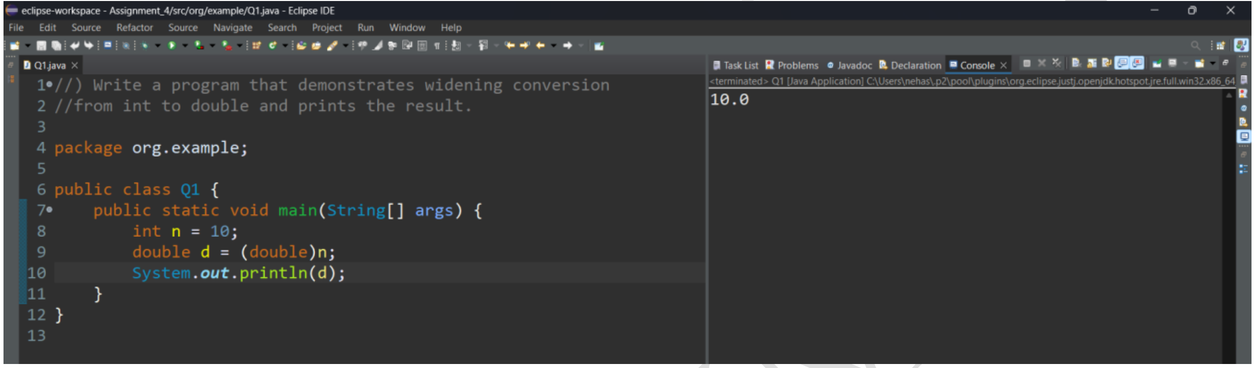
CDAC Mumbai PG-DAC August 24

Assignment No- 4

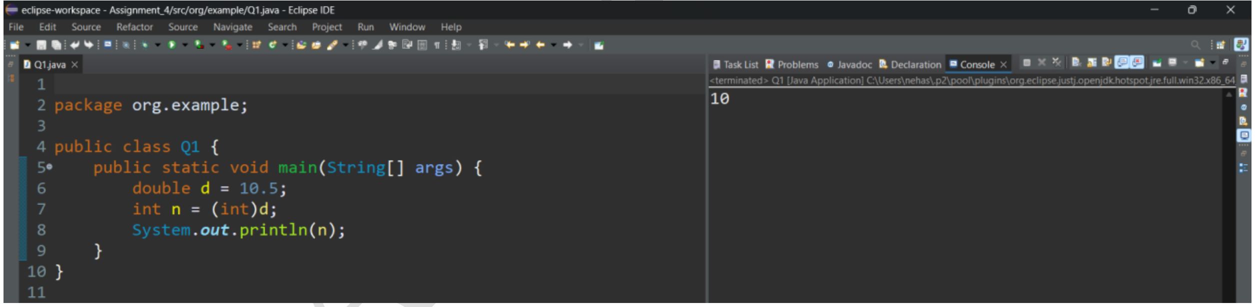
1) Write a program that demonstrates widening conversion from int to double and prints the result.


eclipse-workspace - Assignment_4/src/org/example/Q1.java - Eclipse IDE
File Edit Source Refactor Source Navigate Search Project Run Window Help
Q1.java
Task List Problems Javadoc Declaration Console
<terminated> Q1 [Java Application] C:\Users\nehas\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64

```java
//) Write a program that demonstrates widening conversion
//from int to double and prints the result.

package org.example;

public class Q1 {
    public static void main(String[] args) {
        int n = 10;
        double d = (double)n;
        System.out.println(d);
    }
}
```

```
10.0
```

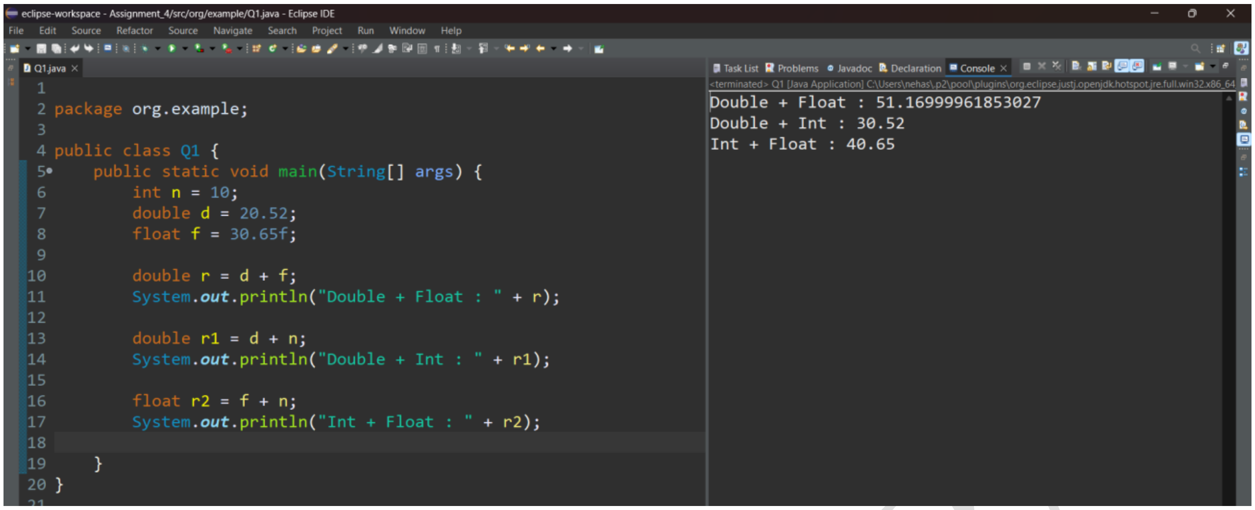
2) Create a program that demonstrates narrowing conversion from double to int and prints the result.


eclipse-workspace - Assignment_4/src/org/example/Q1.java - Eclipse IDE
File Edit Source Refactor Source Navigate Search Project Run Window Help
Q1.java
Task List Problems Javadoc Declaration Console
<terminated> Q1 [Java Application] C:\Users\nehas\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64

```java

package org.example;

public class Q1 {
    public static void main(String[] args) {
        double d = 10.5;
        int n = (int)d;
        System.out.println(n);
    }
}
```

```
10
```

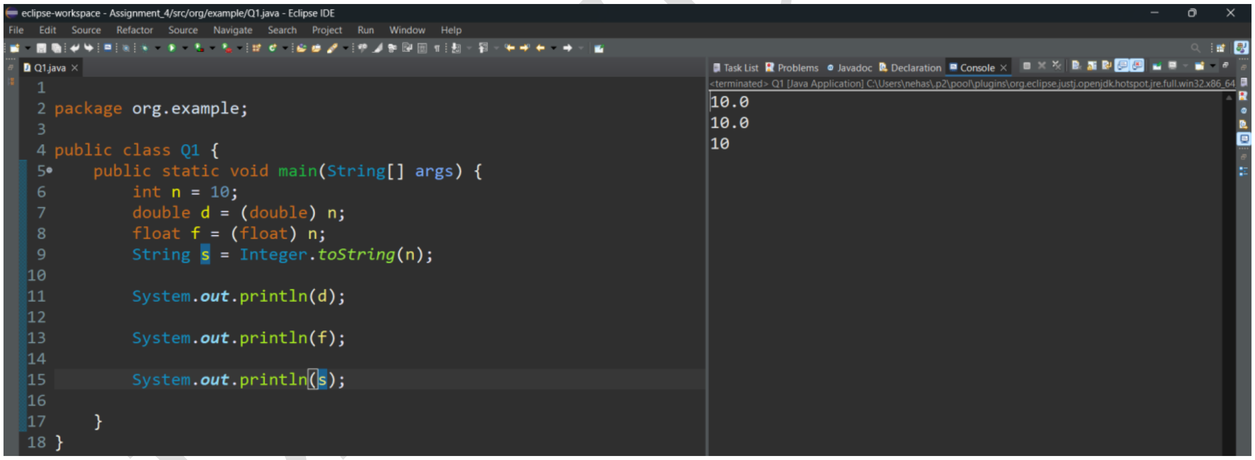
3) Write a program that performs arithmetic operations involving different data types (int, double, float) and observes how Java handles widening conversions automatically.


eclipse-workspace - Assignment_4/src/org/example/Q1.java - Eclipse IDE
File Edit Source Refactor Source Navigate Search Project Run Window Help
Q1.java
Task List Problems Javadoc Declaration Console
<terminated> Q1 [Java Application] C:\Users\nehas\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64

```java

package org.example;

public class Q1 {
    public static void main(String[] args) {
        int n = 10;
        double d = 20.52;
        float f = 30.65f;

        double r = d + f;
        System.out.println("Double + Float : " + r);

        double r1 = d + n;
        System.out.println("Double + Int : " + r1);

        float r2 = f + n;
        System.out.println("Int + Float : " + r2);

    }
}
```

```
Double + Float : 51.16999961853027
Double + Int : 30.52
Int + Float : 40.65
```

4) Write a Program that demonstrates widening conversion from int to (double,float, boolean, string) and prints the result.


eclipse-workspace - Assignment_4/src/org/example/Q1.java - Eclipse IDE
File Edit Source Refactor Source Navigate Search Project Run Window Help
Q1.java
Task List Problems Javadoc Declaration Console
<terminated> Q1 [Java Application] C:\Users\nehas\.p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64

```java

package org.example;

public class Q1 {
    public static void main(String[] args) {
        int n = 10;
        double d = (double) n;
        float f = (float) n;
        String s = Integer.toString(n);

        System.out.println(d);

        System.out.println(f);

        System.out.println(s);

    }
}
```

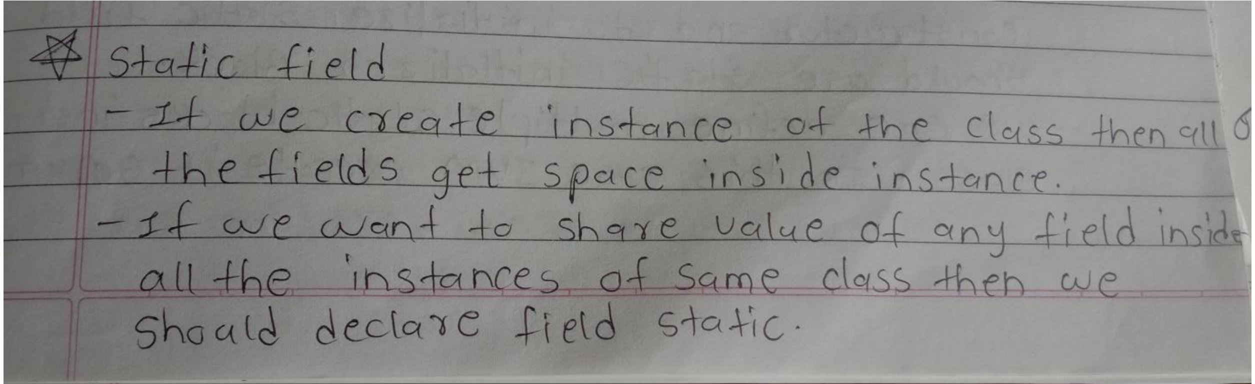
```
10.0
10.0
10
```

INTERVIEW QUESTIONS

Note: Write down this interview question on your notebook ,Take a screenshort & Paste that SS in the word document & upload on your Github.
What does the static keyword mean in Java? Explain

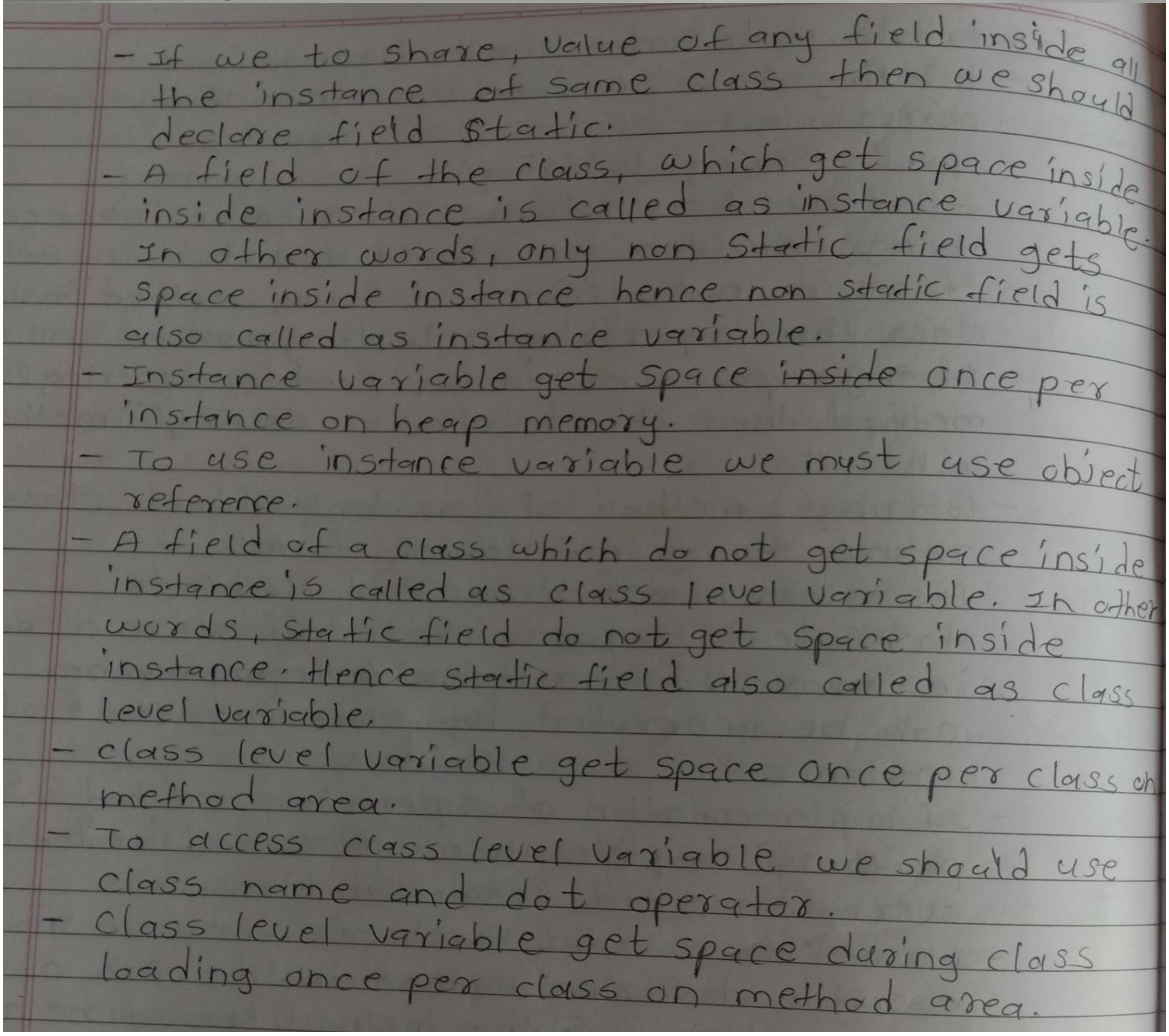
the difference between static and non-static methods.

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



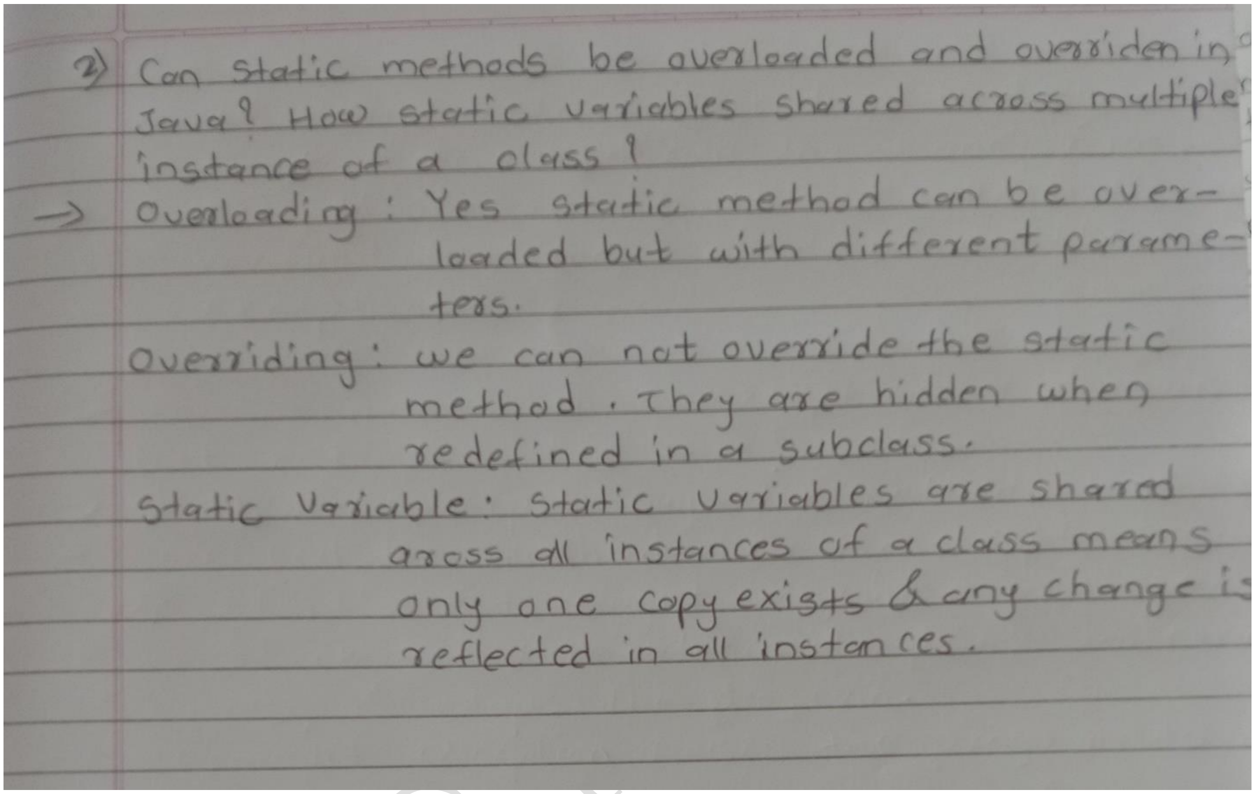
☆ Static field

- If we create instance of the class then all o the fields get space inside instance.
- If we want to share value of any field inside all the instances of same class then we Should declare field static.



- If we to share, value of any field inside all the instance of same class then we should declare field Static.
- A field of the class, which get space inside inside instance is called as instance variable. In other words, only non Static field gets space inside instance hence non static field is also called as instance variable.
- Instance variable get space ~~inside~~ once per instance on heap memory.
- To use instance variable we must use object reference.
- A field of a class which do not get space inside instance is called as class level variable. In other words, Static field do not get space inside instance. Hence static field also called as class level variable.
- class level variable get space once per class on method area.
- To access class level variable we should use class name and dot operator.
- Class level variable get space during class loading once per class on method area.

2. Can static methods be overloaded and overridden in Java?Howstatic variables shared across multiple instances of a class?



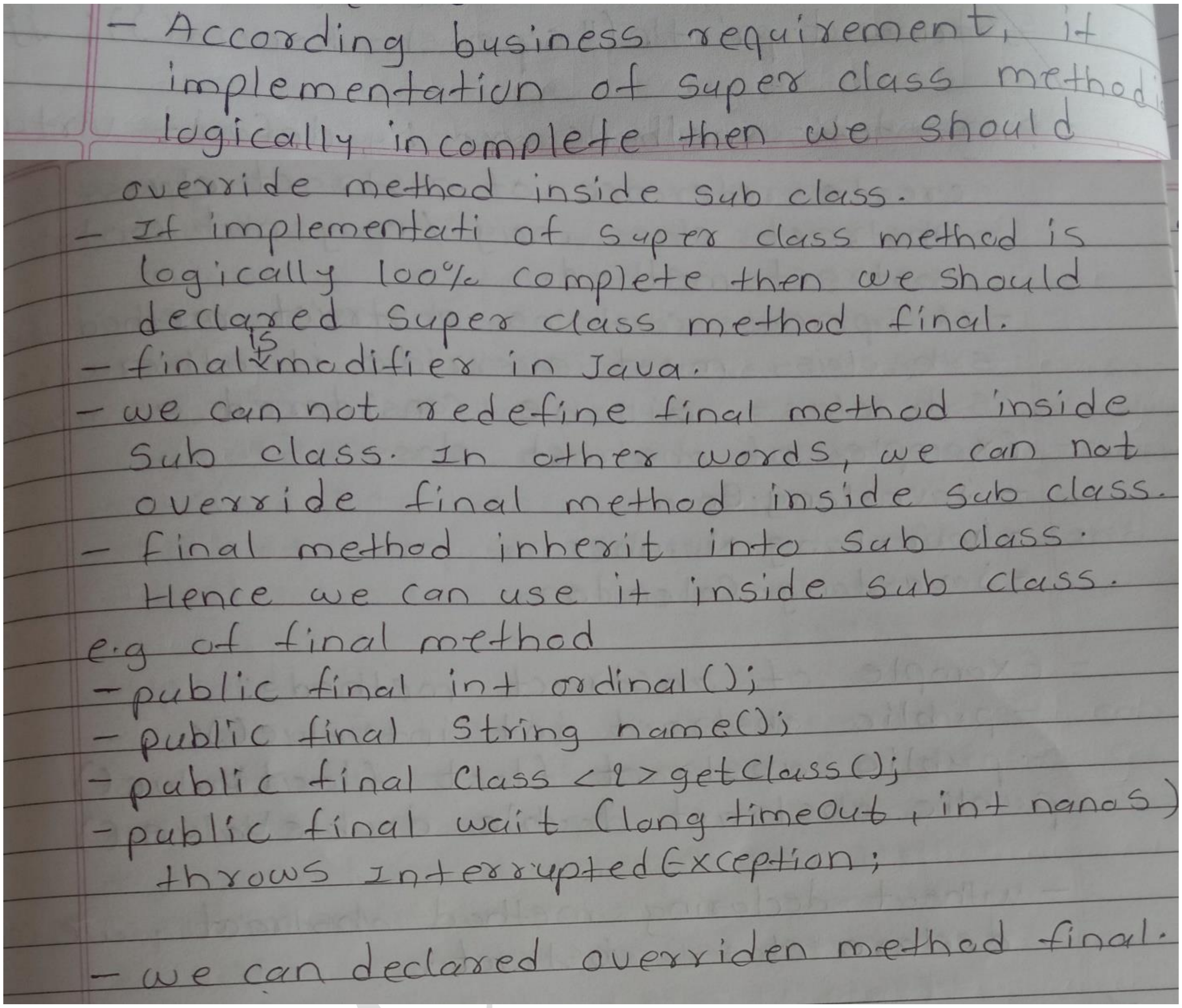
2) Can Static methods be overloaded and overriden in Java? How static variables shared across multiple instance of a class?

→ Overloading : Yes static method can be overloaded but with different parameters.

Overriding : we can not override the static method. They are hidden when redefined in a subclass.

Static Variable : Static variables are shared aross all instances of a class means only one copy exists & any change is reflected in all instances.

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



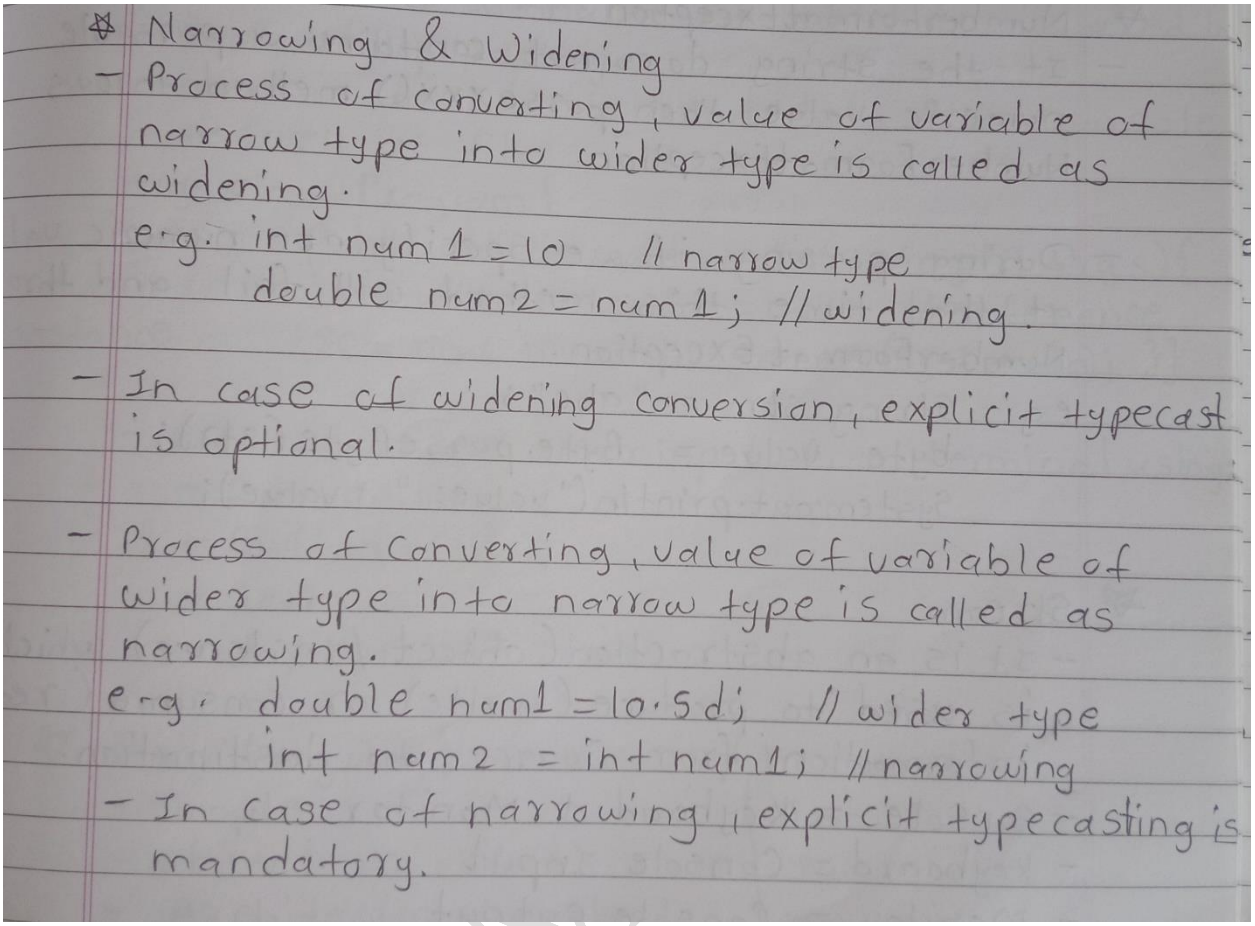
- According business requirement, if implementation of super class method logically incomplete then we should override method inside sub class.
- If implementati of super class method is logically 100% complete then we should declared super class method final.
- final is modifier in Java.
- we can not redefine final method inside sub class. In other words, we can not override final method inside sub class.
- final method inherit into sub class. Hence we can use it inside sub class.

e.g of final method

- public final int ordinal();
- public final String name();
- public final Class <?> getClass();
- public final wait (long timeOut, int nanos) throws InterruptedException;

- we can declared overriden method final.

4. What are narrowing and widening conversions in Java?



✱ Narrowing & Widening

- Process of converting, value of variable of narrow type into wider type is called as widening.

e.g.
```
int num 1 = 10    // narrow type
    double num2 = num 1; // widening.
```

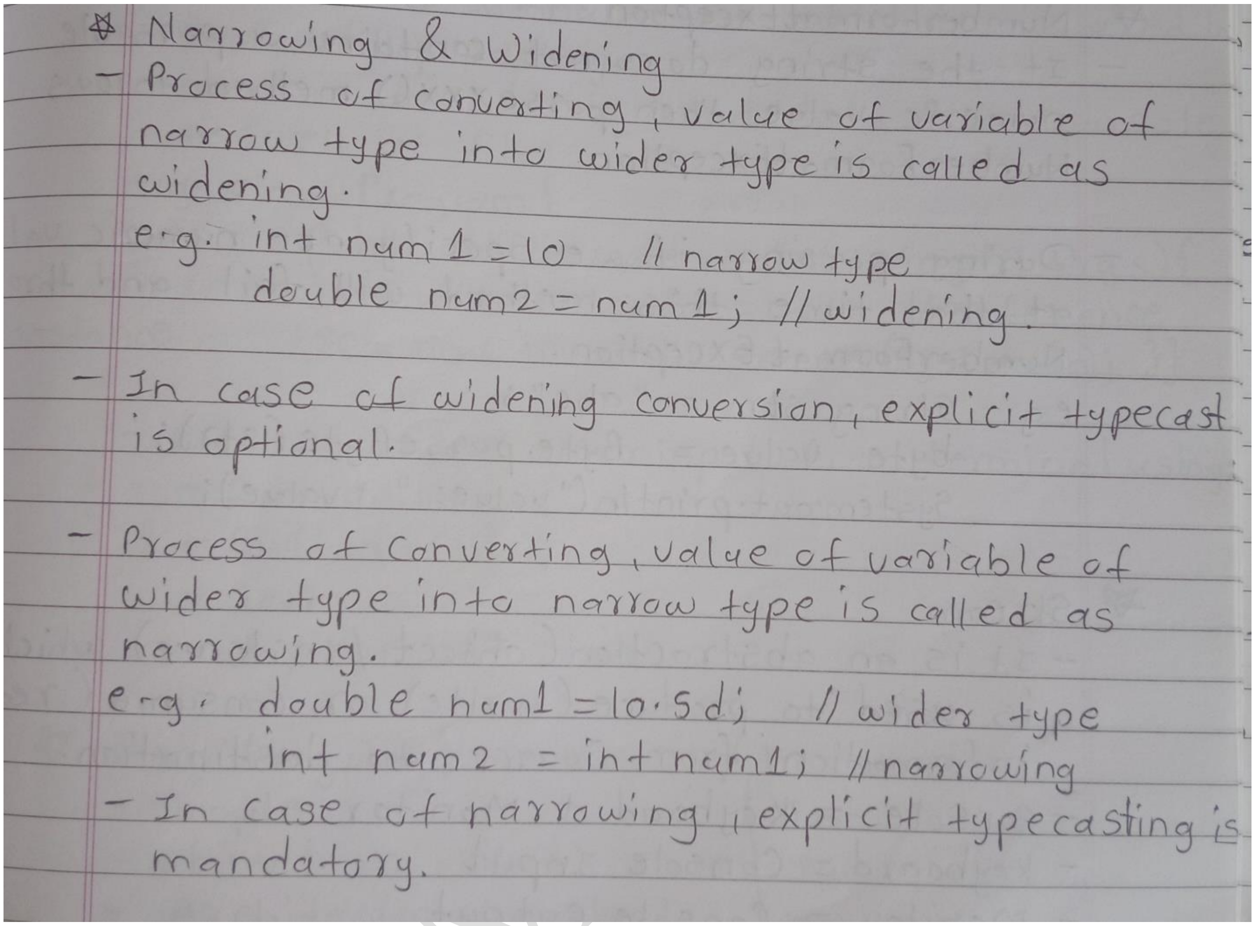
- In case of widening conversion, explicit typecast is optional.

- Process of converting, value of variable of wider type into narrow type is called as narrowing.

e.g.
```
double num1 = 10.5d;  // wider type
     int num 2  = int num1; // narrowing
```

- In case of narrowing, explicit typecasting is mandatory.

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



* Narrowing & Widening

- Process of converting, value of variable of narrow type into wider type is called as widening.

```
e.g. int num1 = 10      // narrow type
     double num2 = num1; // widening.
```

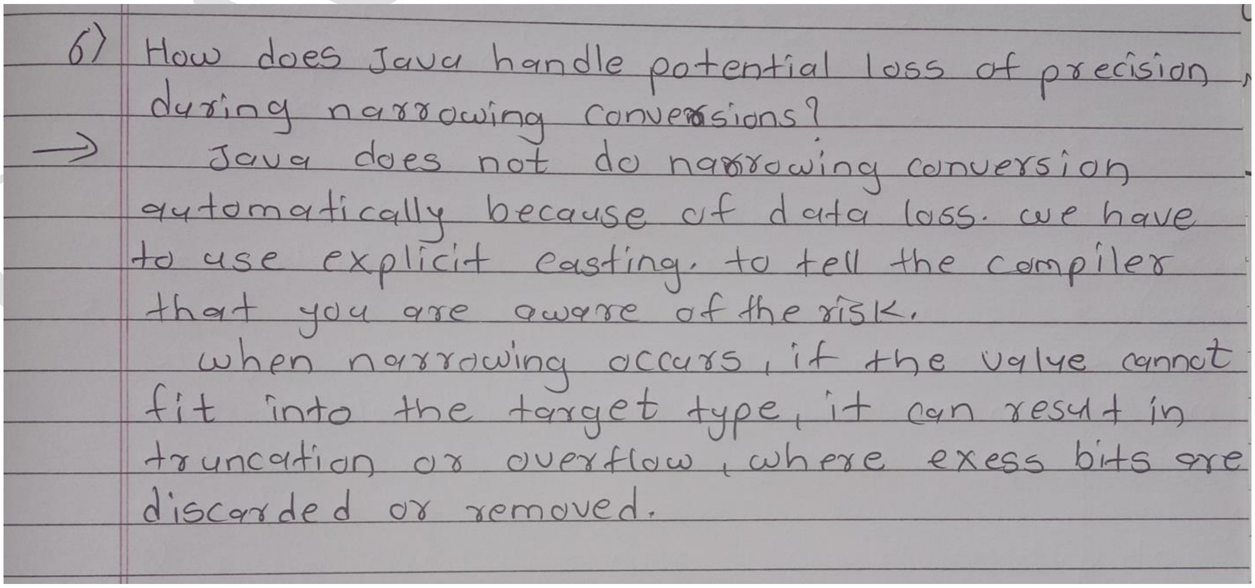
- In case of widening conversion, explicit typecast is optional.

- Process of converting, value of variable of wider type into narrow type is called as narrowing.

```
e.g. double num1 = 10.5d;   // wider type
     int num2 = int num1;   // narrowing
```

- In case of narrowing, explicit typecasting is mandatory.

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

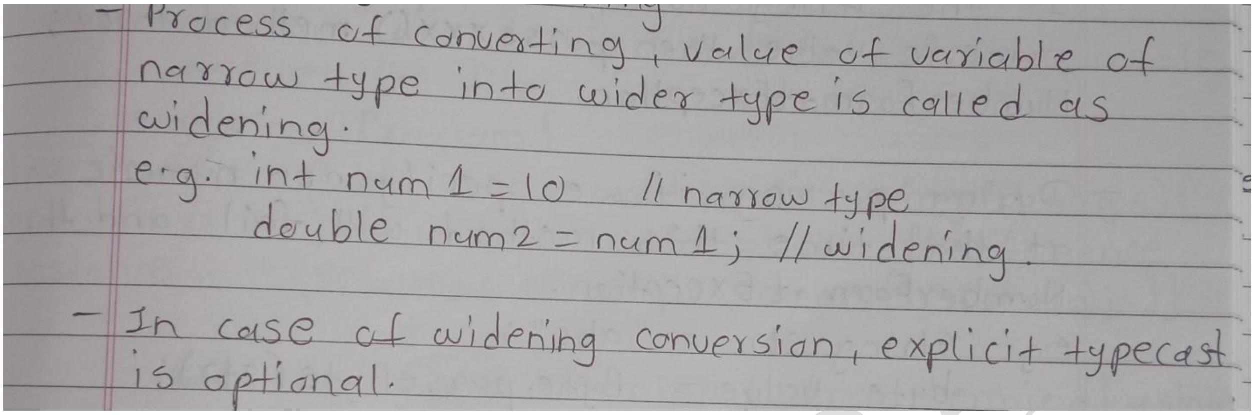


6) How does Java handle potential loss of precision during narrowing conversions?

→ Java does not do narrowing conversion automatically because of data loss. we have to use explicit casting. to tell the compiler that you are aware of the risk.

when narrowing occurs, if the valye cannot fit into the target type, it can resut in truncation or overflow, where exess bits are discarded or removed.

7. Explain the concept of automatic widening conversion in Java.

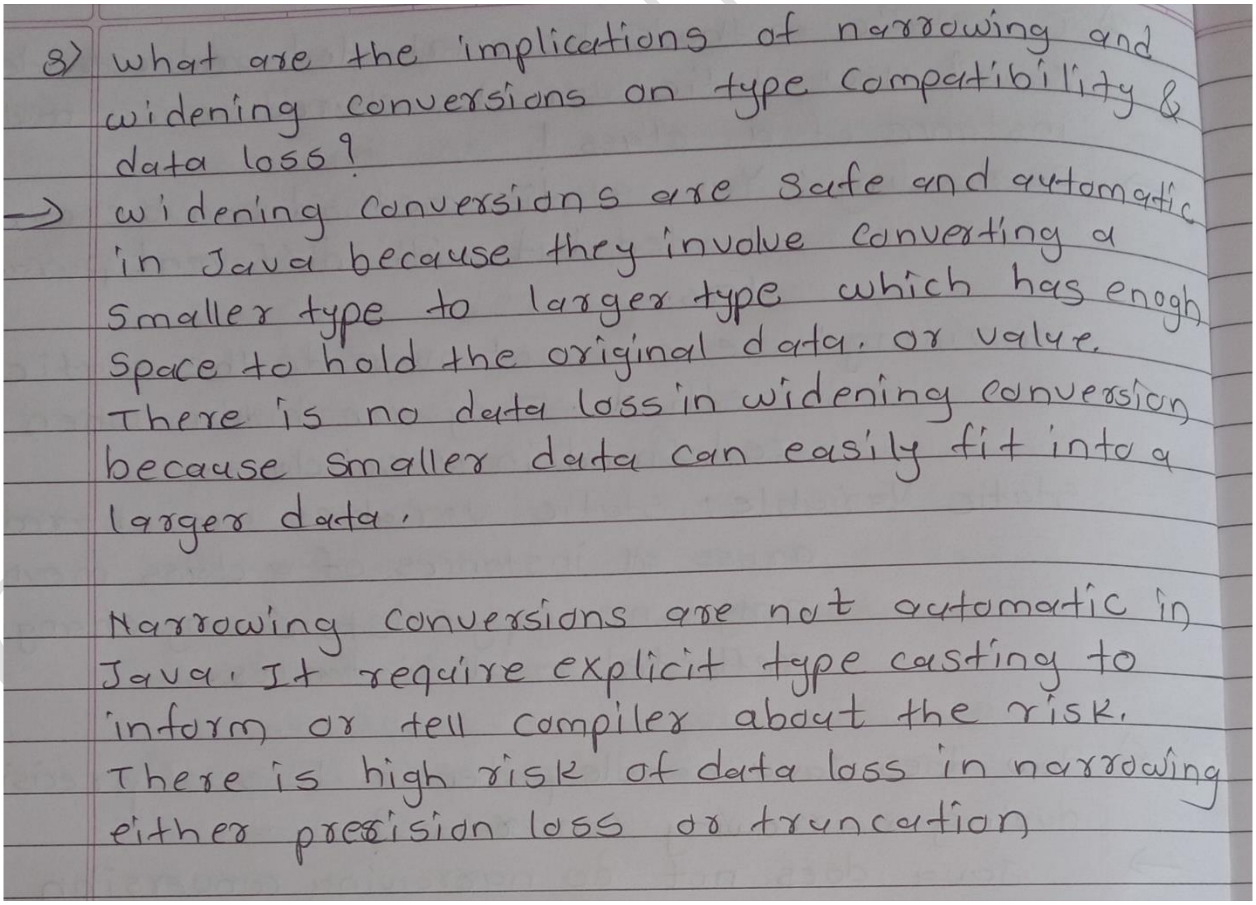


- Process of converting, value of variable of narrow type into wider type is called as widening.

e.g. int num 1 = 10 // narrow type
double num2 = num 1; // widening.

- In case of widening conversion, explicit typecast is optional.

8. What are the implications of narrowing and widening conversions on type compatibility and data loss?



8) what are the implications of narrowing and widening conversions on type compatibility & data loss?

→ widening conversions are safe and automatic in Java because they involve converting a smaller type to larger type which has enogh space to hold the original data. or value. There is no data loss in widening conversion because smaller data can easily fit into a larger data.

Narrowing conversions are not automatic in Java. It require explicit type casting to inform or tell compiler about the risk. There is high risk of data loss in narrowing either precision loss or truncation