Parziale di Programmazione I - BioInformatica

22 gennaio 2021 (tempo disponibile: 2 ore)

Esercizio 1 (20 punti)

(si consegni `permutations.c` e `permutations.h`)

Si scriva un programma `permutations.c` che implementa le seguenti tre funzioni:

```
// inizializza arr, lungo length, con una permutazione casuale
// dei numeri interi da 0 a length-1, usando srand() e rand()
void init_random(int arr[], int length);

// stampa su un'unica riga il contenuto dell'array arr, lungo length, poi va a capo
void print(int arr[], int length);

// determina se gli elementi dell'array arr, lungo length, sono tutti diversi fra loro
int different(int arr[], int length);
```

Per esempio, chiamando la funzione `init_random(arr, 5)` l'array `arr` potrà diventare {`3,0,1,2,4`}, oppure {`2,4,1,3,0`}, oppure {`0,1,2,3,4`}, oppure {`4,3,2,1,0`}, oppure qualsiasi altra permutazione di $0 \ldots 4$.

Si scriva quindi un file di header `permutations.h` che dichiara le precedenti tre funzioni.

Esercizio 2 (12 punti)

(si consegni `main.c`)

Si scriva un programma `main.c` che include le funzioni dell'Esercizio 1 tramite il file `permutations.h`. Il programma `main.c` deve contenere una funzione iniziale `main` che esegue le seguenti operazioni:

1. legge da tastiera la lunghezza `length` di un array, richiedendola ad oltranza se fosse inserita negativa;
2. crea un array `elements` di `length` interi;
3. chiama la funzione `init_random` per inizializzare `elements` a una permutazione casuale;
4. chiama la funzione `print` per stampare `elements`;
5. chiama la funzione `different` con l'array `elements` come parametro e stampa il suo risultato.