1. Quali sono le possibili cause di un’eccezione?  
   Le eccezioni si verificano quando si presentano errori dovuti a situazioni impreviste, come ad esempio sviste dello sviluppatore, problematiche hardware, autorizzazioni fallite, risorse non disponibili, ecc…
2. Descrivere il costrutto formale per la gestione di una eccezione fornendo un esempio.  
   Il costrutto try-catch serve a gestire l’eccezione. Il blocco try contiene il codice che potrebbe generare l’eccezione, mentre il blocco catch gestisce e controlla l’errore dell’eccezione che desidera catturare. Il blocco finally non è obbligatorio, si tratta di un blocco di codice che viene eseguito a prescindere dalla cattura o meno dell’eccezione.  
   Esempio:

try  
{  
 double divisione = 15 / 0;  
}  
catch(DivideByZeroException ex)  
{  
 Console.WriteLine(ex.Message);  
}  
finally  
{  
 Console.WriteLine(“Operazione terminata”);  
}

1. Descrivere i ruoli che esistono all’interno di un progetto  
   - Product owner: chi commissiona il prodotto  
   - Business analyst: l’intermediario tra product owner e team tecnico  
   - Software architect: identifica preventivamente gli aspetti tecnici del sistema  
   - Sviluppatore: riceve indicazioni da analyst e architect e produce il software tangibile  
   - Team di testing: verifica che il software rispetti i requisiti, ne controlla la logica funzionale, le performance e la sicurezza  
   - Team operation: predispone gli ambienti di esecuzione di Staging e di Production per il software finito e testato
2. Descrivere la differenza tra class diagram e object diagram descrivendo gli elementi di cui possono essere composti.   
     
   Il class diagram invece contiene i dati strutturali e funzionali del sistema, mentre l’ object diagram contiene anche dei valori esemplificativi al suo interno, serve a comprendere meglio quali tipi di dato aspettarsi.  
   Nel class diagram ogni classe è rappresentata da un rettangolo, se il suo nome è in corsivo si tratta di una classe abstract.   
   Ogni classe contiene attributi (con tipo di dato specificato) e/o operazioni  
   Ogni elemento del diagramma ha i marcatori di visibilità indicati con +(public), -(private) e #(protected).  
   Le classi sono collegate tra loro tramite linee di associazione che possono essere nominate e contenere la specifica della molteplicità.  
   E’ possibile inserire l’ereditarietà con una freccia che collega l’entità alla classe padre, ed è possibile inserire anche relazioni di aggregazione (collegamento con rombo vuoto) e composizione (collegamento con rombo pieno), dove nel primo caso le entità primaria esiste a prescindere dalle altre, mentre nel secondo l’entità primaria esiste solo se sono presenti le altre.  
   E’ inoltre possibile applicare interfacce con <<interface>> sopra al nome della classe oppure con la notazione compatta.  
   Gli elementi dell’object diagram sono molto simili a quelli del class diagram.