23 settembre – Fondamenti di Informatica – Seconda Parte – 120 minuti
COMPITO A – Esercizi di Programmazione (24.5 punti)

Esercizio A1 (9 punti): Stringhe

Realizzare un'applicazione che consiste di due funzioni.

(6pt) Una funzione ***nascondiNumeri*** riceve come parametro una stringa e la modifica sostituendo tutti i numeri presenti nella stringa (ovvero tutte le sottostringhe massimali di caratteri numerici) con il carattere **'#'**. Ad esempio, la stringa **"il signor rossi nacque il 17/09/1990 e abita in via dei platani 165"** diventa **"il signor rossi nacque il #/#/# e abita in via dei platani #"**, in quanto le sottostringhe di caratteri numerici **"17"**, **"09"** e **"1990"** vengono ciascuna sostituita da un carattere **'#'**. Come altri esempi, la stringa **"buona notte"** rimane **"buona notte"**, mentre la stringa **"12345"** diventa **"#"**.

(3pt) Una funzione ***main*** gestisce l'interazione con l'utente. La funzione ***main*** deve:

- chiedere all'utente se vuole introdurre una stringa, oppure recuperare una stringa da un file
- nel caso in cui l'utente voglia introdurre una stringa, la funzione ***main*** chiede all'utente di inserirla e la memorizza all'interno di un array di 50 caratteri e su di un file "stringhe.txt". La funzione ***main*** invoca quindi la funzione ***nascondiNumeri*** fornendole come parametro la stringa letta; la funzione ***main*** stampa quindi la stringa modificata.
- nel caso in cui l'utente voglia recuperare una stringa da file, la funzione ***main*** legge una stringa da un file "stringhe.txt" e la memorizza all'interno di un array di 50 caratteri (stampando opportuni messaggi nel caso in cui il file non possa essere aperto, oppure non contenga alcuna stringa). Se ha effettivamente recuperato una stringa, la funzione ***main*** invoca la funzione ***nascondiNumeri*** fornendole come parametro la stringa recuperata; la funzione ***main*** stampa quindi la stringa modificata.

Esercizio A2 (5 punti): Ricorsione

Realizzare un'applicazione ***TreDivisibiliTre*** definita come segue.

(4pt) L'applicazione contiene una funzione **ricorsiva** ***treDivisibiliTre*** che verifica, all'interno di un array di interi positivi ricevuto come parametro se, per ogni terna di interi consecutivi della sequenza, sia la loro somma che il loro prodotto sia un numero divisibile per tre. Ad esempio, se l'array è **[5, 6, 1, 2, 3]**, la funzione deve restituire **1**, in quanto nella tripla **[5,6,1]** la somma è 12 e il prodotto è 30 ed entrambi sono divisibili per tre; nella tripla **[6,1,2]** la somma è 9 e il prodotto 12; e nella tripla **[1,2,3]** la somma è 6 e il prodotto è 6. Se l'array è **[1, 4, 1, 5, 6]**, la funzione deve restituire **0**, poiché nella tripla **[1,4,1]** la somma è 6, ma il prodotto è 4, che non è divisibile per tre.

La funzione ***treDivisibiliTre*** ha due parametri: l'array e la sua lunghezza. Nel caso in cui si desideri utilizzare un terzo parametro per realizzare la ricorsione, deve essere definita un'ulteriore funzione ***treDivisibiliTreRic*** che ha tre parametri e che realizza la ricorsione. In tal caso la funzione ***treDivisibiliTre*** invoca la funzione ***treDivisibiliTreRic*** (fornendole opportuni parametri) per calcolare il risultato.

(1pt) In un commento che precede la funzione ***treDivisibiliTre*** descrivere la specifica della funzione ***treDivisibiliTre*** (espressa come Input-Precondizione-Output-Postcondizione).

(0pt) L'applicazione contiene una funzione ***main*** che gestisce l'interazione con l'utente. La funzione main è implementata (a meno dei messaggi finali per l'utente) in un file ***ricorsione.c*** il cui codice può essere scaricato e copiato da moodle.

Esercizio A3 (10.5 punti): Liste

Realizzare un'applicazione ***video*** per descrivere una lista di video digitali.

(1pt) Definire tre strutture, una per rappresentare la durata del video, una per rappresentare un video, ed una per rappresentare un elemento della lista di video. L'applicazione deve gestire la durata di un video come una struttura con tre campi, che rappresentano ore, minuti e secondi. L'applicazione deve gestire ciascun video come una struttura con due campi, che rappresentano il nome del video e la sua durata.

(1pt) Definire una funzione che visualizza la lista, stampando per ciascun video le informazioni ad esso associate.

(3.5pt) Definire una funzione che inserisce un nuovo video in coda alla lista, dopo aver ricevuto le informazioni relative allo stesso dall'utente.

(5pt) Definire una funzione che, data la lista di video, cancella dalla lista il video più lungo.

Definire una funzione main che crea la lista e gestisce l'interazione con l'utente. La funzione main è già parzialmente implementata in un file **liste.c** che può essere scaricato da Moodle.