Settimana 5 Esercizi per casa

Prima di affrontare l'esercizio 3, e' importante familiarizzarsi con la ricorsione svolgendo alcuni dei punto proposti all'esercizio 1.

Es1: Per familiarizzarsi con la ricorsione

Scegliere un paio degli esercizi seguenti. Per ciascun esercizio, saper rispondere alla domanda:
qual e' il caso base (il caso di terminazione)?
qual e' un problema piu' piccolo (e piu' piccolo in che senso?)

(a) Scrivere una funzione ricorsiva int somma (int n, int m) che restituisce la somma degli interi tra n a m (n e m compresi).

(b) Scrivere una funzione ricorsiva che ricevuto come parametro un carattere ch, decide se ch appare in una sequenza di caratteri (letta da cin e terminata da ';').

(c) Scrivere una funzione ricorsiva che moltiplica per due tutti i valori di un array di interi. Modificarla perche' moltiplichi per due solo i valori dispari.

Es 2. Preparazione alle liste.

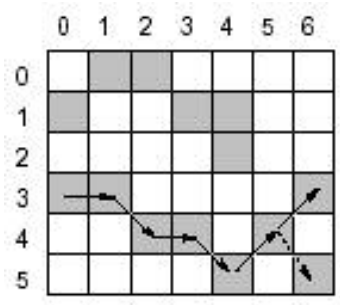
Creare una lista concatenata di 4 interi. Stamparla.

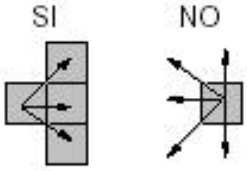
ES 3: Attraversamento di una palude

PROBLEMA:
Una palude è rappresentata da una mappa di RxC quadrati, ciascuno dei quali è un'area di terraferma o un'area di acqua non attraversabile. Vogliamo attraversare la palude dalla sponda sinistra a quella destra.

Si chiede di sviluppare un programma che, usando la mappa della palude, trova un cammino che la attraversa da sinistra a destra in modo che:

- il cammino passi solo su zone di terra, e
- per semplicità, ogni passo faccia avanzare il cammino verso la sponda destra





Un cammino attraverso la palude sarà un percorso tale che:

- inizierà da una zona della colonna 0
- terminerà in una zona della colonna 6
- ad ogni passo si sposta di una colonna in avanti

Nell' esempio in figura abbiamo:

colonna	0	1	2	3	4	5	6
riga	3	3	4	4	5	4	3

STRUTTURE DATI DA USARE:

- la mappa della palude sarà rappresentata da una matrice di dimensioni RxC i cui elementi saranno 1 (per un'area di terraferma) o 0 (per un'area di acqua, non attraversabile).
- Un cammino all'interno della palude sarà rappresentato da un array di C elementi, ognuno dei quali è un indice di riga (tra 0 e R-1). Il passaggio per la zona (i,j) sarà rappresentato memorizzando il valore i nella componente di indice j del vettore.

Nell'esempio sopra, il cammino risulta:

colonna	0	1	2	3	4	5	6
riga	3	3	4	4	5	4	3

REALIZZAZIONE DEL PROGRAMMA.
Nel seguito è data una parte del programma che cerca un attraversamento della palude. Si chiede di completare il programma scrivendo le definizioni delle due funzioni mancanti con le seguenti specifiche:

bool cerca_cammino (int palude[R][C], int i, int j, int cammino[C]);

Questa funzione cerca nella palude un cammino che inizia nella posizione (i,j) e arriva sulla sponda destra.
La funzione ritorna true se l'attraversamento è stato trovato, altrimenti ritorna false.
Il cammino trovato è memorizzato (e restituito al chiamante) nell'array passato come ultimo parametro.
Attenzione: il cammino costruito da questa funzione deve essere fatto di soli passi in avanti.
La funzione cerca_cammino **deve essere RICORSIVA**, cioè richiamare se stessa 'su un caso più semplice' (qual e' un problema piu' piccolo? Piu' piccolo in che senso?)

bool esplora_palude (int palude[R][C], int cammino[C]);

Questa funzione cerca un cammino che porta dalla sponda sinistra alla sponda destra della palude, invocando (in modo opportuno) la precedente funzione cerca_cammino.
Se un attraversamento è stato trovato, la funzione lo memorizza nell'array passato come secondo parametro e restituisce true, altrimenti restituisce false.

```
#include<iostream>
using namespace std;

const int R=5;
const int C=6;

bool cerca_cammino(int palude[R][C], int i, int j, int cammino[C]);

bool esplora_paude(int palude[R][C], int cammino[C]);

void stampa_cammino(int palude[R][C], int cammino[C]);


main(){
  int palude[R][C]={{1,0,0,1,0,0},    // mappa di palude
                    {1,0,0,0,0,0},
                    {0,1,0,0,0,1},
                    {0,0,1,1,1,0},
                    {0,1,0,0,0,0}};

  int cammino[C];           // array dove memorizzero' il cammino trovato


  if ( esplora_palude(palude,cammino) )
       stampa_cammino(palude, cammino);
  else
       cout << "Non ci sono cammini che attraversano la palude"<<endl;
}


void stampa_cammino(int palude[R][C], int cammino[C]){

  for(int i=0;i<C;i++)
       cout << cammino[i] << " ";
  cout << "\n\n";


 cout << "visualizzazione di cammino:";
 for(int i=0;i<R;i++){
       for(int j=0;j<C;j++){
            if(i==cammino[j])          // se il punto fa parte del cammino
                 cout << "* ";          // stampa *
            else                       // altrimenti
                 cout << "- ";          // stampa -
      }
       cout << endl;
}}
```