

Compito di Introduzione alla Programmazione Web
Prova al calcolatore – febbraio 2021

TESTO

Il Sudoku è un gioco nel quale al giocatore viene proposta una griglia quadrata composta di 9x9 celle, ciascuna delle quali può essere vuota o contenere una cifra compresa tra 1 e 9. La griglia è suddivisa in 9 “sottogriglie” quadrate non sovrapposte di 3x3 celle contigue dette “regioni”.

La regola è che in ogni riga, in ogni colonna e in ogni regione non vi siano simboli ripetuti.

Lo scopo del gioco è quello di riempire tutte le caselle rispettando la regola.

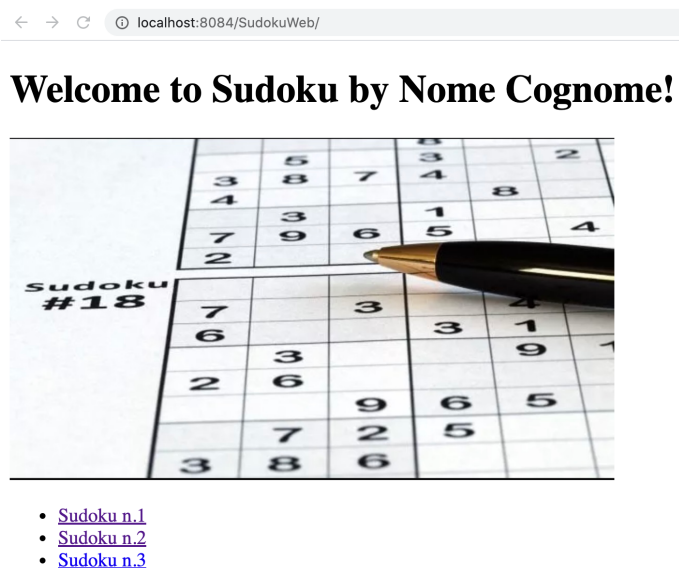
In tal modo in ogni riga, ogni colonna e ogni regione saranno presenti tutti i simboli, senza ripetizioni.

Si scriva un’applicazione Web che gestisce un Sudoku. Si usino CSS per la formattazione, e si riproducano le figure riportate nel seguito.

In particolare:

- 1) Il sistema mantiene una serie di diverse istanze del gioco (configurazione iniziale e soluzione). Noi partiremo da una situazione nella quale sono presenti tre istanze. Ciascuna istanza contiene un identificatore, la soluzione, la definizione delle celle fisse. In nessun caso la soluzione deve essere nota al client.
- 2) Le istanze del gioco sono contenute in un Database. Il DB (Derby) deve chiamarsi ExamDerbyDB, essere memorizzato all’interno del progetto Netbeans, essere creato con user WEBENGINE e pw WEBENGINE. Si veda più avanti come popolarlo.
- 3) La pagina iniziale del gioco (creata dinamicamente leggendo i dati dal DB) propone la lista delle istanze, identificandole con un numero univoco. L’immagine usata è sudoku.jpg, scaricabile da latemar.science.unitn.it/EsameWeb. Cliccando su un elemento della lista viene caricata la pagina del gioco. (Al posto di Nome Cognome va il vostro nome)

Home page:



Pagina del gioco:

4		6		5		7		9
	8	5				6	3	
3	1			7	6		8	2
		7	2		3			
		3		9		8		1
	2				1	5		
9	4		3	8			6	7
	3	8				1	4	
5		2		1		3		8

riga: colonna:

- 4) La pagina di gioco contiene la griglia, un bottone “verifica” e due campi che permettono di inserire le coordinate riga e colonna (cifre tra 1 e 9) della cella per la quale si vuole verificare se il valore presente coincide con la soluzione corretta.
- 5) Le celle presenti nella griglia assumono un colore di sfondo che dipende dalla regione in cui si trovano. Le regioni lungo la diagonale e quelle negli angoli avranno colore MistyRose (FF E4 E1), quelle sopra e sotto la diagonale AliceBlue (F0 F8 FF) (Nota: MistyRose e AliceBlue sono denominazioni HTML ufficiali, si veda https://www.w3schools.com/colors/colors_names.asp).
- 6) Le celle sono di due tipi:
 - FISSE, con valori immutabili assegnati all’inizio di una partita e valore mostrato in colore RED (FF 00 00);
 - MODIFICABILI, inizialmente vuote e modificabili dall’utente. Il colore del valore mostrato dalle celle modificabili, quando non vuote, è BLACK (00 00 00).
- 7) Un click sulle celle fisse non avrà alcun effetto.
- 8) Un click sulle celle libere permetterà di digitarvi un valore di un solo carattere: se il carattere digitato è una cifra compresa tra 1 e 9 questo viene visualizzato, mentre per qualunque altro carattere digitato la cella resta o diventa vuota.
- 9) Il bottone “verifica” chiede al server di controllare se la cella le cui coordinate sono specificate nel relativo campo contiene un valore errato. Il server risponde senza che la pagina sia ricaricata: nel browser viene mostrato un alert che dice “Il valore nella cella selezionata è giusto/sbagliato” a seconda del caso.

Il compito è lungo: per ottenere la sufficienza non è necessario completare tutti i punti. Arrivate fin dove riuscite. **In ogni caso la consegna deve compilare senza errori, ed i punti implementati devono essere funzionanti correttamente.** Si vedano i suggerimenti nel seguito.

CODICE SQL PER LA GENERAZIONE DEI DATI

Come si può vedere:

- La colonna FIXEDCELLS contiene una sequenza di numeri di due cifre separati da spazi. Ciascun numero identifica le coordinate x e y di una cella fissa ($1 \leq x \leq 9$, $1 \leq y \leq 9$). Il numero delle celle fisse generalmente varia da partita a partita.
- La colonna SOLUTION contiene cifre tra 1 e 9 separate da spazi. Tali cifre rappresentano la soluzione corretta, disposta per righe (quindi le prime 9 cifre sono i valori della prima riga, le successive 9 quelle della seconda e così via).

```
CREATE TABLE "DATA" (ID INTEGER NOT NULL, FIXEDCELLS VARCHAR(200),  
SOLUTION VARCHAR(200), PRIMARY KEY (ID));
```

```
INSERT INTO WEBENGINE."DATA" (ID, FIXEDCELLS, SOLUTION)  
VALUES (1, '11 13 17 19 22 23 27 28 31 32 35 36 38 39 43 44 53 55 57 66 67  
71 72 74 75 78 79 82 83 87 88 91 93 97 99', '4 2 6 1 3 8 7 5 9 7 8 5 9 4 2 6 3 1 3 1 9 5  
7 6 4 8 2 8 6 7 2 5 3 9 1 4 1 5 3 4 9 7 8 2 6 2 9 4 8 6 1 5 7 3 9 4 1 3 8 5 2 6 7 6 3 8 7 2 9  
1 4 5 5 7 2 6 1 4 3 9 8');
```

```
INSERT INTO WEBENGINE."DATA" (ID, FIXEDCELLS, SOLUTION)  
VALUES (2, '14 18 22 24 25 31 32 34 38 39 44 45 46 49 51 54 56 59 61 64 65  
66 71 72 76 78 79 85 86 88 92 96', '6 9 7 8 5 4 3 1 2 4 3 8 2 9 1 7 6 5 5 1 2 3 7 6 9 4 8  
1 5 3 4 6 9 8 2 7 9 8 4 7 2 3 1 5 6 2 7 6 1 8 5 4 9 3 7 4 1 4 6 8 2 3 9 3 2 5 9 1 7 6 8 4 8 6  
9 4 3 2 5 7 1');
```

```
INSERT INTO WEBENGINE."DATA" (ID, FIXEDCELLS, SOLUTION)  
VALUES (3, '13 14 16 17 22 24 26 32 33 43 45 47 53 55 57 63 65 67 77 78 84  
86 88 93 94 96 97', '2 7 9 8 6 4 3 1 5 3 6 5 9 1 7 4 2 8 8 1 4 2 3 5 7 9 6 1 2 8 7 5 3 9 6 4  
6 4 3 1 2 9 8 5 7 6 9 7 6 4 8 1 3 2 9 3 6 4 8 2 5 7 1 7 8 1 5 9 6 2 4 3 4 5 2 3 7 1 6 8 9');
```

SUGGERIMENTI E GUIDA PER LO SVILUPPO

Suggerimenti generali:

- 1) Per individuare la regione, si consiglia di calcolare la parte intera di $ry=y/3$ e $rx=x/3$, dove y e x sono le coordinate (riga, colonna) della cella (da 0 a 8): ry e rx danno allora le coordinate della regione (da 0 a 2).
- 2) Per suddividere lato server il contenuto ottenuto dal database, si suggerisce di usare uno StringTokenizer:
<https://docs.oracle.com/javase/7/docs/api/java/util/StringTokenizer.html>

Siete liberi di sviluppare il codice come volete, ma qui avete una traccia per facilitare il lavoro.

- 1) Dopo aver preparato il DB, si costruisca la home dell'applicazione come una pagina dinamica che interroga il database e presenta una lista delle configurazioni disponibili, linkando ciascun elemento della lista ad una configurazione.
- 2) Si crei (per il momento separatamente in un file chiamato dynamicTable.html) una pagina HTML di gioco chiamata contenente una funzione Javascript che al caricamento della stessa generi dinamicamente il contenuto della tabella HTML (9x9 celle vuote). Si editi poi la funzione di generazione in modo da aggiungere progressivamente le seguenti funzionalità (*suggerimento: tenete sempre copia dell'ultima versione funzionante!*):
 - a. Far sì che ogni cella della tabella contenga un text input field il cui identificatore sono le coordinate della cella (un numero con la cifra della riga nelle decine e quella della colonna nelle unità). Per ora lasciamo libero l'inserimento di dati nei text input field: ci preoccuperemo dopo di controllarlo.
 - b. Definire due stili, ciascuno dei quali rappresenta un colore di background
 - c. Applicare lo stile agli input fields a seconda della regione in cui si trovano.*Volendo si possono porre i punti b e c a più tardi.*
- 3) Si faccia in modo che il click su un elemento della lista nella pagina iniziale chieda al server di generare la pagina HTML che abbiamo prototipato al punto 2. Il server inserirà nel JS della pagina l'informazione sulle celle fisse (coordinate e valori), rappresentate ad esempio come un array di oggetti Javascript, ma per ora possiamo porre la loro visualizzazione. (*Ci sono vari modi per implementare questo punto, una possibilità è di copiare il codice che abbiamo prototipato in una pagina JSP e aggiungere quel che serve per la gestione dinamica dei dati*).
- 4) Si implementi il meccanismo di interazione con il server che permette di verificare se il contenuto della cella le cui coordinate sono indicate a fianco del bottone "verifica" sia corretto. Il browser deve passare al server le coordinate della cella e il suo valore ed ottenere la risposta che mostrerà in un alert informativo.
- 5) Si modifichi la pagina di generazione della tabella gestendone l'inizializzazione completa: si definisca una funzione JS "inizializza" che usa l'informazione sulle celle fisse predisposta al punto 3. Per tali celle viene
 - a. inizializzato il valore mostrandone il testo in RED
 - b. modificato il comportamento in modo che le celle non siano editabili.
- 6) Si modifichi la pagina di generazione della tabella gestendo l'inserimento dei dati:
 - a. Si vincoli l'input nei textfield delle celle ad un valore numerico di una sola cifra compresa tra 1 e 9. La pressione di un qualunque altro tasto svuota il contenuto della cella.