ASSIGNMENT-6

Q1. What is method overloading in Java? with Eg.?

→ → Ability to define multiple methods in the same class with the same name but different parameters.

→ It is used when we want the methods to perform similar tasks but with different inputs or Value.

Q2 What are the rules for method overloading resolution in Java? How does Java determine which overloaded method to call?

→ Rules:

1. Methods must have same name but diff parameter.
2. Java selects the most spc. method that matches the agoments passed.
3. If no exact match is found, Java attempts to perform implicit type conversion to match a method signature.
4. If ambiguity arises, Java throws a compile time error indicating an ambiguous call.
5. Java does not consider return types when resolving overloading ambiguity.

Q3 What does static keyword mean in Java? Explain diff. b/w static and non-static.

⇒ Static keyword is used to declare members that belong to the class itself. Rather than to instances of class. This means that they are shared among all instances of class & can be accessed without creating an object of the class.

Static	Non-Static
* It Belongs to class itself.	* It Belongs to instances of the class.
* Shared among all instances of the class.	* Each instance has its own copy.
* Can access Static and non-Static members	* Can access only non-Static members.
* Initialized once, when the class is loaded.	* Initialized separately for each instance.
* Eg → Static int count; Static void method().	* Eg → int age; void printInfo().

Q4. Can Static methods be overloaded & overridden in Java? How Static Variable share across multiple instance of class?

⇒ Yes, It can be overloaded, meaning you can have multiple static methods with same name in same class but with different parameter. However, It can't be ~~overloaded~~ overridden, when we declare static in subclass with same signature as static method in superclass, it hides the superclass method instead of overriding it.

* Static variable are shared across multiple instances of class because there is only one copy of variable for entire class.

Q5 Role of static keyword in memory management?

→ i) It belongs to the class itself rather than to instances of class.

ii) It is stored in "Method area" or class area of memory.

iii) Allocated when the class is loaded into memory & remains until the class is unloaded.

iv) Proper management is crucial to avoid potential memory leaks.

Q6 Significance of final keyword?

→ 1. It Gives security that none can modify our classes, variables & methods.

2. It leads to better performance & maintability.

3. When applied to method, it means method cannot be overridden by subclass.

Q7 Can final method be overridden in a subclass? How does the final keyword affect variable method & class in Java?

⇒ No, If you attempt to override final method you'll get a compilation error.

i) Final Variable → It means the variable's value cannot be changed once initialized. It must be initialized either at declaration or with the constructor of class.

ii) Final method → means method cannot be overridden by subclass. Used to prevent subclasses from altering the behavior of method in superclass.

iii) Final classes → means class cannot be subclass

Q8 This keyword in Java and how this is used in construction & methods?

⇒ Current instance of class in which it appears. It is often used to distinguish b/w instance variable & parameters with same name.

⇒ Used in → i) Accessing instance variables.
ii) Calling another constructor.

i) If a parameter of method or constructor has same name as instance variable then this can be used to refer instance variable.

ii) Useful in constructor chaining, where one constructor can initialize common variables & then call other constructor to avoid code duplication.

Q9. Narrowing and Widening in Java?

⇒ Widening → It occurs when a data type with smaller range is converted into a data type of larger Range. It happens automatically by compiler.

Eg. → conversion from int to float.

⇒ Narrowing → Occurs when data type of larger Range is converted into smaller range. It req. explicit casting because it may Result in loss of data.

Eg → double to int.

→ explicity call to constructor

→ instance initializer Block in the Class...

→ everytime before.

Q10 Eg 1→ int smallInt = 10;
double bigDouble = smallInt;
S.O.P ("Widening (implicit) : " + bigDouble);
Eg 2 → double bigDouble = 123.45; // 10.0 ← O/P
int smallInt = (int) bigDouble; // 123. O/P

Q11 How java handles potential loss of precision during narrowing?

⇒ It handles by simply discarding the excess bits beyond the capacity of target data type.

Q12 Automatic widening in Java?

⇒ Occurs when a value of smaller primitive DT is converted into target Data type.

Eg → int smallNumber = 10;
long largeNumber = smallNumber;

Q13 What are implications of narrowing & widening conversions on type compatibility & data loss?

⇒ · Implications →

i) Narrowing can lead to potential data loss because some values from the source type may not be representable in target type.

ii) Widening don't result in data loss since the target can represent all the value of source type.