Esercitazione 7

Argomento: Ricorsione su vettori, stringhe e SCL

1. Scaricare i file per l'esercitazione
- Scaricare la cartella Esercitazione_7 contenente i file dell'esercitazione

2. Modificare il file functions.c
Le funzioni devono essere implementate in maniera ricorsiva.

3. Compilare ed eseguire il programma
- comando per la compilazione:

```
make
```

- esecuzione del programma principale:

```
./e7_test
```

Esercizio 7.1

Scrivere la funzione C

```
bool tuttiMinuscoli(char* s)
```

che, data in ingresso una stringa, restituisce 1 se la stringa contiene solo caratteri minuscoli, 0 altrimenti.

Esercizio 7.2

Scrivere la funzione C

```
void converti(char* s);
```

che converte tutti i caratteri minuscoli presenti nella stringa di input s nei corrispondenti caratteri maiuscoli.

Esercizio 7.3

Scrivere la funzione C

```
int contaParentesi(char *s);
```

che, data in ingresso una stringa, calcoli e restituisca il numero di parentesi presenti nella stringa. Considerare come parentesi i caratteri: `([{}])`.

Esercizio 7.4

Scrivere la funzione C

```
void concatenate(char* dest, char* src);
```

che, date in ingresso due stringhe, concateni la stringa `src` a `dest` e la memorizzi in `dest`.

Esercizio 7.5

Scrivere la funzione C

```
int prodotto(int a[], int n);
```

che, dati in ingresso un array `a` e la sua lunghezza `n`, calcoli e restituisca il prodotto degli elementi di `a`. (la ricorsione va eseguita sia su `a` ed `n`).

Esercizio 7.6

Sia data la seguente struttura collegata `TipoSCL`:

```
typedef float TipoInfoSCL;
struct ElemSCL {
  TipoInfoSCL info ;
  struct ElemSCL* next;
};
typedef struct ElemSCL TipoNodoSCL;
typedef TipoNodoSCL * TipoSCL;
```

Implementare la funzione:

```
float SCL_media(TipoSCL head_ptr);
```

che restituisce la media degli elementi della lista, il cui puntatore di testa e' `head_ptr`.
Eseguire l'esercizio implementando due funzioni ausiliare ricorsive: una che calcola la lunghezza della lista e l'altra che calcola la somma dei suoi elementi.

Esercizio 7.7

Implementare la funzione:

```
void SCL_integral(TipoSCL head_ptr);
```

che scrive, in ogni elemento della lista (il cui puntatore di testa e' `head_ptr`) la somma degli elementi precedenti (dal primo a se stesso).

Esercizio 7.8

Implementare la funzione:

```
float SCL_dot(TipoSCL head1_ptr, TipoSCL head2_ptr);
```

che ritorna il prodotto scalare, delle liste con puntatori alla testa `head1_ptr` e `head2_ptr`.