# Modulo 5

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | Nome | Elisa |
|  |  | Cognome | Posani |
|  |  | Data | 26-02-2021 |

Leggete attentamente ogni domanda e argomentare quanto più possibile **fornendo anche degli esempi**.  
ATTENZIONE: Le domande a risposta multipla possono contenere più risposte corrette.

1. *Spiegare il funzionamento di Enum*

Enum è un value type definite dall’utente che permette di generare delle enumerazioni: ad una parola, il nome, viene associato un numero intero, quindi il suo valore. Sarà poi possible risalire al valore a partire dal nome, o viceversa. E’ utile in particolare per associare degli attributi ad un numero intero, in modo che sia utilizzabile all’interno del codice in modo più agevole, o anche solo per avere una corrispondenza chiave – valore numerico.

1. *Spiegare la differenza tra Array, Collection e Collezioni Generiche*

Tutti questi elementi sono reference type e sono quindi allocate nel managed heap. Gli array sono vettori fortemente tipizzati e di lunghezza predefinita. I vari oggetti nel vettore potranno essere nulli. Per aggiungere un elemento ad un array pieno, bisogna effettuare un resize, andando a spostare il vettore nell’heap, allogandogli un maggiore spazio.

Le Collection e le Collection Generiche funzionano allo stesso modo e contengono liste, stack, queues, hashtable… Questi sono facilmente gestibili dal punto di vista del numero di elementi contenuti, in quanto vengono allocati separatamente. Per questo sono anche meno efficienti rispetto agli array, che occupano spazi in memoria vicini. Per aggiungere o eliminare un elemento si possono usare le funzionalità Add e Remove, senza dover effettuare un Resize come negli array.

Le Collections hanno tipizzazione debole, in quanto gli elementi contenuti possono essere di qualsiasi tipo, anche diversi tra loro. Le Collezioni Generiche sono invece più tipizzate perché, pur accettando tutti I tipi in input, come del resto gli array, questi dovranno essere tutti dello stesso tipo.

1. *Quanti valori di ritorno può avere un metodo? Commentare la risposta.*

Se si considera il numero di oggetti che possono ritornare da un metodo, questo può essere solo uno. Ciò non toglie che l’oggetto possa avere più elementi al suo interno, ad esempio un array, se sono tutti dello stesso tipo. Inoltre si può utilizzare una tupla in modo da avere un massimo di 8 valori che ritornano (estendibili mettendo al suo interno un’altra tupla). Un altro modo per avere più valori di ritorno può essere l’aggiunta di un output (out) che va specificato nei parametri del metodo. Infine, I diversi valori di ritorno si possono memorizzare come campi di una classe specifica, ritornando più valori.

1. *Spiegare la differenza tra classe, oggetto e tipo.*

Un tipo è una generale definizione di un oggetto: ad esempio un int è la rappresentazione di un numero intero. E’ caratterizzato da proprietà e metodi. Una classe è un tipo, però definite dall’utente. Si possono definire le proprietà ed i metodi associate. Infine un oggetto è tutto ciò che viene istanziato: quindi un oggetto avrà una classe (e quindi un tipo), ma anche un’identità propria poicheé distinto dagli altri oggetti della stesssa classe. Più oggetti possono essere della stessa classe e avere le stesse proprietà ma saranno sempre oggetti distinti.

1. *Descrivere la struttura di una classe*

Una classe è un oggetto creato dall’utente per relizzare l’incapsulamento all’interno del codice.

Questa potrà avere campi e proprietà, che possono essere read-only {get;}, write-only {set;}, o accessibili sia per scrittura che in lettura {get; set;}. Queste possono essere settate con il costruttore di default, che associa solo i valori di default, o avere uno o più costruttori (con l’overloading) definiti dall’utente.

All’interno della classe possono esserci metodi, cioè funzioni relative alla classe stessa. Questi potranno essere privati, se accessibili solo dalla classe stessa, o pubbliche se si possono utilizzare anche all’esterno. Inoltre possono essere d’istanza o statiche, se relative ad un oggetto istanziato o generali per la classe. Infine potranno avere parametri d’ingresso e valori di ritorno.

1. *Descrivere le differenze tra classe e struct*

La differenza principale tra classi e struct sta nel fatto che le prime sono Reference Type, mentre le seconde sono Value Type. Entrambe possono avere campi, proprità, metodi e costruttori, anche se le struct non supportano i costruttori di default.

1. *Definire cosa è un Generic e descriverne possibili utilizzi.*

Un Generic è un tipo non specificato che può essere utulizzato all’interno di procedure e classi. Questo permette l’utilizzo delle stesse da parte di più tipi, senza dover scrivere una funzione per tipo. Una volta definite il tipo generic T però questo deve essere lo stesso all’interno di tutta la procedura, quindi se la mia funzione prende in ingresso un parametro Funzione(T parametroEsempio) e all’interno della funzione ho alter variabili di tipo T, una volta passato un int come T, tutte le alter variabili di tipo T devono essere interi.

Un esempio di questo si trova nelle liste generiche, definite allo stesso modo per tutti i tipi, ma una volta definite List<string>, accetteranno in input solo stringhe.

1. *Descrivere le differenze tra metodo statico e d’istanza*

I metodi statici sono comuni a tutti gli oggetti della stessa classe. Questi non hanno bisogno di istanziare oggetti per essere definiti. Al contrario, I metodi d’istanza sono relative proprio al singolo oggetto della classe e necessitano quindi la creazione di un oggetto per la definizione. All’interno del metodo però, essendo relativo ad un oggetto della classe, si potranno richiamare proprietà e campi dell’oggetto a cui è riferito il metodo senza doverlo passare in input.

1. *Fornire una definizione di modificatore e accessor. Spiegarne l’uso e le differenze.*

Gli accessor definiscono l’accessibilità di una proprietà: get indicherà la possibilità di lettura della stessa, mentre set dà la possibilità di scrivere nella proprietà. Se la proprietà ha entrambi, sarà possible leggerla e settarla. Se è presente solo il get sarà read-only, se solo il set, write -only.

Proprietà { get; set;}

I modificatori invece servono per modulare l’accessibilità. Nell’esempio, l’aggettivo public indica che posso accedere a Proprietà dall’esterno, sia al get che al set. Questi possono avere un accesso differenziato, aggiungendo ad esempio un private, si potrà accedere solo dall’interno della classe stessa. Si possono modificare anche i singoli accessori:

Public proprietà { get; private set;}

In questo modo proprità si potrà leggere anche dall’esterno, perché prende il public, ma settata solo dall’interno della classe.

Altri modificatori sono protected, (solo dalla classi derivate) o internal (solo dall’interno dell’assembly).

*Esercitazione pratica*

*Creare una Console App che gestisca l’iscrizione ad un esame di uno Studente.*

*Lo studente è definito con:*

* *Nome*
* *Cognome*
* *AnnoDiNascita*
* *Immatricolazione*
* *Esami*
* *RichiestaLaurea*

*L’immatricolazione ha le seguenti caratteristiche:*

* *Matricola*
* *DataInizio*
* *CorsoDiLaurea*
* *FuoriCorso*
* *CFUAccumulati*

*Un Corso di laurea è dato da un Nome, AnniDiCorso, i cfu per ottenere la laurea e una lista di corsi associati.*

*Un Corso ha un nome e dei CFU.*

*Un Esame si riferisce ad un corso e tiene conto se esso è stato passato.*

*I possibili nomi dei Corsi di Laurea possono essere solo i seguenti: Matematica, Fisica, Informatica, Ingegneria, Lettere.*

*La matricola dello studente deve essere univoca, autogenerata e read-only.*

*Uno studente può richiedere un esame solo se esso è presente nel Corso di Laurea associato allo studente, se i CFU del corso associato all’esame non superino i CFU massimi del Corso di laurea e se non ha il flag RichiestaLaurea assegnato a vero.*

*Nel caso le condizioni siano verificate, lo studente aggiunge l’esame alla lista Esami.*

*Scrivere inoltre un metodo EsamePassato che, dato un esame, vada ad aggiornare i CFU accumulati dallo studente, metta il flag Passato sull’esame e verifichi se con tale esame sono stati raggiunti i CFU necessari per richiedere la laurea (e quindi metta il flag Richiestalaurea a true);*

*Requisiti tecnici:*

*-Specificare almeno 3 costruttori*

*-Usare almeno una volta enum*

*Consigli:*

*-Potrebbe essere utile creare un paio di metodi ad hoc per creare al volo delle liste di corsi, corsi di laurea…*

*-Visto che le classi sono collegate strettamente l’una con le altre, verificate l’inizializzazione di ciascuna sia adeguata e che i riferimenti siano corretti.*

*Mettere la prova pratica e teorica su Github.*