Chapter- 14

Unit Testing

# Unit Testing:-

Program এ কোন বাগ আছে কিনা, এই সমস্যা সমাধানের জন্য একটি বিশেষ পদ্ধতির আশ্রয় নেওয়া হয় তাকে unit testing বলে।

#Unit annotation:-

* @Test – test method কে নির্দেশ করে।
* @Test(expected=Exception.class)- Test টি Fail করবে নাকি Exception throw করবে তা নির্দেশ করে।
* @Test(timeout=100)- Test fail করবে যদি ১০০ মিনি সেকেন্ডের বেশি সময় নেয়।
* @Before- এটি সবগুলু মেথড এর আগে execute হবে।
* @After- সবার পরে execute হবে।

#Enum:-

Enum সম্পকে বিশেষ তথ্য,

1. Enum declaration মূলত একটি ক্লাস যা কিনা java.lang.Enum class কে extends করে। একে extends করা যায় না। এনাম interface কে implement করতে পারে।
2. এনাম টাইপ সেইফ।
3. ‘==’ এবং equals() তুলনা করা যায়।
4. এতে new দ্বারা ইনস্ট্যান্স তৈরি করা যায় না।

#null:-

যেই object কাওকে refere করে না তাকে null বলে।

#object class:-

Java তে সব class এর প্যারেন্ট ক্লাস java.lang.Object.

* getClass()

এই মেথডটি একটি object এর run time represent class এর object return করে।

#Object clone()

1. যে class এর object কে clone করা হবে সেই ক্লাস কে java.lang.Cloneable interface কে implement করে।
2. Clone() method কে override করতে হবে।

#Types of Cloning:-

* Shallow cloning
* Deep cloning
* Constructor cloning

#equals() এর শর্তঃ-

* Reflexive:- যে কোন নন-নাল reference যদি x হয় তাহলে, x.equals(x) সব সময় true return করবে।
* Symmetric:-

Non-null reference যদি x & y হয় তবে, x.equals(y) & y.equals(x) true return করবে।

* Transtive:-

X=y=z;

#HashCode:-

এই hashCode() method প্রত্যেকটি object এর জন্য একটি unique value return করবে।

এর মেথড override করার শর্ত,

1. যদি দইটি object তাদের equals() অনুসারে সমান হয় তবে তাদের hash code অবশ্যয় সমান হবে।
2. দুটি অব্জেক্টের hashcode সমান হলে তারা সমান নাও হতে পারে।

Ex:-

@Override

Public int hashCode()

{

Return Object.hash(id,firstname,lastname)

}

* toString()

কোন object এর String represent করে।

#pass by reference value:-

Pass by copy of the variable value/ pass by value

#String Formatting:-

Format Specifier:-

%c- character

%s- String

%d- decimal

%f- float

%e- exponential floating point

%o- octal number

%x- hexadecimal number

Ex:-

System.out.printf(“%c”,’A’);

#sub-specifier:-

যেমন, width, flag, precision, argument index

“%[argument index] [flag] [width] [precision] type”

% নির্দেশ করে format specifier শুরু।

Argument index:-

%c, %d etc.

Flag:-

এটি বিশেষ ফরমাটিং instruction .

এটি তিন প্রকারের ‘-‘, ‘০’, ‘+’।

Width:-

যে object Format করা হবে সেই object টি সর্বনিম্ন কতগুলু character নিয়ে ফরম্যাট হবে।

Precision:-

Floating point এ ‘.’ এর কত ঘর পর্জন্ত প্রিন্ট করবে তা।

Type:-

সর্বশেষ টাইপ যেমন, c,d,s etc.

Ex:-

“%-\*6d”,5

“%.2f”,4.1254

#java keyword:-

* assert:-

কেনো statement এর ফলাফল সত্য কিনা তা যাচায় করতে ব্যবহ্যত হয়।

Int x=5;

Assert x==5;

* instance of:-

কেনো object একটি নিদিষ্ট ক্লাস এর তৈরি করা হয়েছে কিনা তা পরিক্ষা করে,

If(obj instanceof Student)

* native:-

এটি যে সব মেথড এর আগে ব্যবহার করা হয় সেগুলো হোস্ট সিস্টেমে সাধারণত সি,সি++ বা আসেম্বলিতে implement করা হয়।

* stictfp:-

floating pint operation গুলো জাতে ieee 750 এর মতো হয়।

public strictfp class Calculator

{

}

* transient:-

কোন ভেরিয়েবলের আগে এটি ব্যবহার করা হলে তা সিরিয়ালাইজড হয় না।

Public class Person implements Serializable

{

Private transient String name;

}

* Volatile:-

কোন variable কে main memory তে store করার জন্য ব্যবহ্যত হয়।

Private volatile int a=10;