METODOLOGIE DI PROGRAMMAZIONE PER IL WEB

PRIMI PASSI CON NODE.JS

Lo scopo di questo primo lab è quello di iniziare ad acquisire familiarità con JavaScript (in Node.js), mettendo in pratica quello che abbiamo visto nelle lezioni (ed esercizi) delle prime due settimane. In aggiunta, questo lab fornirà le basi per i seguenti, iniziando un progetto che continuerà fino alla fine del corso.

ESERCIZIO 0 - PREPARAZIONE

Prima di tutto, controlla che la tua installazione di *Node.js* funzioni all'interno di Visual Studio Code.

Crea un nuovo progetto che, per ogni stringa in un array, ritorni una nuova stringa composta dai primi due e dagli ultimi due caratteri della stringa originale. La nuova stringa dovrà rimpiazzare quella vecchia nello stesso array.

Per esempio, '*estate*' genererà '*este*'

Se la stringa è più corta di due caratteri, ritornare la stringa vuota.

ESERCIZIO 1 – UN GESTORE DI "COSE DA FARE"

Data una serie di task (cioè, azioni che l'utente vuole fare in futuro), implementa un programma chiamato `todo-manager`.

In particolare, un task è composto dai seguenti campi:

- una *descrizione* testuale (obbligatoria)
- se è *importante* o no (default: "non importante")
- se è un task *privato* o *condiviso* (default: "privato")
- una *scadenza* (cioè, una data con o senza un'ora, opzionale)

Il programma deve permettere 4 azioni:

1. inserire un nuovo task;
2. rimuovere un task;
3. mostrare tutti i task esistenti, in ordine alfabetico;
4. chiudere il programma.

All'avvio, il programma deve mostrare un menu con le 4 opzioni e, per ogni scelta, attuare l'azione richiesta. Dopo ogni azione (tranne l'azione 4), il programma deve ritornare al prompt per richiedere un nuovo inserimento.

Per inserire un task, il programma chiederà i campi necessari, uno per linea.

Per rimuovere un task, l'utente dovrà indicare l'esatta descrizione testuale del task da rimuovere.

Suggerimenti:

1. *Per leggere dal terminale, puoi usare il modulo* `readline-sync`*:*
   https://www.npmjs.com/package/readline-sync.
   Per installare il modulo, lancia un terminale, vai nella cartella del progetto and digita "`npm install readline-sync`*" (avrai bisogno di una connessione Internet per farlo).*
2. *Hai 3 opzioni per eseguire il programma:*
   a. *manualmente, nel terminale (quello integrato in Visual Studio Code o in quello di sistema);*
   b. *in Visual Studio Code, per leggere qualcosa dal terminale, devi creare una launch configuration (launch.json) da Run activity (nella "Activity Bar", sulla sinistra) e aggiungere una nuova configurazione:*
      `"console": "integratedTerminal"`
3. *Per creare e gestire le date, puoi usare l'oggetto* `Date`*.*

ESERCIZIO 2 – ESTENDERE LA CANCELLAZIONE...

Modificare il programma sviluppato nell'esercizio precedente per cancellare tutti I task con una certa scadenza.

Per esempio, quando l'utente digita "2020-03-25", il programma userà le informazioni fornite per cancellare tutti i task la cui scadenza è alla data specificata.

ESERCIZIO 3 – ... E DIMENTICARE IL PASSATO!

Estendere il programma sviluppato negli esercizi precedente per cancellare automaticamente un task quando scade. Puoi testare questa funzionalità aggiungendo dei task la cui data è oggi (per esempio, "2020-03-27") e con un orario che corrisponda a qualche minuto dopo l'inserimento.

Extra: attenzione, 2020-03-25 deve venire dopo di 2020-03-25 14:21!

Suggerimento: Usa la funzione `setTimeout()` *di Note.js e l'oggetto* `Date` *per calcolare il valore di timeout quando il task viene inserito. Tieni presente che* `setTimeout()` *non funziona bene con i cicli: rimpiazza il ciclo con la funzione* `setInterval()`*.*