Esercitazione 4

Argomento: Array

Scaricare il file *es4_array.c*. Aggiungere in *_es4_array.c* la definizione delle funzioni indicate negli esercizi seguenti. Modificare opportunamente la funzione main per effettuare delle verifiche di funzionamento delle funzioni scritte. Per testare le funzioni bisognerà utilizzare degli array generati randomicamente.

Altri Esercizi Proposti

Esercizio 4.1

Scrivere la funzione C

```
void vec_scale(double v[ ], int dim, double d);
```

che dato in ingresso un vettore v di dimensione dim, modifichi v scalando le sue componenti di un fattore d. Si ricorda che scalare un vettore di un fattore d significa moltiplicare tutte le sue componenti per il valore d.

Esercizio 4.2

Scrivere la funzione C

```
double vec_dot(double src1[ ], double src2[ ], int dim);
```

che, dati in ingresso due vettori della stessa dimenzione dim, ritorni il loro prodotto scalare. Si ricorda che il prodotto scalare tra due vettore è uguale alla somma dei prodotti delle componenti dei due vettori:

```
(x_1 * y_1) + (x_2 * y_2) + … + (x_dim * y_dim)
```

Esercizio 4.3

Scrivere la funzione C

```
double* vec_clone(double v[ ], int dim);
```

che, dato in ingresso un vettore v di dimensioni dim, allochi e restituisca una copia del vettore v.

Esercizio 4.4

Scrivere la funzione C

```
bool vec_positive_check(double v[ ], int dim);
```

che, dato in ingresso un array v di dimensioni dim, restituisca in output un booleano. Il booleano sarà true se tutti i valori contenuti nell'array di input sono positivi, altrimenti sarà false.

Argomento: Stringhe

Scaricare il file *es4_string.c*. Aggiungere in *es4_string.c* la definizione delle funzioni indicate negli esercizi seguenti. Modificare opportunamente la funzione main per effettuare delle verifiche di funzionamento delle funzioni scritte.

Nota: In tutte le funzioni che ritornano una stringa, la stringa deve essere allocata dinamicamente.

Esercizio 4.5

Scrivere la funzione C

```
char* copia(char s[ ], int N);
```

che, date in input la stringa s e un intero N, restituisca in output una stringa contenente tutti i caratteri di s fino all'intero N. Se N eccede la lunghezza della stringa, la funzione restituisce l'intera stringa s.

Esercizio 4.6

Scrivere la funzione C

```
void seleziona_alcuni_char(char s[ ], int idxs[ ], int dim);
```

che, data in input la stringa s e un array di indici idxs con la sua dimensione dim, stampi a schermo i caratteri corrispondenti agli indici. (Nota: la funzione deve controllare che gli indici siano contenuti in s).

Esercizio 4.7

Scrivere la funzione C

`char* copia_con_eliminazione(char s1[ ], char s2[ ]);`

che, date in input due stringhe `s1` ed `s2`, crei e restituisca una stringa con i caratteri di s1 presenti in s2. Se il carattere invece non è presente viene sostituito dal carattere spazio (‘ ‘).

Esercizio 4.8

Scrivere la funzione C

`char* inverti(char s[ ]);`

che, data in input una stringa `s` restituisca in output la stringa s con i caratteri invertiti.

Altri Esercizi Proposti

Esercizio 4.9

Scrivere la funzione C

`void print_vocali(char s[ ]);`

che, data in input una stringa `s`, stampi a schermo solamente le vocali contenute in s.

Esercizio 4.10

Scrivere la funzione C

`char* sostituisci_carattere(char s[ ], char c1, char c2);`

che, dati in input una stringa `s`, e due caratteri `c1` e `c2`, crei e restituisca in output una stringa contenente tutti i caratteri di s1 nella quale il carattere c1 deve essere sostituito dal carattere c2.

Esercizio 4.11

Scrivere la funzione C

`bool check_minuscole(char s[ ]);`

che, data in input una stringa `s`, restituisca in output un booleano. Il booleano sarà true se tutte i caratteri della stringa sono minuscoli, altrimenti restituirà false.

Esercizio 4.12

Scrivere la funzione C

`char* sostituisci_maiuscole(char s[ ]);`

che, dati in input una stringa `s`, costruisca e restituisca in output una nuova stringa in cui il primo carattere di ciascuna parola nella stringa di partenza è stato reso maiuscolo. Tutti gli altri caratteri devono essere resi minuscoli. Esempio:

`input: “sTUDiare TdP mI piACe”`
`output: “Studiare Tdp Mi Piace”`

per controllare che un carattere c sia maiuscolo si può effettuare il seguente test:

`(c > ‘A’ && c < ‘Z’)` (lo stesso vale per controllare che un carattere sia minuscolo

per convertire un carattere c da maiuscolo a minuscolo si può procedere nel seguente modo:

`c - ‘A’ + ‘a’` (invece, per convertire invece da minuscolo a maiuscolo: `c - ‘a’ + ‘A’`)

Argomento: Matrici e File

Scaricare il file es4_mat.c e i file di esempio per le matrici mat_1.txt, mat_2.txt, mat_3.txt e mat4.txt. Aggiungere nel file es4_mat.c la definizione delle funzioni indicate negli esercizi seguenti. Modificare opportunamente la funzione main per effettuare delle verifiche di funzionamento delle funzioni scritte. Modificare o aggiungere altri file di matrici per ulteriori verifiche.

Esercizio 4.13 (5.1)

Scrivere la funzione C

`Mat* Mat_alloc(int rows, int cols);`

che, dato in ingresso il numero di righe rows ed il numero di colonne cols, allochi e restituisca una struttura Mat contenente una matrice di dimensione rows x cols. La matrice deve essere memorizzata come array di puntatori alle righe della matrice stessa.

Esercizio 4.14 (5.2)

Scrivere la funzione C

```
Mat* Mat_read(const char *filename);
```

che, dato in ingresso il nome di un file filename, allochi e restituisca una struttura Mat contenente una matrice letta dal file filename. Il file contiene un primo numero che indica il numero di righe ed un secondo che indica il numero di colonne della matrice, seguiti dalla lista di elementi che la compongono. Per esempio il file contenente la matrice

```
m =

[1.1 2.2 3.3]

[4.4 5.5 6.6]
```

avra' la seguente forma:

```
2 3

1.1 2.2 3.3

4.4 5.5 6.6
```