

Progetti per il Corso di Laboratorio di Interfacce Uomo – Macchina

Progetto “Editor di Grafi”

Docenti di riferimento: Proietti, Forlizzi

Si realizzi una versione sviluppata in linguaggio C o C++ di un editor di grafi sviluppato in linguaggio Java. L’editor permette di creare dei grafi, di vario tipo, attraverso un’interfaccia grafica di facile utilizzo. La nuova versione dell’editor dovrà essere compatibile a livello di formato dei file con la precedente, e dovrà offrire (più o meno) le stesse funzionalità.

Progetto “Gestione Convegni”

Docenti di riferimento: Proietti, Forlizzi

Si realizzi un sistema per la gestione di un archivio di convegni. Il sistema deve permettere la memorizzazione delle informazioni principali relative ad un convegno, tra le quali le più importanti sono: nome, luogo, periodo di svolgimento, termine per la sottomissione di lavori, data di notifica di accettazione/rifiuto di un lavoro, termine per la sottomissione della versione finale di un lavoro. Inoltre dovrebbe prevedere varie funzionalità da esprimere a video. Ad esempio la possibilità di ordinare (mostrando l’ordine a video) i convegni in base a vari criteri, e la segnalazione automatica di sovrapposizioni e/o prossimità di scadenze relative a diversi convegni.

Progetto “Interfaccia Solutore”

Docente di riferimento: Rossi

Progettare un’interfaccia per un solutore di Programmazione Lineare a linea di comando. Ambiente: UNIX, LINUX. L’interfaccia può essere sviluppata in Java o Tcl/tk.

Progetto “Gonnect”

Docenti di riferimento: Nesi, They

L’obiettivo del progetto è implementare il gioco "Gonnect" in Java. Questo gioco è stato inventato da João Neto. Il gioco è una combinazione di due giochi famosi: Go and Hex. Si gioca su una tavola costituita da una griglia di righe orizzontali e verticali di dimensione 13x13. I due giocatori, alternativamente, riempiono la tavola mettendo delle pedine (una alla volta) sulle intersezioni libere. Le regole del gioco sono le seguenti:

Swap: Invece della sua prima mossa, il secondo giocatore può scegliere di cambiare i colori.

Catena: Pedine dello stesso colore che sono adiacenti (verticalmente o orizzontalmente) formano una *catena*: due pedine appartengono alla stessa catena se e solamente se esiste un modo per passare dall’una all’altra camminando (verticalmente o orizzontalmente) solamente su pedine dello stesso colore.

Libertà: Una pedina ha una *libertà* se almeno una delle sue intersezioni adiacenti è libera. Una catena ha una libertà se contiene almeno una pedina con una libertà.

Cattura: Se in seguito all’inserimento di una pedina una catena si ritrova senza libertà, tutte le pedine che la compongono vengono catturate e sono rimosse dalla tavola.

No suicide: Una mossa non può essere fatta se ha per conseguenza la cattura di una catena dello stesso colore della pedina aggiunta.

Mossa: Si può fare una mossa solamente se la catena alla quale appartiene la nuova pedina ha una libertà. La libertà è calcolata dopo le eventuali catture delle catene del colore opposto alla pedina aggiunta.

Vittoria: Un giocatore vince se connette con una catena due lati opposti della tavola o se l'altro giocatore si trova nell'impossibilità di fare una mossa, ovvero tutte le sue mosse formano catene senza libertà.

Un primo obiettivo del progetto è di avere un'implementazione del gioco in Java che convalida le mosse dei due giocatori e indica alla fine quale dei due ha vinto. A partire da questa base, l'implementazione può essere completata realizzando per esempio:

- Un'interfaccia grafica (forse un applet) con cui si può visualizzare la tavola e giocare utilizzando il mouse
- Un programma che gioca il ruolo dell'avversario
- Un server di giochi che permetta a diversi giocatori di connettersi per giocare fra di loro
- Una versione tridimensionale del gioco

Per ulteriori informazioni si veda l'indirizzo web

<http://www-sop.inria.fr/lemme/Laurent.Thery/lsp/progetto2006.html>

Progetto “Schedina Automatica”

Docente di riferimento: Muccini

Si realizzi una interfaccia per un sistema chiamato “Schedina Automatica”. Tale sistema si comporta come descritto nel seguito. “Sia data la lista delle squadre di calcio di un certo campionato. Ogni settimana, le squadre di calcio si confrontano in campo, ed il sistema deve generare una schedina automatica, basata sui risultati delle cinque partite precedenti. L'utente inserisce per ogni squadra considerata i risultati delle ultime cinque partite, nella forma (v,p,s) con v = #vittorie, p = #pareggi, e s = #sconfitte, tale che v+p+s sia sempre uguale a cinque (tali dati vengano inseriti da console). Successivamente, l'utente inserisce due codici (corrispondenti alle due squadre rivali salvate in memoria) (di nuovo da console). Il risultato previsto viene calcolato come nel seguito: date due squadre A e B, ed i risultati delle cinque partite piu' recenti $A = (v_A, p_A, s_A)$ e $B = (v_B, p_B, s_B)$, l'algoritmo calcola $\text{MAX} (|v_A - v_B|, |p_A - p_B|, |s_A - s_B|)$ (dove | rappresenta il valore assoluto). Se $\text{MAX} (|v_A - v_B|, |p_A - p_B|, |s_A - s_B|)$ e' il primo/secondo/terzo elemento della terna, allora A vince/pareggia/perde.”

Progetto “Distribuzione Multimediale”

Docente di riferimento: Muccini

Si realizzi l'interfaccia grafica per il seguente sistema. “Un importante distributore all'ingrosso di contenuti multimediali richiede la progettazione di un software per la gestione del magazzino. Ciascun prodotto offerto dal distributore è caratterizzato da: nome del file, dimensione, titolo, autore, tipologia. Alcuni esempi di tipologie sono: programma, suoneria, CD, film, gioco Playstation, etc. Il sistema deve permettere inserimento e cancellazione di prodotti nel magazzino, nonché la ricerca dei prodotti in base al titolo e/o all'autore. Di tanto in tanto, il distributore archivia su supporto ottico (CD, DVD) un'insieme di contenuti, al fine di spedirli alle bancarelle. Per ogni contenuto si dovrà tenere traccia del fatto che sia stato archiviato o meno, e in caso positivo, del numero d'indice del supporto che lo contiene. Il sistema dovrà selezionare i contenuti non ancora archiviati e pianificare in che modo memorizzarli in supporti ottici, cercando di contenere il numero di supporti richiesto.”

Progetto “Correttore Ortografico”

Docente di riferimento: Muccini

Si realizzi l’interfaccia grafica per il seguente sistema. “Si realizzi un correttore ortografico di testi. Il programma memorizza un insieme di regole di correzione ortografica e le applica ad un file di testo in ingresso, producendo in uscita un nuovo file di testo. Ciascuna regola è costituita da una coppia di stringhe, dette Errata e Corrige. L’applicazione di una regola consiste nel trovare nel testo in ingresso tutte le occorrenze della stringa Errata e sostituirle con altrettante occorrenze della stringa Corrige. Per ogni regola, inoltre, si possono specificare due modificatori: Case e Entire. Case indica se, nella ricerca delle occorrenze e nella sostituzione, lettere maiuscole e minuscole debbano essere considerate equivalenti o meno. Entire indica se la regola debba essere applicata o meno anche se la stringa Errata compare come parte di una stringa più grande. Il set di regole deve essere vasto, e tali regole devono essere salvate in memoria con il .data. Deve esser possibile visualizzare tali regole.”

Progetto “Bacheca Esami”

Docenti di riferimento: Proietti, Della Penna

Il progetto prevede la realizzazione di un’applicazione web per la prenotazione e l’organizzazione automatica degli appelli d’esame. L’applicazione deve prevedere l’accesso da parte di due categorie di utenti: gli *amministratori* e i *docenti*.

Agli amministratori spetta il compito di definire le finestre temporali in cui si tengono le sessioni d’esame. I docenti, invece, devono poter inserire, all’interno delle sessioni predefinite, le date degli appelli degli esami di cui risultano titolari, specificandone anche l’orario e, opzionalmente, il numero previsto di studenti.

Il sistema dovrà quindi verificare se la proposta del docente rispetta una serie di vincoli, tra i quali

- evitare il sovrappollamento di esami nello stesso giorno
- non inserire più di due esami appartenenti allo stesso anno di corso nello stesso giorno, e a meno di quattro ore di distanza l’uno dall’altro
- distanziare gli appelli di uno stesso esame di almeno 15 giorni l’uno dall’altro

L’applicazione dovrà prevedere un sistema tramite il quale gli amministratori possano variare i parametri dei vincoli predefiniti (ad es. la distanza tra gli appelli) o inserire dinamicamente nuovi vincoli simili a quelli appena descritti (la versatilità di questa funzione è a discrezione dei realizzatori, ed aumenterà notevolmente il valore del progetto).

Una volta verificati i vincoli, l’esame sarà inserito nel calendario ufficiale, consultabile da ogni utente del sistema, e non potrà più essere modificato dal docente. In caso contrario, il docente sarà avvisato dei problemi riscontrati e invitato a modificare la sua richiesta. Il sistema a questo punto potrebbe anche proporre una collocazione adeguata per l’appello in base a delle semplici euristiche. L’amministratore, in ogni caso, deve avere la possibilità di modificare a proprio piacimento e in ogni momento la collocazione degli esami, anche violando i vincoli elencati sopra.

Per ciascuna delle funzionalità deducibili dalle specifiche appena illustrate è necessario sviluppare un sistema di interfaccia il più possibile intuitivo e *user friendly*. Ad esempio, le funzionalità che prevedono la definizione di un intervallo di date (sessioni, appelli, ecc) dovrebbero essere gestite tramite un’interfaccia di tipo “calendario”, in cui siano evidenziati i giorni tra cui scegliere, quelli eventualmente già occupati (e in che misura), ecc.

Progetto “Giochi di Congestione non Cooperativi”

Docenti di riferimento: Flammini, Melideo

Il progetto prevede la realizzazione di una interfaccia grafica per la simulazione di giochi di congestione non cooperativi.

Nota: *il progetto è stato assegnato*