Es. 6.1 – Massimo in vettore

Scrivere una funzione massimo(…) che dato un vettore di double in ingresso, restituisce il valore massimo presente nel vettore.
Scrivere un programma che dato il vettore di double {1.3, 4.5, 2.4, 8.4, -3.14, -3.14} ne calcola il massimo con la funzione massimo(…) e lo stampa a video.

Output di esempio:

```
Il valore massimo presente nel vettore e': 8.4
```

Es. 6.2 – Posizione minimo in vettore (per casa)

Scrivere una funzione int posMinimo(const double vett[], int dim) che dato un vettore di double in ingresso, restituisce la posizione dell'elemento minimo. Se ci sono più minimi uguali, restituisce la posizione del primo.
Scrivere un programma che dato il vettore di double {1.3, 4.5, 2.4, 8.4, -3.14, -3.14} ne calcoli la posizione del minimo con la funzione posMinimo(…) e la stampi a video.

Output di esempio:

```
La posizione del primo minimo all'interno del vettore e': 4
```

Es. 6.3 – Somma pari di vettore (per casa)

Scrivere una funzione bool pari(const int v[], int n) che prende in ingresso un vettore di interi e restituire vero se la somma degli elementi del vettore è pari, falso altrimenti.
Scrivere un programma che crea un vettore di 10 interi, chiede all'utente di digitare da tastiera 10 interi e li inserisce nelle 10 locazioni del vettore; successivamente invoca la funzione e scrive a video "PARI" se la somma degli elementi dell'array è pari, "DISPARI" altrimenti.

Output di esempio:

```
Inserisci 10 elementi dell'array:
1
2
3
3
3
4
8
9
12
11

La somma degli interi inseriti e' PARI.
```

Es. 6.4 – Concatena vettori

Scrivere una funzione void concatena(const int* v1, const int* v2, int n1, int n2, int* v3) che prende in ingresso i vettori v1 e v2 di dimensioni rispettivamente n1 ed n2, e restituisce mediante l'argomento v3 il vettore risultante dalla loro concatenazione. Si assuma che il vettore v3 sia stato correttamente allocato della dimensione opportuna dal chiamante.
Scrivere un programma che chiama la funzione concatena(…) sui vettori {2, 6} e {3, 15, 4} e stampi il vettore risultante.

Output di esempio:

```
v3: 2 6 3 15 4
```

Es. 6.5 – Stringa palindroma

Scrivere la funzione bool palindroma(const char stringa[]) che restituisce true se la C-stringa passata come parametro è palindroma (cioè se è identica sia che venga letta da sinistra a destra che da destra a sinistra), false altrimenti. Esempi di C-stringhe palindrome sono "ingegni", "otto", "radar". Si consideri le lettere maiuscole diverse dalle minuscole, quindi per esempio 'a' è diversa da 'A'.

Scrivere un programma che:

- legge da tastiera una parola di al più 40 caratteri,
- richiama la funzione palindroma(…) su di essa,
- stampa a video se la parola è palindroma oppure no.

Output di esempio:

```
Inserisci una parola:
radar
La parola e' palindroma
```

```
Inserisci una parola:
Anna
La parola NON e' palindroma
```

Es. 6.6 – Conta parole in una C-stringa

Scrivere la funzione int contaParole(const char stringa[]) che restituisce il numero di parole presenti nella C-stringa. Si consideri come separatore di due parole una sequenza di uno o più caratteri spazio.
Ad esempio la seguente C-stringa ha 4 parole: "`Il tempo e' bello`".

Scrivere un programma che chiama la funzione contaParole(…) sulla C-stringa:
"`Lorem ipsum  dolor  sit amet  `" (notare gli spazi multipli alla fine e tra le parole)
e poi sulla C-stringa:
"`  Lorem ipsum  dolor  sit amet  `" (notare gli spazi multipli all'inizio)
e stampa a video entrambi i risultati.

Output di esempio:

```
Numero di parole della prima C-stringa: 5
Numero di parole della seconda C-stringa: 5
```

Es. 6.7 – Concatena stringhe (per casa)

Scrivere una funzione my_strcat(…), che riceve in ingresso una stringa destinazione ed una sorgente, e che modifica la stringa destinazione concatenandovi la stringa sorgente. Esempio: Nel caso in cui la stringa destinazione e quella sorgente abbiano i seguenti valori:

dest = "123"
sorg = "45678"

dopo la chiamata della funzione la stringa destinazione deve contenere "12345678". Si assuma che la stringa destinazione sia stata correttamente allocata dal chiamante della dimensione opportuna a contenere la stringa finale. Per realizzare la funzione my_strcat(…) è possibile utilizzare la funzione di libreria strlen(…), ma non (ovviamente) la strcat(…).

Scrivere un programma che definisce una stringa destinazione pari a "Fondamenti di ", una stringa sorgente pari a "Programmazione", chiama la funzione my_strcat(…) e stampa a video il nuovo valore della stringa destinazione.

Output di esempio:

```
Contenuto C-stringa dest prima della concatenazione: "Fondamenti di "
Contenuto C-stringa sorg: "Programmazione"
Il nuovo contenuto della C-Stringa dest, dopo la concatenazione, e':
"Fondamenti di Programmazione"
```