Compito di Programmazione
11 Luglio 2023

Nota Bene. Ogni esercizio deve essere svolto su un foglio diverso.
Scrivere Nome, Cognome e Matricola su ogni foglio.

1. **(punti 8)** Scrivere una funzione RICORSIVA che prende un array di interi e restituisce il numero massimo di duplicati che contiene. Ad esempio, se l'array in input è `1, 2, 3, 1, 2, 3, 3, 2, 2`, la funzione ritorna 4 perchè ci sono 4 duplicati di 2.
   [**N.B.** È possibile usare funzioni ausiliarie ricorsive, purchè definite.]

2. **(punti 8)** Si vogliono gestire gli appunti presi dagli studenti relativi al corso di programmazione del mese di Ottobre. Ogni appunto é caratterizzato da un testo (array di `char`), un numero di pagine (`int`) e una data (per semplicitá si consideri solo il giorno come `int`). Ogni appunto é collegato al successivo e al precedente (lista **bidirezionale**). La lista é **ordinata** per data (dal piú vecchio al piú recente). Definire la struttura dati necessaria per rappresentare la lista di appunti e le seguenti funzioni:

   - `rimuovi_appunto` che prende in input la lista ed il giorno di un appunto e rimuove l'appunto dalla lista. Nel caso in cui siano presenti piú elementi nello stesso giorno vanno rimossi tutti dalla lista.
   - `inverti_lista` che prende in input la lista e la restituisce in ordine inverso, ovvero dall'appunto piú recente a quello piú vecchio. La funzione deve modificare la lista originale: non é possibile creare una nuova lista.
   - `testo` che prende in input la lista e un array di `char` e copia il testo dell'appunto piú lungo (ovvero quello con il numero di pagine maggiore) all'interno dell'array.

   [**N.B.** É possibile utilizzare le funzioni strncpy, strcpy e strcmp della libreria cstring.]
   [**N.B.** Non é possibile restituire puntatori a `char`.]
   [**N.B.** Si possono usare funzioni ausiliarie se definite.]

3. **(punti 8)** Un aeroporto privato mette a disposizione un insieme di *slot* prenotabili per voli. Ciascuno slot è caratterizzato da una variabile binaria che indica se è prenotato o no, dal codice alfanumerico del volo che lo occupa e dalla destinazione del volo. Per semplicità si assuma che ci siano tanti slot quante sono le ore di un giorno e che la classe gestisca un giorno solo. Si implementi la classe `Aeroporto`, il relativo costruttore e i seguenti metodi:

   - `aggiungi_volo`: il quale riceve come parametri il codice, la destinazione e l'orario, verifica che quello slot sia libero e in caso positivo vi salva i relativi dati.
   - `rimuovi_volo`: il quale dato il codice di un volo, ne verifica la presenza negli slot e in caso positivo lo rimuove.

   Esistono anche aeroporti privati che mettono a disposizione una seconda pista, ma solo per elicotteri. È quindi possibile prenotare anche questa pista. Per semplicità, anche in questo caso, si assuma che ci siano tanti slot quante sono le ore di un giorno e che la classe gestisca un giorno solo. I voli con elicotteri si distinguono perché il primo carattere del codice è sempre la `h`. Si implementi la sottoclasse `AeroportoPlus` con il relativo costruttore. Sfruttando l'ereditarietà si modifichino opportunamente i metodi in modo tale da prenotare/rimuovere lo slot corretto a seconda del tipo di volo.

   [**N.B.** Si rappresenti uno slot tramite `struct` e si rappresenti il codice volo tramite array di `char`.]
   [**N.B.** In tutto l'esercizio 3 non è consentito usare le liste.]