

int economy;

int distanza;

struct lista\_destinazioni \*next;

} **destinazione;**

**Utenti:**

Nodi di un ABR, con chiave di ricerca il nomeutente

**Campi**:

-nome utente

-password

-Lista prenotazioni

-Sottoalbero sinistro

-Sottoalbero destro

**typedef struct nodo\_utente** {

char[maxstring] nomeUtente;

// char[maxstring] pswd;

struct lista\_prenotazioni \*prenotazioniUtente;

struct nodo\_utente \*sx;

struct nodo\_utente \*dx;

} **Utente**;

**Grafo:**

Numero vertici

Lista di adiacenza\*\*

L'arco avrà:

-target;

-pesoEconomico;

-pesoDistanza;

-next;

**typedef struct graphType** {

int nv;

edge \*\*adj;

} **Graph**;

**typedef struct edgeType** {

-int key

-char[maxstring] città;

-int pesoEconomy;

-int pesoDistanza;

-struct edge \*next;

} **edge**;

**Interfacce:**

**PRINCIPALE**:

avvio programma con 3 possibilità:

-Login;->UI1.a ->UI2

-Registrazione;->UI1.b

-Chiusura programma;

**Registrazione:** avvio interfaccia e processo di registrazione, al termine di essa si ritorna al UI1.

**Chiusura**: termina il programma.

**Login**: inserimento dei dati utenti e avvio del menù corrispettivo

(Sys, admin -> menuSys ; nomeutente,passutente -> menuUtente)

collegamento a UI2.

**Interfaccia Registrazione(UI1.b)**:

-Inserire nome utente -> // inserire password // -> Registrazione effettuata/rifiutata;

-Controllo se l'utente è già esistente affinché si abbia un risultato positivo o negativo.

-Registrazione effettuata-> messaggio di riuscita -> UI1;

-Registrazione rifiutata-> messaggio di errore -> UI1 o si riprova finché non viene accettata?

**Interfaccia Login(UI1.a)**:

-Inserire nome utente -> // inserire password // -> Login effettuato/rifiutato;

-Controllo sull'esistenza dell'utente, // o se la password è corretta//

-Login effettuato-> messaggio di benvenuto -> UI2 Utente o System

-Login rifiutato -> messaggio di errore -> UI1 o riprova

**Interfaccia Login UtenteAdmin(UI2):**

UI2.1(system Menu):

**Menu dell'amministratore di sistema**:

-Aggiungere Destinazione;->UI2.1.a

-Aggiungere tratta;->UI2.1.b

-Rimozione Destinazione;->UI2.1.c

-Rimozione tratta;->UI2.1.d

-Logout;

**Logout**: permette di uscire dalle credenziali di sistema, ritorno a UI1;

**Aggiungere Destinazione**: permette di aggiungere un ulteriore nodo al grafo,

e quindi aggiungere una nuova città;

**Aggiungere Tratta**: permette di aggiungere un arco tra un nodo partenza e un nodo destinazione,

e quindi aggiungere un volo da una città ad un altra;

**Rimozione Destinazione**: permette di rimuovere il nodo di un grafo,

e quindi rimuovere una città (Conseguente rimozione di tutti gli archi da e per quel nodo);

**Rimozione Tratta**: permette di rimuovere l'arco da un nodo all'altro,

e quindi rimuovere il volo da una città ad un altra

Sulle rimozioni è ben notare la necessità di controllare se questo possa aver influito le prenotazioni

degli utenti.

Al termine di ogni operazione(ad eccezione di logout), si ritorna a **SYSTEM MENU’(UI2.1);**

**AGGIUNGERE DESTINAZIONE (UI2.1.a):**

inserire nome città -> controllo -> aggiunta destinazione riuscita/fallita

controllo se la cità è già presente nel grafo

se è presente, l'aggiunta fallisce con conseguente messaggio di errore

se non è presente, si effettua l'aggiunta del nodo con conseguente messaggio di riuscita

(provvedere anche a chiedere l'inserimento di archi?)

**AGGIUNGERE TRATTA(UI2.1.b):**

-inserire partenza -> controllo -> inserire destinazione -> controllo -> aggiunta tratta riuscita/fallita

-controllo se le città inserite sono presenti nel grafo e se l'arco non è già presente

-se uno di questi controlli fallisce allora l'aggiunta fallisce con messaggio di errore

altrimenti viene aggiunto l'arco con conseguente messaggio di riuscita

**RIMOZIONE DESTINAZIONE (UI2.1.c):**

-inserire città da rimuovere -> controllo -> rimozione riuscita/fallita -> avviso agli utenti?

-controllo se la città da rimuovere è presente nel grafo

-a rimozione riuscita,se la città rimossa faceva parte di una prenotazione,

-bisogna eliminare la prenotazione coinvolta e avvisare gli utenti coinvolti(il come da definire)

**RIMOZIONE TRATTA(UI2.1.d):**

-inserire la città di partenza ->controllo -> inserire la città di destinazione -> controllo x 2 ->

-> rimozione tratta -> avviso agli utenti?

-controllo se le città inserite esistono nel grafo e se esiste l'arco dalla città di partenza a quella d'arrivo

-se le condizioni sono soddisfatte, elimino l'arco e elimino le prenotazioni che usufruivano di quell arco avvisando gli utenti della rimozione della prenotazione.

**MENU’ UTENTE(UI2.2):**

**-Visualizzare Prenotazioni Attive;->UI2.2.a**

**-Effettuare una prenotazione;->UI2.2.b**

**-Logout;**

Visualizzare prenotazioni attive: Stampa della lista prenotazioni per l'utente specifico;

Effettuare una prenotazione: aggiunta di un nodo (in testa o coda?) nella lista delle prenotazioni;

Logout: permette di uscire dalle credenziali dell'utente, ritorno a UI1;

Al termine di ogni operazione(ad eccezione di logout), si ritorna a **MENU’ UTENTE(UI2.2)**;

**VISUALIZZA PRENOTAZIONI ATTIVE(UI2.2.a):**

Semplice stampa della lista delle prenotazioni.

**EFFETTUA UNA PRENOTAZIONE(UI2.2.b):**

-Partenza e destinazione;

-Solo partenza;

i) inserire città partenza -> controllo -> inserire città destinazione -> controllo x2 ->

->Tratta economica / tratta breve ->aggiunta/fallimento della prenotazione

-Controllo della presenza delle città e dell'esistenza di almeno un percorso,

-Se il percorso è unico, sarà sia la più economica che la più breve

-Se esistono più percorsi, effettuare Dijikstra sia per peso di costo che per peso di distanza

e chiedere la preferenza, riepilogo di costo e percorso a video con richiesta di conferma.