Alessia Gualtieri, 16/7/2021

**ESERCITAZIONE 1 (ACADEMY)**

1. **Descrivere le differenze tra Reference Type e Value Type**

I Value Type sono tutti i tipi primitivi (escluse le stringhe), le loro particolarità consistono nel fatto che contengono direttamente il dato (salvato nello stack), una loro copia implica la copia dei dati in esso contenuto, una loro modifica ha effetto solo sull’istanza corrente e non accettano il valore null.

I Reference Type invece sono tutte le classi, a differenza dei Value Type contengono un riferimento all’oggetto, la loro copia implica la copia del riferimento e la loro modifica ha effetto su tutti gli oggetti che sono stati generati da una loro copia. Sono salvati nell’heap e accettano il valore null se non referenziano alcuna istanza.

1. **Cosa è un delegate?**

I delegate rappresentano puntatori a funzioni. Sono definiti con la keyword **delegate** e vengono definite dai parametri che deve avere la funzione puntata e dal valore ritornato.

Es: data la funzione somma come segue

Public int Sum (int a, int b)

{ return a+b; }

Posso definire il seguente delegate

delegate int DelegateSum (int a, int b);

Quindi istanziarlo e usarlo come segue

DelegateSum del = new DelegateSum(Sum);

del(1,3); //restituisce 1+3

1. **Definire come costruire il design pattern Factory e qual è la sua utilità**

Il Factory serve per istanziare una variabile di un certo tipo in base a certi parametri evitando che l’utente conosca la differenza dei diversi tipi (esempio: ho diversi tipi di Azienda in base al numero di dipendenti e all’utente non serve sapere la differenza di azienda).

Per costruire un design pattern Factory è necessario creare una interfaccia (IFactory) contenente tutti parametri comuni alle diverse classi, quindi creare le classi implementando tale interfaccia. Si crea infine una classe Factory che implementa un metodo che restituisce un oggetto di tipo IFactory e prende come argomento il paramentro rispetto cui è necessario istanziare tale oggetto come quella specifica classe. Esegue quindi uno studio del parametro, inizializza l’oggetto di tipo IFactory come oggetto della specifica classe e lo restituisce.