1. Quali sono le possibili cause di un’eccezione?
   1. Le possibili cause di un’eccezione sono:
      1. Disattenzioni dello sviluppatore
      2. Problemi di Hardware
      3. Autorizzazioni o autentificazioni fallite
      4. Risorse non disponibili o non trovate
2. Descrivere il costrutto formale per la gestione di una eccezione fornendo un esempio.
   1. Le eccezioni si possono gestire principalmente in due modi.

In modo preventivo con dei controlli. Ad esempio sul formato di input dell’utente. Se è richiesto l’inserimento di un intero è possibile fare un semplice ciclo Do While con all’interno il tryparse, che continuerà a girare fino a quando l’utente non inserirà un valore intero.

*bool isInt = true;*

*int num;*

*do*

*{*

*isInt = int.TryParse(Console.ReadLine(), out num);*

*} while (!isInt);*

*return num;*

Oppure si possono gestire con un blocco TryCatch.

All’interno del TRY ci sarà l’operazione che l’utente cerca di compiere, mentre nel CATCH ci saranno le varie eccezioni, alcune specifiche ed altre più generali, che possono venire a crearsi.

*try*

*{*

*//OPERAZIONE DELL’UTENTE  
 //In questo caso potrebbe essere la scrittura su file*

*}*

*catch (FileNotFoundException)*

*{*

*//IL FILE NON E’ STATO TROVATO*

*//Qui lo sviluppatore potrà inserire una frase attraverso il console.writeline nella quale specifica l’errore*

*}*

*catch (FileLoadException)*

*{*

*//ERRORE DI CARICAMENTO*

*//Come sopra è possibile inserire una frase di cortesia*

*}*

*catch (Exception ex)*

*{*

*//ERRORE GENERICO*

*//Attraverso ex.Message, inserito in una console.writeline è possibile riportare l’errore che non rientra nei casi specificati dallo sviluppatore*

*}*

1. Descrivere i ruoli che esistono all’interno di un progetto
   1. All’interno di un progetto è possibile individuare sei figure.
      1. Il Product Owner, ovvero il proprietario del progetto finale
      2. Il Business Analyst, le figure che si pongono come tramite tra il proprietario e gli sviluppatori
      3. Software Architect, gli ‘architetti’ coloro che forniscono agli sviluppatori il modo nel quale costruire il progetto richiesto
      4. Gli Sviluppatori, coloro che sviluppano il progetto seguendo le direttive degli analisti e degli architetti
      5. Il Team di Testing, che si occupa dei controlli sulla funzionalità del sistema, sulla qualità e sulla sicurezza
      6. Il Team Operations, è la squadra che prende il software completo e testato e lo configura in modo che risponda ai requisiti tecnici di esecuzione.
2. Descrivere la differenza tra class diagram e object diagram descrivendo gli elementi di cui possono essere composti.
   1. Sia le Class Diagram che i Object Diagram fanno parte dei Structure Diagram che hanno il compito di fornire una vista logica e fisica del contenuto del sistema.

Con le Class Diagram infatti è possibile mostrare le relazioni che intercorrono tra le varie classi e i loro attributi, ma anche i metodi che la distinguono.

Ad esempio si può prendere la classe Veicolo (in modo molto semplificato), nel Class Diagram potrei avere:

*+Nome: string //INDICO CHE SI TRATTA DI UN ATTRIBUTO CHIAMATO ‘Nome’, DI TIPO STRINGA*

*+Prezzo: decimal //ATTRIBUTO PREZZO CON TIPO DECIMAL*

Collegato a questo si trova l’Object Diagram che fornisce i valori ideali delle classi descritte nel Class Diagram

+Nome: Fiat Panda

+Prezzo: 6340,99