**Test Gestione di Errori e Assertion**

1. *Quanti e quali tipi di errori sono presenti in Java?*

In Java ci sono 4 tipi di errori: errori di input dell’utente, errori dei dispositivi, restrizioni fisiche ed errori di codice

1. *In caso di errore, java permette di uscire dal metodo con un percorso che non restituisce nessun valore?*

Si Java lo permette, al contrario degli altri linguaggi di programmazione che restituiscono un codice di errore che viene analizzato dal metodo chiamante

1. *Cosa sono le eccezioni?*

Sono oggetti appartenenti alla Classe Exception che rappresentano gli errori che possono essere gestiti.

1. *Quali sono le gerarchie che partano da Throwable? Scrivere in dettaglio.*

Ci sono due gerarchie: Error e Exception. Error descrive errori non gestibili. Si tratta di errori interni ed esaurimento delle risorse. Invece le Exception sono quelle eccezioni che possono essere gestite e rappresentano una situazione imprevista che il flusso di un’applicazione può incontrare.

1. *Qual’ è la differenza tra checked e unchecked Exception e come derivano?*

L’eccezione checked non deriva da RuntimeException, mentre le unchecked si.

Le RuntimeExceptio è un errore che intercettato dalla JVM durante il suo funzionamento.

1. *Possiamo lanciare delle eccezioni personalizzate?*

Si, possono essere lanciate a patto di estendere Exception oppure una sua sottoclasse

1. *Throw e return lavorano insieme per la gestione degli errori?*

* Si
* No

1. *Come possiamo intercettare le eccezioni?*

Per intercettare le eccezioni si usa il blocco try/catch. All’interno del primo blocco(il try) si inserisce il codice dove ci si aspetta che si presenti l’eccezione, mentre nel secondo blocco(il catch) viene inserito il gestore dell’eccezione. Nel caso in cui si verifichi l’eccezione all’interno del blocco try, il programma salta direttamente al gestore nel blocco catch

1. *È possibile intercettare eccezioni multiple?*

Si è possibile intercettare eccezioni multiple, mettendo un solo blocco try e più blocchi catch. I vari blocchi catch devono specificare che tipo di eccezione specificano e prima vanno inserite le eccezioni più specifiche e poi quelle più generiche

1. *Scrivere la giusta interpretazione per il meccanismo vincolato dell’eccezione*

Gestione dell’eccezione => try/catch

Propagazione dell’eccezione => throws

1. *Possiamo estendere il tipo di un’eccezione della classe super?*

* Si
* No

1. *A cosa serve finally?*

La keyword finally, identifica un blocco di codice che verrà eseguito indipendentemente che si verifichi o meno un’eccezione.

1. *Scrivere il blocco di codice dove lanci e gestisci una eccezione.*

public static void main(String args[]) {

int a = 10;

int b = 0;

try {

int c = a/b;

System.out.println(c);

}

catch (ArithmeticException exc) {

System.out.println(“Divisione per zero …”);

}

}

1. *Nel Try catch, quando possiamo utilizzare l’istruzione break, continue e return? A cosa servono?*

All’interno di un blocco try/catch queste istruzioni vengono eseguite solo dopo l’esecuzione del blocco finally. Questo perché con la finally si tenta di mantenere la consistenza dello stato, quindi prima di eseguire le istruzioni che vanno a modificare il flusso del codice

1. *Cosa si intende per assertion e come lo possiamo usare in termini di linguaggio java?*

È un’istruzione che permette di testare eventuali comportamenti che un’applicazione deve avere. La parola chiave è assert e il formalismo è il seguente:

\_assert espressione booleana;

\_assert espressione\_booleana: espressione\_stampabile;