

## Assignment 4

Q1) what is the role of static keyword in the context of memory management?

=>

Static is the class level keyword i.e. all the instances of a class.

- variable allocated during compile time & lasts for the entire program run
- Static variable is shared among all object of a class level
- Static variable are declared using the static keyword
- accessed using classname
- we don't need to access using object of class
- only one copy of static variable exist in memory.

Q2) can static method be overloaded and overridden in java? How static variable shared across multiple instances of a class.

=>

- Static variable will overloaded but not overridden.
- overloading i.e. having multiple methods with the same name in the same class but different parameters.



- DATE
- Static do not overriding i.e. overridden in subclass.
  - when we declare a static method in a subclass with the same name & parameters as a static method in the superclass.
  - Static are associated with the class itself not with any individual instance.
  - one copy of the static variable regardless of how many objects are created from the class.
  - all instances refer to the same static variable changes made to the static variable through one instance are visible to all other instances.

Q) What is the significance of the final keyword in Java.

⇒

- Final value can't be changed once it has been assigned
- `final int = 10`

- There is final method also which cannot be overridden by any subclasses
- when we prevent alteration of a method's behaviour in subclasses ensuring that the method's implementation remain consistent



- Final ~~in~~ helps in creating more reliable and predictable code by ensuring certain aspects of the program's structure remain fixed & unchanged.

Q) what are narrowing & widening conversions in Java?

Ans:-

= widening  
Converting value of narrow type to wider type. This conversion is widening.

```
int num1=10
```

```
double num2=(double)num1;
```

```
System.out.println(num2);
```

- In widening explicit type casting is optional.


⇒ Narrowing

Converting wider type to narrow type

```
double num1=10.5
```

```
int num2=(int)num1;
```

```
System.out.println(num2);
```

Q) provide example 

Q) How does Java handle potential loss of precision during narrowing conversions?

⇒ By implicit & Explicit Narrowing.



Implicit Narrowing:- converting from a larger to a smaller data type without explicit casting in a compile time error.

Explicit Narrowing:- making

Q) Explain the concept automatic widening conversion in Java?

⇒

Automatic Widening

- The data is automatically casted from byte to short, short to int, int to long, long to float and float to double.
- process where java automatically convert the smaller data type to a larger data type without requiring explicit casting by the programmer.

```
int i = 5;
double d = 10.5;
double result = i + d;
```

⇒ Here i is automatically widened to double.

Q) What are the implications of narrowing and widening conversions on type compatibility and data loss?

⇒

- widening conversions are always safe and compatible. The larger data type can represent all possible values of the smaller data type.

⇒ Java handles widening conversions automatically without requiring explicit casts.

⇒ No loss of data because larger type can accommodate the value of the smaller type.

Example: widening conversions

```
int i = 10;
long l = i; // automatic widening
double d = l; // automatic widening
```

Example: narrowing conversions (require explicit cast)

```
long l = 10L;
int i = (int) l; // explicit narrowing cast
```

Example: automatic widening conversions

```
byte b = 10;
short s = b; // automatic widening
int i = s; // automatic widening
```

Example: automatic widening conversions

```
float f = 10.0f;
double d = f; // automatic widening
```