Page No.
Date

Assignment 4 Interview questions.

Q1] What is the Role of the static keyword in the Context of memory management

ans:- The Static Keyword in Java is mainly used for memory management. Static Keyword is used to share the same variable or method of given class. The user can apply Static keyword with variable, methods, blocker & nested classes. the Static Keyword belong to the Class than an instance of the class.

Q2] Can static methods be overloaded & overriden In Java & How to static variables share access multiple instanter of a Class?

ans:- Yes Static method be overloaded in Java but cannot override thum.

- Static variables are shared access multiple instances of a class because only one Copy of the variable is created & use for the entire class. regardless of how many instances are Created.

③ what is significance of final keyword in Java?

ans:- final keyword in Java is the non-access modifier that prevents entities from being Changed or modified.

- In Java final keyword or class Cannot be modified or extended.

(4) What are ~~meaning of~~ narrowing & widening ~~in~~ Conversion in Java?

ans:- Widening Conversion

- It is used as implicit type of Conversion.
- In this Conversion we Convert lower datatype into higher datatype

- Narrowing:

- It is Caused as Explict type of Conversion.
- In this Conversion we Convert higher datatype into lower datatype

5) Provide examples of narrowing & widening Conversions between Primitive datatypes.

→ Example of Narrowing ~~widing~~ widening Conversion are between Primitive data types are:-
- Narrowing Conversion example:- byte short int long double, float.

short - Int long, double float
int — ~~low~~ long, double
long — double

1byte = 8 bits = 2^8 = 256 ~~byte~~ byte range

- Primitive Datatypes

Ex:-

```
int i = 128
byte b = 1
int i = 128
byte b = byte(i);
```

6) How does Java handles potensical loss of percession during narrowing Conversions.

ans:→ Because Java requires explicit Casting in order to narrow Conversions, programers must actively detect & manage the risk of data loss. This helps Java handle any percision loss that might happen when narrowing Conversions are made Sma. the language does not necessarily always prevent or alert against precession loss such transactions must be alone with the almost Corr. Case.

7) Explain the Concept of automatic widening Conversion in Java?

→ The Process by which the Java Compiler Conversi in Java.

Page No.
Date

(7) Explain the Concept of automatic (widening Conversion in Java.

ans→

The process by which the Java Compiler automatically transaction a value from a Samaller premitive data type to a larger one is known as automatic widenity Conversion'' in java. because The large type Can allways represent every possible value of the Smaller type without loss of ~~precision~~ perdision or data.

- In Java widening Conversions follow a special hierarchy from Smaller to larger type.

byte → short → int → long → float → double

Primitive data types

bytes (8 bits)
short (16 bits)
int (32 bits)
float (32 bits)
long (64 bits)
double (64 bits)

Narrowing Conversion:-

Converting a value from a bigger datatypes to a Smaller one is remain as a narrowing Conversion Ex. Converting double to ~~float~~ float or long to int

- Implications :-

(i) Type of Compatibility

(1) Type of Compatibility :-

1) Explicit Casting :- Narrowing Conversions require explicit Casting Due to the possibility of data loss the Java Compiler does not convey out these transformation (conversion) automatically.

2) Data loss :-

- Potential Data loss - Narrowing Conversion Can in data loss or truncation.

- overflow :- when narrowing between integer types values outside the range of the target type are loss

(8) what are the implication of narrowing & widening Conversions on type & Compatibility & data loss ?

→ Converting a value from a smaller data type to a larger one. It is known as widening Conversion for example Converting int to long or from to double.

Page No.
Date

Implecatims of Widening Conversions :-

1) Type Compatibilty :-

(i) Automatic :- Widening Conversion are automatic & safe Because there is no risk of data loss to overflow when using the larger type to represent all values of the smaller type. the java Compiler implicity Conducts these Conversions

(ii) Compatibility :- Compatibility ensure that we can assign a value of a smaller type to large type without explicit Casting.

(iii) Data loss :- No risk of data loss with widing Conversions.