Esercizio 2

Scrivere la dichiarazione e la definizione di una funzione ricorsiva compute sum, che prende come argomento un intero positivo num e restituisce:

- la somma delle cifre del numero se questa somma è minore di 10
- il risultato della riapplicazione della funzione sulla somma calcolata addizionata di 10 se la somma calcolata è pari
- il risultato della riapplicazione della funzione sulla somma calcolata addizionata di 1 negli altri casi

Per evitare ambiguità, riportiamo un esempio di applicazione delle regole sopra specificato.

Sia `sum` una funzione che calcola la somma delle cifre che compongono il numero.
Allora

1. `sum(99019) = 28` che è maggiore di 10 e pari, quindi sommo 10
2. `sum(38) = 11` che è maggiore di 10 e dispari, quindi sommo 1
3. `sum(12) = 3`, minore di 10 e quindi `compute_sum(99019) = 3`

La funzione `compute_sum` deve essere ricorsiva e non deve contenere iteratori espliciti (**`for`**, **`while`**, **`do-while`**). Sono consentite (se ritenute necessarie) chiamate a funzioni ricorsive ausiliarie che a loro volta non contengano iterazioni esplicite. La funzione è inserita in un semplice programma che legge un intero positivo come unico argomento a linea di comando, lo converte in intero, chiama la funzione `compute_sum`, stampa a video il numero specificato ed il risultato della chiamata a `compute_sum`. Alcuni esempi di esecuzione sono i seguenti:

```
$ ./a.out 38
The initial integer is: 38
The value of compute_sum(38) = 3

$ ./a.out 99019
The initial integer is: 99019
The value of compute_sum(99019) = 3

$ ./a.out 91019
The initial integer is: 91019
The value of compute_sum(91019) = 3

$ ./a.out 191019
The initial integer is: 191019
The value of compute_sum(191019) = 4
```