Capitolo 6

Lezione 5 aprile 2019
Esercitazione

6.1 Modalità di consegna e valutazione esercitazioni

V. Esercitazione 1 e 2.

Date di assegnazione e consegna esercitazione 3 5 aprile → 11 aprile

Note aggiuntive Le funzioni consegnate non devono produrre nessun output. Nel caso siano presenti istruzioni che producono output (es. utilizzando printf), toglietele/commentatele prima di consegnare poiché possono allungare il tempo di esecuzione ed interferire con il processo di correzione. Il programma deve compilare senza errori o warning con le opzioni `-g3 -Og -ansi -pedantic-errors -Wall -Wextra`.

ATTENZIONE, NON UTILIZZARE LE PAROLE FOR e WHILE IN NESSUNA PARTE DEL CODICE, NEPPURE NEI COMMENTI (in caso contrario l'esercizio sarà considerato nullo)

6.2 Esercitazione 3 - Esercizio A

Scrivere una funzione C che, data una stringa di lunghezza arbitraria, utilizzi la ricorsione per determinare se è palindroma non tenendo conto degli spazi. La funzione deve ritornare 1 in caso affermativo, 0 altrimenti. Non è consentito utilizzare funzioni di libreria ad eccezione di `strlen()`. Se necessario è possibile utilizzare funzioni ausiliarie e se lo si ritiene utile la ricorsione può avvenire all'interno delle funzioni ausiliarie.

Consegna Il prototipo della funzione **DEVE** essere

```
int palindroma_ns (char *s)
```

Esempio Le stringhe seguenti, ignorando gli spazi, sono tutte palindrome:

```
char a[] = "anna";
char b[] = "    a  n    n  a    ";
char a[] = "  ara";
char a[] = "a    ra    ";
```

6.3 Esercitazione 3 - Esercizio B

Scrivere una funzione C che, dati una numero intero ed un array di interi rappresentanti dei tagli di moneta, utilizzi la ricorsione per determinare in quanti modi è possibile ottenere l'importo specificato utilizzando le banconore

indicate. La funzione deve ritornare il numero calcolato oppure -1 nel caso non sia possibile. Non è consentito utilizzare funzioni di libreria. Se necessario è possibile utilizzare funzioni ausiliarie e se lo si ritiene utile la ricorsione può avvenire all'interno delle funzioni ausiliarie.

Consegna Il prototipo della funzione **DEVE** essere

```
int change(int coins[], int size, int n)
```

Esempio Il frammento di codice seguente

```
int arr[] = {1, 2, 3};
printf("%d\n", change(arr, 3, 4));
```

stamperà 4, che corrisponde alle seguenti possibilità:
$4 = 1 + 1 + 1 + 1$
$4 = 1 + 1 + 2$
$4 = 1 + 3$
$4 = 2 + 2$

6.4 Esercitazione 3 - Esercizio C

Scrivere una funzione C che risolva il problema dello zaino 0/1 (v. paragrafo successivo) utilizzando la ricorsione. La funzione deve ritornare il valore totale di tali oggetti. Non è consentito utilizzare funzioni di libreria. Se necessario è possibile utilizzare funzioni ausiliarie e se lo si ritiene utile la ricorsione può avvenire all'interno delle funzioni ausiliarie.

Il problema dello zaino 0/1 Il problema dello zaino 0/1 consiste nello scegliere da un insieme di oggetti di valore e peso noti il sottoinsieme di oggetti di valore massimo avente peso complessivo inferiore ad una soglia data.

Consegna Il prototipo della funzione **DEVE** essere

```
int zaino(int valore[], int peso[], int n, int peso_massimo)
/*
int valore[], peso[]: valori e pesi degli oggetti
int n: numero di oggetti (lunghezza degli array peso, valore e selezione)
int peso_massimo: limite di peso (capacità dello zaino)
*/
```

Esempio Dopo l'esecuzione del codice seguente:

```
int peso[] = {2,3,3,4},
    valore[] = {1,2,5,9},
    peso_massimo = 7,
    valore_massimo, n;

n = sizeof(peso)/sizeof(peso[0]);
valore_massimo = zaino(valore, peso, n, peso_massimo);
```

`valore_massimo` avrà valore 14

6.5 Esercitazione 3 - Esercizio D

Scrivere una funzione C che, dato un array di interi, utilizzi la ricorsione per individuare la più lunga sequenza decrescente (non strettamente, A[i]≥A[i+1]). La funzione deve ritornare l'indice della prima posizione della sequenza individuata e modificare una variabile esterna contenente la lunghezza (parametro di tipo puntatore ad intero). Nel caso siano presenti più sequenze descrescenti con la medesima lunghezza, ritornare l'indice minore. Se necessario è possibile utilizzare funzioni ausiliarie e se lo si ritiene utile la ricorsione può avvenire all'interno delle funzioni ausiliarie.

Note Questo è un esercizio aggiuntivo: **il numero totale di punti da raggiungere per superare le esercitazioni rimane 9**.

Consegna Il prototipo della funzione **DEVE** essere

```
int sequenza_decrescente_massima (int A[], int *lung) {
```

Esempio Nell'array

```
int A[] = { 5, 4, 10, 12, 10, 10, 4, 3, 4, 3, 2, 1, 0};
```

la sequenza decrescente di lunghezza massima ha lunghezza 5 e l'indice della sua prima posizione è 3 (quarta posizione, valore 12).