Secondo parziale di Programmazione I - Bioinformatica

20 giugno 2023 (tempo disponibile: 2 ore)

INDICAZIONI GENERALI

- Utilizzare il comando `ulimit -v 500000` per limitare l'utilizzo delle risorse al terminale su cui viene eseguito il comando ed evitare spiacevoli inconvenienti dovuti ad eccessive allocazioni di memoria.
- Scaricare il file di ogni esercizio e riconsegnarlo senza modificarne il nome.
- I file non consegnati o consegnati con errori di compilazione non verrano presi in considerazione.
- Si possono utilizzare funzioni aggiuntive non presenti nei file modello e aggiungere linee di commento alle funzioni giá implementate nel modello.
- I file possono essere consegnati piú volte. Per ogni esercizio, solo l'ultimo file consegnato sará considerato valido.
- Compilare con l'opzione `-Wall` per avere tutti i warning.
- È vietato utilizzare funzioni di libreria diverse da `scanf()`, `printf()`, `malloc()` e `sizeof()`.

Esercizio 1 (16 punti).
(si consegni es1_targhe.c)

Si vogliono trovare le targhe pari a partire da un insieme di targhe. Una targa valida e' una stringa alfanumerica di 7 caratteri, in particolare due alfabetici 3 numerici e 2 alfabetici (es. `AS234NN`). Si completi l'implementazione del codice allegato `es1_targhe.c`.

La funzione `leggiTarghe()` legge da stdin 10 targhe valide e le memorizza in una matrice di caratteri.

La funzione `trovaTarghePari()` riceve in input la matrice m di caratteri contenente le targhe e una matrice di caratteri vuota, e popolera' la matrice vuota con le targhe pari presenti in m, infine restituirà il numero di targhe pari trovate.

Completare il file `es1_targhe.c` che ha la seguente struttura.

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#define M 10
#define L 8//lunghezza standard di una targa


/*legge le targhe da stdin e le memorizza in una matrice di caratteri*/
void leggiTarghe(char [][L] );
/*memorizza le targhe pari in una matrice di caratteri e resituisce il
   numero di targhe pari*/
int trovaTarghePari(char [][L], char [][L] );
int main(void){
  char targhe[M][L];
  leggiTarghe(targhe);

  char targhePari[M][L];
  int k = trovaTarghePari(targhe,targhePari);
  printf("Le seguenti targhe sono pari:\n");
  for (int i=0;i<k;i++){
    printf("%s\n",targhePari[i]);
  }
  return -1;
}
void leggiTarghe(char targhe[M][L] ){
   // da completare
}

int trovaTarghePari(char targhe[][L], char targhePari [][L] ){
   // da completare
}
```

Esercizio 2 (14 punti)
(si consegni `es2_elim.c`)

Completare il file `es2_elim.c` che ha la seguente struttura

```
#include <stdio.h>
#include <stdlib.h>
//definizione del tipo di dato per gli elementi di una lista
typedef struct elem {
  // da completare
} Elem;

Elem* crealista(int a[], int size);
void printList(Elem* head);

int main(void) {
  int arr[] = {2,4,4,4,4,6,6,8,8,8,10,10,10};
  int size = sizeof(arr) / sizeof(int);
  Elem* result = crealista(arr, size);
  printf("Lista senza duplicati:\n ");
  printList(result);
  return 0;
}
//funzione che dato un array e la sua lunghezza restituisce
//il puntatore al primo elemento della lista
Elem* crealista(int a[], int size) {
  // da completare
}

//funzione che dato il puntatore di testa ad una lista stampa
//tutti gli elementi della lista
void printList(Elem* head) {
  // da completare
}
```

in modo che, dato un array di n interi ordinati in modo non decrescente, si trasferiscano tutti gli elementi dell'array senza duplicazioni in una lista di interi.

Il `main()`, dato l'array
`int arr[] = 2,2,2,2,2,2,2,2,4,4,4,4,6,6,8,8,8,10,10,10;`
dovrà stampare
`2 4 6 8 10`