LEGGETE LA GUIDA PER LA CREAZIONE DEI PROGETTI E PER IL DEBUGGING!

Gli esercizi seguenti devono essere risolti, compilati e testati utilizzando il debugger. Per ognuno si deve realizzare una funzione main() che ne testi il funzionamento. ***Fate progetti diversi per ogni esercizio.***

Esercizio 1

Creare un file `main.c`. Nel file, si scriva:

```
01    #include <stdlib.h>
02
03    int main(void)
04    {
05          double a[] = { 2, 2, 2, 2, 2, 2, 2, 2, 2, 2 };
06          size_t i, n = 10;
07
08          for (i = 0; i < n; ++i) {
09                double d = a[i];
10                potenza(&d, i);
11                a[i] = d;
12          }
13          return 0;
14    }
```

Aggiungere sopra al `main()` la definizione della funzione `potenza()` che eleva `d` alla i e mette il risultato di nuovo in d. La funzione deve essere fatta in modo da non modificare il `main()` fornito.

Esercizio 2

Una volta risolto l'esercizio precedente, senza modificare la funzione `potenza()`, eliminare le righe 09 e 11 (indicate qui sopra prima di inserire la funzione potenza). Modificare quindi solo la riga 10 in modo che il programma produca lo stesso risultato.

Esercizio 3 (bonus)

Una volta risolto l'esercizio precedente, modificare **solo** la riga 10 in modo che il programma produca lo stesso risultato **senza utilizzare l'operatore &**.

Esercizio 4

Nel file `terna.c` definire la funzione corrispondente alla seguente dichiarazione:

```
extern bool is_terna_pitagorica(unsigned int a, unsigned int b, unsigned int c);
```

La funzione accetta 3 numeri interi senza segno e deve verificare se questi formano una terna pitagorica, ovvero se la somma dei quadrati di due di questi è pari al quadrato del numero rimanente. Ad esempio la terna 5,4,3 è una terna pitagorica, come anche 5,3,4, mentre 1,2,3 no.

Attenzione al tipo di ritorno. Includere l'header della libreria necessaria.

Esercizio 5

Creare un file `lungh.c`. Nel file, si realizzi in linguaggio C la funzione corrispondente alla seguente dichiarazione:

```
extern size_t lungh (const char *str);
```

La funzione riceve un puntatore a un vettore di char zero terminato (stringa C) e restituisce il numero di caratteri contenuti (senza il terminatore). Un esempio di chiamata è il seguente:

```
int main (void) {
	char s[] = "Ecco la stringa di prova";
	size_t len;

	len = lungh(s);

	return 0;
}
```

In questo caso `len` varrà 24.

Inserite la funzione `main()` in un file `main.c`, ovvero non nello stesso file che contiene la definizione di `lung()`. Non create un file `lung.h`.

Esercizio 6

Nel file `contaspazi.c` implementare la definizione della funzione:

```
unsigned int conta_spazi (const char* s);
```

La funzione accetta come parametro un puntatore a un vettore di `char` zero terminato (stringa C) e deve restituire il numero di caratteri <spazio> presenti nella stessa.
Ad esempio, data la stringa “prova stringa in cui contare gli spazi” la funzione deve ritornare il valore 6.

Inserite la funzione `main()` in un file `main.c`, ovvero non nello stesso file che contiene la definizione di `conta_spazi()`. Non create un file `contaspazi.h`.

Esercizio 7

Creare i file `cerca.h` e `cerca.c` che consentano di utilizzare la seguente funzione:

```
extern int cerca_primo (const char *s, char c);
```

La funzione accetta come parametro un puntatore a un vettore di char zero terminato (stringa C) in cui cercare la prima occorrenza del carattere passato col parametro `c`. La funzione restituisce l’indice di `c` nella stringa `s` (come al solito, partendo da 0 per il primo carattere). Nel caso il carattere non sia presente deve ritornare -1. Ad esempio se cercassimo in `s=“aereo”` il carattere `c=‘e’` dovrebbe ritornare 1; se cercassimo il carattere `c=‘x’` dovrebbe ritornare -1.