Esercizi 3AIIN 19.03.2021

- Salva in Moodle un file zip che contiene solo i pgm sorgenti.
- Formato nome file: **CognomeNomeGGMMAAAA.estensione**

dove GGMMAAAA è la data di assegnazione del lavoro.

- Terminata ogni lezione di laboratorio devi caricare in Moodle il lavoro svolto.
- A casa eventualmente puoi finirlo e/o correggerlo. Hai tempo una settimana.
- Anche chi è assente il giorno della consegna è tenuto a svolgere il lavoro assegnato.
- Verranno fatti controlli a campione e sicuramente durante le interrogazioni.

Puoi consultare il sito: **https://www.cplusplus.com/**

Esercizio n.1

Si considerino le seguenti dichiarazioni di tipo che rappresentano rispettivamente un punto nel piano ed un segmento nel piano (identificato dai suoi estremi).

```
typedef struct{
   double x;
   double y;
}punto;
```

```
struct segmento{
     punto p1;
     punto p2;
};
```

Diciamo che un segmento appartiene ad un quadrante (tra 1 e 4) se entrambi i suoi estremi sono contenuti nello stesso quadrante. Se gli estremi sono in quadranti diversi, un segmento non appartiene ad alcun quadrante. Si assuma che i punti non possano giacere sugli assi del piano.

Si scriva una funzione C++ che prenda come parametri un vettore di segmenti, il numero effettivo di segmenti memorizzati e restituisca un valore tra 1 e 4 che corrisponde al quadrante del piano a cui appartiene il maggior numero di segmenti.

Suggerimento: La funzione richiesta **maxSegmenti()** utilizza le funzioni **quadranteSegmento()** e **quadrantePunto()** (da codificare).
Utilizza il passaggio di parametri; i parametri di tipo record passali by reference (utilizza const se la direzionalità è solo di input).

E' inoltre richiesto una funzione main() di prova.

Alcune considerazioni:

- Una buona **modularizzazione** consiste nel definire una funzione, chiamata **quadranteSegmento()** che restituisce il quadrante del segmento, se esiste, e 0 se il segmento non appartiene ad alcun quadrante.
- Questa funzione, a sua volta, utilizza una funzione, chiamata **quadrantePunto()** che restituisce il quadrante di un singolo punto.
- Il risultato di **quadranteSegmento()** può essere utilizzato direttamente come indice di un vettore che conta il numero di segmenti per ciascun settore.
- I segmenti che non appartengono ad alcun settore vengono contati nella locazione 0.

```
#define MAX_SEGMENTI 6

typedef struct{
      double x;
      double y;
}punto;

struct segmento{
      punto p1;
      punto p2;
};

int main(){
struct segmento v[MAX_SEGMENTI];
int numSegmenti;

            …
}
```