Programmazione ad Oggetti - parte A: Esercitazione di laboratorio 3

Esercizio 0:

Scrivere una classe “mobile” con due dati membro privati di tipo int: “altezza” e “larghezza”. La classe mobile deve inoltre definire un costruttore per inizializzare a “0” i valori dei due dati membro, e deve contenere dei metodi pubblici “set” e “get” per impostare e leggere il valore dei due dati membro. Scrivere una classe “armadio” derivata da “mobile” con ereditarietà di tipo pubblico. La classe “armadio” deve aggiungere un dato membro privato di tipo int “numero_ante”, un costruttore per inizializzare a “0” il numero_ante, e i rispettivi metodi pubblici “get” e “set” per impostare e leggere il valore di numero_ante. Scrivere un programma di test per verificare il comportamento delle classi.

Esercizio 1:

Scrivere una funzione template `template <class T> T max2(T primo, T secondo)`

per calcolare il maggiore tra due elementi.

Scrivere una funzione template `template <class T> T max3(T primo, T secondo, T terzo)`

per calcolare il maggiore tra tre elementi, che utilizzi la funzione max2. Scrivere un programma di test per verificare il funzionamento delle funzioni realizzate su elementi di tipo int e su elementi di tipo float.

Esercizio 2:

Scrivere una funzione template: `template <typename T> T most_common(T *A, int size)`

che accetti in ingresso un array di elementi di tipo generico T e la sua dimensione “size” e fornisca in uscita l’elemento dell’array ripetuto più volte. Se vi sono due o più elementi con lo stesso numero di ripetizioni massimo la funzione restituisce il primo elemento trovato.

Esercizio 3:

Scrivere una classe template `template<class T> class Pair` i cui oggetti rappresentino coppie di elementi (“first” e “second”) dello stesso tipo generico T. La classe oltre ai costruttori deve contenere i seguenti metodi pubblici:

- `void set_element(int position, T value); // imposta value in posizione 1 o 2`
- `T get_element(int position) const; // restituisce l’elemento in posizione 1 o 2`
- `void add_up(const Pair<T>& the_pair); // somma gli elementi corrispondenti di un oggetto Pair con quelli contenuti nell’argomento the_pair`

Esercizio 4:

Scrivere una classe template `template <class T> class Matrix2x2` per rappresentare array a due dimensioni generici di dimensione 2x2 (matrici). La classe oltre ai costruttori deve contenere un metodo pubblico per sommare ad un oggetto `Matrix2x2` una matrice passata come argomento

```
Matrix2x2<T> Add(Matrix2x2 x);
```

(la somma di due matrici è la matrice i cui elementi sono la somma di elementi in posizione corrispondente)