A.A. 2017/2018 - Elaborato 1

Dato un array di puntatori a stringa (ogni cella del vettore contiene l'indirizzo a 32 bit del primo carattere della stringa), cercare la prima occorrenza di una sottostringa all'interno di ogni stringa del vettore.

Input: un array di puntatori (indirizzi di memoria a 32 bit), una DWORD (il numero di elementi nell'array) e un vettore di BYTE (la sottostringa da cercare)

Output: un array di DWORD signed (la posizione della sottostringa, 0=primo carattere, …; -1 se la sottostringa non è presente)

Esempi di casi importanti da verificare:

- Sottostringa in posizione zero
- Sottostringa alla fine della stringa
- Sottostringa in posizione intermedia
- Sottostringa di un carattere
- Sottostringa uguale alla stringa
- Sottostringa non presente
- Sottostringa più lunga della stringa

Scheletro da utilizzare per il programma:

```c
#include <stdio.h>

void main()
{
    // Input
    char* strings[] =   {
                            "Elaborato di Architetture","139",
                            "L'Arco del cerchio","L'arco di trionfo"
                        };                          //Array di puntatori a stringa
    int num = sizeof(strings) / sizeof(strings[0]); //Il numero di stringhe nell'array
    char subStr[] = "Arc";                          //La sottostringa da cercare

    // Output
    int posizioni[256]; //Posizioni in cui è stata trovata la sottostringa

    __asm
    {

    }

    // Stampa su video
    {
        int i;
        for (i = 0; i<num; i++)
        {
            printf("Sottostringa in posizione=%d della stringa[%d]\n", posizioni[i], i);
        }
    }
}
```

Attenzione:

- Non cambiare il nome delle variabili del codice C scheletro degli elaborati (attenzione anche alle minuscole/maiuscole)
- Non aggiungere altre variabili C al programma, ma usare solo quelle presenti nel testo degli elaborati

A.A. 2017/2018 - Elaborato 2

Dato in input il livello massimo richiesto, restituire il rispettivo triangolo di Tartaglia memorizzando consecutivamente i vari livelli all'interno di un array monodimensionale.

Input: una DWORD (livello massimo richiesto)

Output: un array di DWORD (contenente il triangolo di Tartaglia).

Esempi di casi importanti da verificare:

Livello 0

Livello 1

Livello 14

Link utili:

http://it.wikipedia.org/wiki/Triangolo_di_Tartaglia

Scheletro da utilizzare per il programma:

```c
#include <stdio.h>

void main()
{
    // Input
    unsigned int livello = 8;       // livello massimo del triangolo di tartaglia da generare

    // Output
    unsigned int triangolo[1024];   // risultato: il vettore contiene i numeri del
                                    // triangolo dal livello 0 al più alto richiesto

    unsigned int i;
    unsigned int k = 0;

    __asm
    {

    }

    // Stampa su video
    printf("Triangolo di Tartaglia fino al livello %d\n", livello);
    for (i = 0; i <= livello; i++)
    {
        unsigned int j;
        for (j = 0; j <= i; j++)
            printf("%d ", triangolo[k++]);
        printf("\n");
    }
}
```

Attenzione:

- Non cambiare il nome delle variabili del codice C scheletro degli elaborati (attenzione anche alle minuscole/maiuscole)
- Non aggiungere altre variabili C al programma, ma usare solo quelle presenti nel testo degli elaborati

A.A. 2017/2018 - Elaborato 3

Dato un array di bit, restituire le posizioni dei soli bit uguali a 1.

Input: un array di BYTE da considerare come una sequenza di bit; una WORD (il numero totale di bit).

Output: un array di WORD in cui memorizzare le sole posizioni dei bit uguali a 1. Inserire -1 per indicare che le posizioni sono terminate (elemento terminante).

N.B.

- Viene specificata in input la lunghezza in *bit*: può quindi accadere che l'ultimo BYTE dell'array sia utilizzato solo in parte.
- I *bit* all'interno di ciascun BYTE devono essere analizzati dal meno significativo al più significativo.

Scheletro da utilizzare per il programma:

```c
#include <stdio.h>

void main()
{
    // Input
    unsigned char vet[] = { 0xAA,0xFC,0x09 };   //Array di byte
                                                //(da considerare come una sequenza di bit)
    unsigned short int len = 19;                //Lunghezza (numero di bit)

    // Output
    short posizioni[1024]; //Posizioni dei soli bit con valore 1

    __asm
    {

    }

    // Stampa su video
    {
        int i = 0;
        while (i <= len && posizioni[i] != -1)
        {
            printf("%d\n", posizioni[i++]);
        }
    }
}
```

Attenzione:

- Non cambiare il nome delle variabili del codice C scheletro degli elaborati (attenzione anche alle minuscole/maiuscole)
- Non aggiungere altre variabili C al programma, ma usare solo quelle presenti nel testo degli elaborati