# Test di fine settimana – Week 4

|  |  |  |  |
| --- | --- | --- | --- |
|  |  | Nome | Michela |
|  |  | Cognome | Putzu |
|  |  | Data | 23/07/2021 |

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. *Dare una definizione di classe statica*

Una classe è un tipo definito dall’utente. Quando una classe è statica ed è quindi è marchiata con la parala chiave “**static”** può essere utilizzata senza la necessità di istanziare gli oggetti che usiamo nel nostro codice. Con la classe statica andiamo a descrivere informazioni comuni a tutti gli oggetti di stesso tipo.

1. *Nell’istruzione* ***Persona p = new Dipendente();*** *la variabile p si comporta in maniera polimorfa. Perché? Cosa si intende per comportamento polimorfico?*

Con polimorfismo si intende la capacità degli oggetti di assumere comportamenti diversi a seconda della classe di appartenenza: è quindi la possibilità di trattare un’istanza di tipo come se fosse un’istanza di un altro tipo. Utilizza i marcatori virtual o abstract.

1. *In che modo è possibile utilizzare la combinazione di keyword del linguaggio virtual e override? Fornire un esempio.*

La keyword *virtual* viene usata per modificare una dichiarazione di metodo, proprietà, per consentire che sia sottoposta a override in una classe derivata.

public class Veicolo

{

public int ID { get; set; }

public string Proprietario { get; set; }

public int NumeroRuote { get; set; }

public string Descrizione { get; set; }

public override string Descrizione ()

{

return $"Descrizione: {Descrizione}";

}

public class Veicolo : Car

{

public string Targa { get; set; }

public int NumeroPorte { get; set; }

public override string Descrizione ()

{

return $"{Targa} Numero Porte {NumeroPorte}";

}

1. *Spiegare il significato di Tipo, Classe e Istanza nella programmazione OOP*

**Tipo:** è una rappresentazione concreta di un concetto

**Classe:** una classe è un tipo definito dall’utente

**Istanza:** è un particolare oggetto di una determinata classe. Ogni istanza è separata dalle altre ma condivide con altri oggetti della stessa classe le sue caratteristiche generali, quali attributi e metodi.

1. *Quali tipologie di ereditarietà sono consentite in C# e come si definisce? Fornire un esempio*

L’ereditarietà è uno strumento molto potente della programmazione ad oggetti che consente di definire una nuova classe che mantiene le proprietà (attributi e metodi) di una classe esistente ma aggiunge alla nuova classe nuovi attributi e nuovi metodi. Esistono due tipi di ereditarietà:

* Singola 🡪 una classe deriva da un’unica classe più grande
* Multipla 🡪 una classe deriva da due o più classi più grandi.

In C# è consentita solo l’ereditarietà singola.

Un esempio può essere: il CANE deriva dalla classe ANIMALE.

*Esercitazione pratica*

*Realizzare una Console Application per gestire una squadra di calcio.*

*La squadra è caratterizzata dai seguenti attributi:*

* *Nome*
* *Data di Fondazione*
* *Campionato disputato*
* *Nome del Presidente*
* *Nome dell’Allenatore*
* *Rosa (elenco di Calciatori)*

*Ciascun Calciatore in Rosa appartiene ad una categoria: Portiere, Difensore, Centrocampista, Attaccante.*

*Tutti i giocatori sono in possesso dei seguenti attributi:*

* *Numero di Maglia*
* *Nome*
* *Data di Nascita*

*Inoltre:*

* *Per i Portieri vanno registrati il Numero di Gol Subiti e il Numero di Rigori Parati*
* *Per i Difensori vanno registrati il Numero di Tackle Riusciti*
* *Per i Centrocampisti vanno registrati il Numero di Passaggi Tentati e il Numero di Passaggi Riusciti*
* *Per gli Attaccanti vanno registrati il Numero di Gol Realizzati*

*L’applicazione deve permettere di:*

* *Aggiungere Giocatori alla Rosa*
* *Vendere Giocatori (rimuoverli dalla Rosa)*
* *Gestire la squadra titolare (ovvero identificare gli 11 giocatori titolari) con lo schema 4-4-2 (4 Difensori, 4 Centrocampisti, 2 Attaccanti)*
* *Stampare le statistiche di tutta la squadra titolare*

*Mettere la prova pratica e teorica su Github.*