Assignment : 4

1. What is the role static keyword in the context of memory management?

→ - If we want to share value of a field in all the instances of same class then field should be static
- This is also called as class level variable
- It get space once per class during class loading on a method area.
- To access the static variable we should define static method inside class.
- Static methods are designed to call on class name, it doesn't get "this" reference.
- Inside non static method, we can access static as well as non static members but inside static method we can access only static members of the class.

Difference bet^n static & non static methods

• Static	• Non static
- Associated with class	- Associated with the instance
- can only directly access static members & methods of class	- can directly access both static & non static members and methods of the class
- Initialized when the class is loaded into memory	- Initialized when an instance of the class is created
- Available throughout the program's execution	- Available as long as the instance exists

2. can static methods be overloaded & overridden in java? how static variables shared across multiple instances of a class?

→ - Static methods can be overloaded in java. overloading means having same methods name but different parameters. for ex:

```
class sample {
    static void display() {
        System.out.println("Hello"); }
    static void display(int a) {
        System.out.println("Hey"+a); } }
```

- Static methods cannot be overridden in java because static methods are associated

with the class itself, not with instances of the class.

- Static variables are shared across all instances of a class. This means that there is only one copy of a static variable, regardless of how many instances of the class are created. All instances of the class access the same static variable & changes made to it by one instance are visible to all other instances.

3 What is the significance of the final keyword in Java?

→ In Java, the final keyword is used to indicate that a variable, method or class cannot be modified or extended.

- Final variables: When a variable is declared as final, its value cannot be changed once it has been initialized.

- Final methods: When a method is declared as final, it cannot be overridden by a subclass. This is useful for methods that are part of a class' public API & should not be modified by subclass.

- Final classes: When a class is declared as final, it cannot be extended by a subclass.

Final variables must be initialized either at the time of declaration or in the constructor of the class.

4 What are narrowing & widening conversions in java?

→ These type of conversions that occur when data types are converted from one to another

- Widening: (Automatic type conversion): This occurs when a smaller data type is converted into a larger data type automatically by the java compiler. No data loss occurs.

for ex: Byte → Short → Int → long → float → double

- Narrowing: (Explicit type casting): This occurs when larger data type is converted to a smaller data type. Potential to data loss.

for ex: Double → float → long → int → short → byte

5. Provide examples of narrowing and widening conversions bet" primitive data types.

→ Widening happens when converting a smaller type to a large type. This is done implicitly in java because there is no risk of losing data. Some of the examples are:

1] byte to short:

```
byte b = 20;
short s = b
```

2] short to int

```
short s = 1000;
int i = s;
```

3] int to long

```
int i = 1000;
long l = i;
```

4] float to double

```
float f = 10.5f
double b = f;
```

Narrowing conversions: It occurs when converting a larger type to a smaller type. This can potentially lose data, so it must be done explicitly using a cast. Some of the examples are:

1] int to short

```
int i = 1000;
short s = (short) i;
```

2] long to int

```
long l = 100000L;
int i = (int) l;
```

6 How does java handle potential loss of precision during narrowing conversions?

→ Some of the ways are:

- One way to avoid this issue is to use larger data types that can fully accomodate the values without conversion.
- Another way is to implement validations logic to catch potential overflows before they occur, for ex. by comparing the value against the max & min limits of smaller data types.
- Using methods like like math.round() for rounding the floating values before converting to integers.

7 What are the implications of narrowing & widening conversion on type compatibility & data loss?

→ Widening conversion occurs when a value of smaller data type is assigned to a larger data type. This type of conversion is implicit because smaller type can always fit into larger data type without data loss. There is no risk of data loss.

• Narrowing conversion occurs when a value of a larger data type is assigned to a smaller type. This requires explicit casting because the larger type may not fit within the range of smaller type, There is a significant risk of data loss if value being converted is outside the range

Key implications:
- Widening is safer: no risk of data loss & automatically done by compiler
- Narrowing needs caution: It leads to unintended results. like overflow or truncation. We need to do explicit type casting