Relazione progetto d'esame di algoritmi e strutture dati

Per rappresentare il piano è stato implementato un grafo non orientato non pesato e fortemente connesso. Il grafo è rappresentato con una lista di adiacenzadove ad ogni inserimento viene aggiornata l'adiacenza con gli altri elementi all'interno della lista del nodo.

Strutture dati coinvolte

Il tipo `piano` è usato per rappresentare un grafo non orientato pesato e connesso, mediante lista di adiacenza. Per rappresentare la lista di adiacenza il piano è rappresentato come una slice di punti contenenti ostacoli oppure automi. Ogni ostacolo o automa è memorizzato tramite una struttura `punto` che rappresenta il nodo del grafo.

Il tipo `punto` è usato per rappresentare un generico punto nel piano. Esso è commposto da un id, due valori interi che rappresentano rispettivamente la coordinata x e la coordinata y, una variabile `richiamo` che indica se il richiamo α vale per l'automa considerato e una slice `adiacenza` che contiene i puntatori ai punti adiacenti ad un'automa se non sono vuoti. Di fatti se un automa ha adiacente un ostacolo o un altro automa esso verrà inserito nella slice. Se l'`adiacenza` è vuota vuol dire, per l'automa, che è circondato da punti vuoti.

Gli ostacoli hanno la slice `adiacenza` sempre vuota. Questa è una scelta di comodo dato che un ostacolo non si può muovere all'interno del campo.

Metodi e funzioni

Funzioni

- `crea()` : Questa funzione inizializza la variabile globale `Campo`, in modo da poter inserire le varie entità all'interno del piano. Questa funzione è preferibile utilizzarla se si vuole creare un nuovo piano, partendo da uno già esistente. All'inizio il `Campo` è vuoto e non c'è bisogno di inizializzarlo per inserire elementi al suo interno.
- `stampa()` : Questa funzione stampa prima un elenco contenente tutti gli automi che fanno parte del piano e poi un altro contenente tutti gli ostacoli.
- `stato(x, y int)` : Questa funzione stampa il tipo di entità presente in un determinato punto. Se nelle coordinate `x`, `y` è presente un automa, allora la funzione stamperà `A`, se vi è un punto fancente parte di un ostacolo, stamperà `O` e invece se non c'è niente, stamperà `E`.
- `automa(x, y int, eta string)` : Questa funzione aggiunge un automa al campo se le coordinate `x`, `y` non fanno parte di un ostacolo, se no non fa nulla. Se le coordinate `x`, `y` non fanno parte di un'ostacolo allora controlla

se l'automa `eta` esite già e in caso affermativo sposta l'automa altrimenti lo crea nuovo.

- `ostacolo(x0, y0, x1, y1 int)` : Questa funzione aggiunge un ostacolo all'interno del `piano`. Se l'are del quadrato compresa tra il vertice in basso a sinistra, `x0` e `y0`, e il vertice in alto a destra, `x1` e `y1`, contiene un'automa la funzione termina. Altrimenti aggiunge un punto con coordinate, `(x0, y1)` con id uguale `(x0, y0, x1, y1) ostacolo`.

  Considerazioni sulla funzione ostacolo: Per il calcolo dell'area del quadrato verranno usate le funzioni `dentroAreaOstacolo` e `estraiCoordinate` all'occorrenza all'inetrno del programma. Gli ostacoli non hanno adiacenze. Questa scelta è stata presa dato che, non salvando l'intera area del quadrato ma solo un punto in memoria, non sembrava utile inserire le adiacenze con un solo punto dell'ostacolo. Inoltre gli ostacoli non cambiano posizione durante lo svolgimento del programma quindi non seve salvarne l'adiacenza.

  - `dentroAreaOstacolo(x, y int) bool` : Questa funzione serve per calcolare se un punto è all'interno dell'area di un ostacolo. Restituisce `true` se il punto `(x, y)` fa parte dell'area di un generico ostacolo all'interno del campo.
  - `estraiCoordinate(id string) (x0 int, y0 int, x1 int, y1 int)` : Questa funzione è ustata unicamente all'interno di `dentroAreaOstacolo` per estrarre le coordinare del punto in basso a sinistra e in alto a destra dall'ostacolo.

- `richiamo(x, y int, alpha string)` : Questa funzione avvia il richiamo per, e soltanto per, gli automi che come prefisso hanno la stringa `alpha`.
- `posizioni(alpha string)` : Questa funzione mi stampa tutti gli automi all'intenro del campo che come prefisso per l'id hanno la stringa `alpha`.
- `esistePercorso(x, y int, eta string)` :
- `esegui(p piano, s string)` :
- `newPiano() piano` :

Metodi

- `(*piano)cerca(x, y int, id string)*punto` : Questo metodo serve per cercare uno specifico punto, attraverso le coordinate ed eventualmente il nome, all'interno del piano. Se il punto non è presente il metodo restituisce `nil`, altrimenti da il punto all'ineterno del piano.
- `(p *punto)adiacenze()` : Questo metodo gestisce e regola le adiacenze di un punto all'interno del piano. Peremettendo di cambiare o definire i

punti adiacenti ad un altro punto. Più a basso livello permette di collegare i vari nodi fra di loro mettendoli in relazione.

- `(*piano)remouve(eta string)` : Questo metodo rimuove un automa dal piano. Non cambia le adiacenze degli altri nodi che erano in relazione con esso.
-