## Tecniche di Programmazione (2018/19)

# 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.

## Esercizio 4.1

Scrivere la funzione C

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

che, dato in ingresso un vettore v, di dimensione dim, stampi il vettore nel seguente formato:

```
[x_1  x_2  …  x_i  …  x_dim]
```

## Esercizio 4.2

Allocare due vettori

```
v2 = [2.1, -3.5, 1.0, 6.5, -5.2]
v3 = [4.8, 0.1, -6.2, -2.5, 7.2]
```

Allocando il primo in maniera statica ed il secondo in maniera dinamica. Calcolare inoltre la dimensione del vettore v2 utilizzando la funzione sizeof()

## Esercizio 4.3

Scrivere la funzione C

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

che, dato in ingresso un vettore v di dimensioni dim, allochi e restituisca un vettore v dove ogni elemento è la somma di tutti i valori a lui successivi.

Esempio:

v = [ 1.0 -0.1 3.4 7.2 -5.3];
output = [ 6.2 5.2 5.3  1.9 -5.3 ]

# Esercizio 4.4

Scrivere la funzione C

double* vec_rec(double v1[ ], double v2[ ], int dim1, int dim2);

che, dato in ingresso due vettori v1 e v2 e le loro rispettive dimensioni dim1 e dim2, allochi e restituisca un vettore v di dimensioni dim1 dove l'i-esimo elemento sarà quello del vettore v1 se ne esiste una ricorrenza nel vettore v2, altrimenti 0.

v1 = [ 1.0 -0.1 3.4 7.2 -5.3];
v2 = [ 1.0 2.5 7.2 ];
output = [ 1.0 0 0 7.2 0 ];

# Altri Esercizi Proposti

# Esercizio 4.5

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 4.6

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 4.7

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 4.8

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 4.1

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 4.2

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 4.3

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 4.4

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 4.5

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 4.6

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 4.7

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 4.8

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’`)