HOMEWORK di STEFANO BINOTTO

COMMITTENTI: EDOARDO BASTIANELLO, FILIPPO MANZARDO

INTRODUCTION

Questo codice contiene le basi per costruire un gioco più complesso. Il sistema è in grado di risolvere tutti i problemi richiesti in fase di colloqui dal committente. In sostanza il programma si occupa di creare una campo da gioco, una mappa, la quale poi viene riempita di pedine create secondo le indicazione ricevute da file. Il programma poi risponde a 6 domande chieste dal committente. I servizi offerti al cliente sono: calcolare i valori di attacco e di difesa di ogni casella nelle due fasi della giornata, calcolare il numero di pezzi per ciascun tipo presenti sulla mappa e individuare la casella con il maggiore numero di pezzi della stessa tipologia. Tutto ciò viene dopo la lettura da file delle coordinate e del tipo del pezzo che viene aggiunto nella mappa.

USER REQUIREMENTS DEFINITION

Input()	
Function	Lettura dati da file ed assegnazione pedine su mappa
Description	Lettura da file di input dei dati che serviranno ad aggiungere pedine di diverso tipo in caselle diverse
Inputs	/
Source	/
Outputs	Memorizza i vari pezzi con relative coordinate
Destination	Mappa
Action	Il codice permette l'accesso ad uno dei tre file di input disponibili (input1 senza errori, input2 con coordinate fuori limite e tipologia pedine errate, input3 con una parola al posto della coordinata) ed eseguirebbe l'aggiunta dei pezzi riportati all'interno della mappa, massimo 5 per casella. I file di input devono avere, per ogni riga, un numero per indicare la coordinata X, uno per la coordinate Y ed il tipo della pedina (elfo, nano, orco), in questo esatto ordine. Se le coordinate riportate nel file e la tipologia di pedina fossero non accettabili la pedina viene ignorata. Mentre se il tipo di dato inserito non corrisponde a quello aspettato (un numero invece di una parola), il programma si interrompe. Esegue questa scansione fino a quando vi saranno righe da leggere nei file. I file a disposizione contano ognuno 30 pedine. Il primo non contiene errori e quindi permette l'aggiunta di tutte e 30 le pedine, il secondo file contiene delle coordinate fuori limite e dei tipi non accettabili, quindi verranno aggiunte solamente le pedine con relativi dati accettabili. Il terzo file contiene parole invece di numeri portando il programma al blocco immediato. Di default il programma imposta il tempo della mappa come "Giorno"
Requirements	La presenza dei tre file di input
Pre-condition	/
Post-condition	/
Side effects	Se le coordinate riportate nel file e la tipologia di pedina fossero non accettabili la pedina viene ignorata. Mentre il tipo di dato

	inserito non corrisponde a quello aspettato (int invece di string), il programma si interrompe. Se durante la scelta del file da leggere di inserisce il nome del file sbagliato allora il programma darà l'errore "File non trovato"

Gioca()	
Function	Stampa domande
Description	Lettura da terminale per selezionare il file da cui estrarre i dati e stampa delle risposte
Inputs	Lettura sul terminale del file da scegliere come input
Source	- Tastiera via terminale - File "input1", "input2" o "input3"
Outputs	Risposte alle domande
Destination	Terminale
Action	-Lettura da terminale per selezionare file da aprire. -Richiamo di Input() -Stampa di ogni risposta utilizzando i metodi
Requirements	La mappa
Pre-condition	La presenza di almeno una pedina nella mappa
Post-condition	/
Side effects	/

SYSTEM ARCHITECTURE

Il sistema è composto da 4 classi:

Gioco : {main()}

Mappa : {size(), isEmpty(), setTIme(), Input(), setTipo(), totElfi(), totOrchi(), totNani(), maxDefDay(), maxDefNight(), maxAttDay(), maxAttNight(), maxTipo(), Gioca()}

Casella : {size(), isEmpty(), addPezzo(), removePezzo(), getTotAtt(), getTotDef(); contaElfo(), contaNano(), contaOrco(), toString()}

Pezzo : {getAtt(), getDef(), getTipo()}

Quest'ultima è a sua volta composta a tre classi figlie, ognuna per il tipo di pedine accettabili:

Elfo : {getDef()}

Nano : {getAtt()}

Orco : {getAtt(), getDef()}

SYSTEM REQUIREMENTS SPECIFICATION

Domanda 1	
Function	Il numero di pezzi presenti sulla mappa per ciascuna tipologia

Description	Conta il numero di pezzi di ciascun tipo presenti sulla mappa
Inputs	Contatori dei tre pezzi di ogni casella
Source	Tutte le caselle della mappa
Outputs	Quantità di ciascun tipo di pezzi presenti in tutta la mappa
Destination	Output su terminale
Action	Prende la quantità di elfi, nani ed orchi di ogni casella e li somma tra di loro in modo da restituire il numero di quante pedine dello stesso tipo ci sono nella mappa.
Requirements	La mappa
Pre-condition	/
Post-condition	/
Side effects	/

Domanda 2	
Function	La casella con il maggior valore di difesa di giorno
Description	Trova la casella o le caselle che hanno complessivamente il maggior valore di difesa di giorno
Inputs	Quantità dei tre tipi di pezzi in ogni casella
Source	Caselle
Outputs	Casella/e con il maggiore valore di difesa di giorno
Destination	Output su terminale
Action	Si ricerca il massimo valore di difesa di giorno tra tutte le caselle della mappa. Questo valore è la somma delle singole difese dei pezzi dentro la casella di giorno. Si restituiscono le coordinate e l'elenco dei pezzi presenti all'interno della casella o delle caselle selezionate
Requirements	La mappa
Pre-condition	/
Post-condition	/
Side effects	/

Domanda 3	
Function	La casella con il maggior valore di difesa di notte
Description	Trova la casella o le caselle che hanno complessivamente il maggior valore di difesa di notte
Inputs	Quantità dei tre tipi di pezzi in ogni casella
Source	Caselle
Outputs	Casella/e con il maggiore valore di difesa di notte
Destination	Output su terminale

Action	Si ricerca il massimo valore di difesa di notte tra tutte le caselle della mappa. Questo valore è la somma delle singole difese dei pezzi dentro la casella di notte. Si restituiscono le coordinate e l'elenco dei pezzi presenti all'interno della casella o delle caselle selezionate
Requirements	La mappa
Pre-condition	/
Post-condition	/
Side effects	/

Domanda 4	
Function	La casella con il maggior valore di attacco di giorno
Description	Trova la casella o le caselle che hanno complessivamente il maggior valore di attacco di giorno
Inputs	Quantità dei tre tipi di pezzi in ogni casella
Source	Caselle
Outputs	Casella/e con il maggiore valore di attacco di giorno
Destination	Output su terminale
Action	Si ricerca il massimo valore di attacco di giorno tra tutte le caselle della mappa. Questo valore è la somma dei singoli attacchi dei pezzi dentro la casella di giorno. Si restituiscono le coordinate e l'elenco dei pezzi presenti all'interno della casella o delle caselle selezionate
Requirements	La mappa
Pre-condition	/
Post-condition	/
Side effects	/

Domanda 5	
Function	La casella con il maggior valore di attacco di notte
Description	Trova la casella o le caselle che hanno complessivamente il maggior valore di attacco di notte
Inputs	Quantità dei tre tipi di pezzi in ogni casella
Source	Caselle
Outputs	Casella/e con il maggiore valore di attacco di notte
Destination	Output su terminale
Action	Si ricerca il massimo valore di attacco di notte tra tutte le caselle della mappa. Questo valore è la somma dei singoli attacchi dei pezzi dentro la casella di notte. Si restituiscono le coordinate e l'elenco dei pezzi presenti all'interno della casella o delle caselle selezionate

Requirements	La mappa
Pre-condition	/
Post-condition	/
Side effects	/

Domanda 6	
Function	La casella con il maggior numero di pezzi dello stesso tipo
Description	Trova le caselle con il maggior numero di pezzi dello stesso tipo
Inputs	Contatori dei tre pezzi di ogni casella
Source	Tutte le caselle della mappa
Outputs	Casella/e con il maggior numero di pezzi dello stesso tipo
Destination	Output su terminale
Action	Per ogni tipologia di pedina si cerca il numero massimo di pedine di quel tipo nelle caselle della mappa. Si restituisce le caselle con quel numero di pedine
Requirements	La mappa
Pre-condition	/
Post-condition	/
Side effects	/

Casella.java	
Function	Implementazione di una casella
Description	Implementazione attributi e metodi di una casella della mappa
Inputs	/
Source	/
Outputs	Diversi metodi da utilizzare per risolvere compiti più elaborati
Destination	Mappa
Action	La casella viene definita così: -una serie di massimo 5 pedine -un numero che indica quanti pezzi contiene -il tipo della casella (bosco, pianura montagna) -metodi per aggiungere e rimuovere a scelta una pedina -metodi per il calcolo della difesa e attacco della casella in base al momento della giornata -metodi per contare il numero di pedine dello stesso tipo nella casella -metodo che stampa la descrizione della pedina
Requirements	La classe Pezzo e figlie
Pre-condition	/
Post-condition	/

Side effects	/

Pezzo.java	
Function	Implementazione di una pedina
Description	Implementazione attributi e metodi di una pedina
Inputs	/
Source	/
Outputs	Diversi metodi da utilizzare per risolvere compiti più elaborati
Destination	Casella
Action	La pedina viene definita così: -valori di attacco, difesa e tipologia associati alla pedina -metodi che ritornano gli attributi della pedina Sono associate le tre classi figlie elfo, nano e orco con override dei metodi precedentemente citati
Requirements	/
Pre-condition	/
Post-condition	/
Side effects	/

Mappa.java	
Function	Implementazione di una mappa
Description	Implementazione attributi e metodi di una mappa
Inputs	Pedine da inserire lette da file
Source	/
Outputs	/
Destination	Gioco
Action	La mappa viene definita così: -un’array bidimensionale di 25 caselle -un numero che indica quante pedine contiene -il tempo della giornata della mappa (giorno, notte) -metodi per settare il tempo e la tipologia delle caselle -metodo per input da file delle pedine -metodi per calcolare il n° totale di elfi/orchi/nani nella mappa -metodi per trovare le caselle con il maggior valore di attacco o difesa di giorno o notte -metodo per trovare le caselle con il maggior numero di pezzi dello stesso tipo -metodo per stampare le risposte alle domande
Requirements	Classi Casella, Pezzo e figlie
Pre-condition	/
Post-condition	/

Side effects	/

Gioco.java	
Function	Metodo main()
Description	Eseguire il gioco stampando le risposte richieste
Inputs	/
Source	/
Outputs	/
Destination	Terminale
Action	Creazione di una mappa e ritorno delle risposte tramite metodo Gioca() della classe Mappa
Requirements	Classi Mappa, Casella, Pezzo e figlie
Pre-condition	/
Post-condition	/
Side effects	/

SYSTEM EVOLUTION

Il codice fino ad ora sviluppato, oltre a disporre delle funzioni richieste per risolvere i problemi assegnati, contiene anche dei metodi addizionali, come quelli per il settaggio del tempo o del tipo delle caselle, che potranno costituire le basi per l'implementazione di un sistema più complesso ed elaborato.