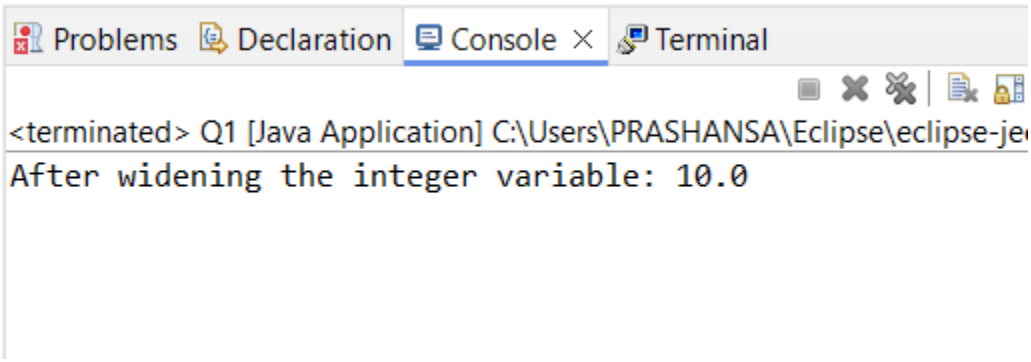
Name: Prashansa Nalawade

Assignment 4

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

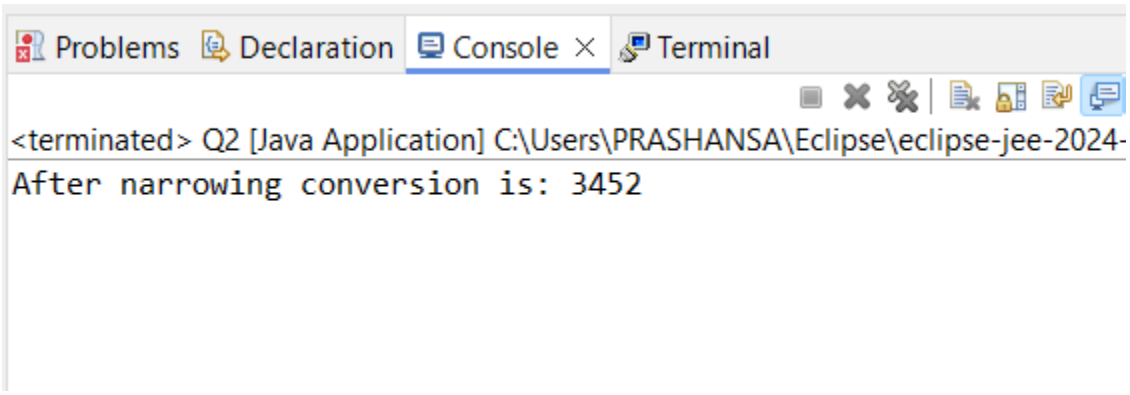
```
public class Q1 {

public static void main(String[] args) {

	// TODO Auto-generated method

	int num1= 10;

	double num2= num1;

	System.out.println("After widening the integer variable: "+num2);

   }

}
```



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

```
public class Q2 {
	public static void main(String[] args) {
		// TODO Auto-generated method stub
```

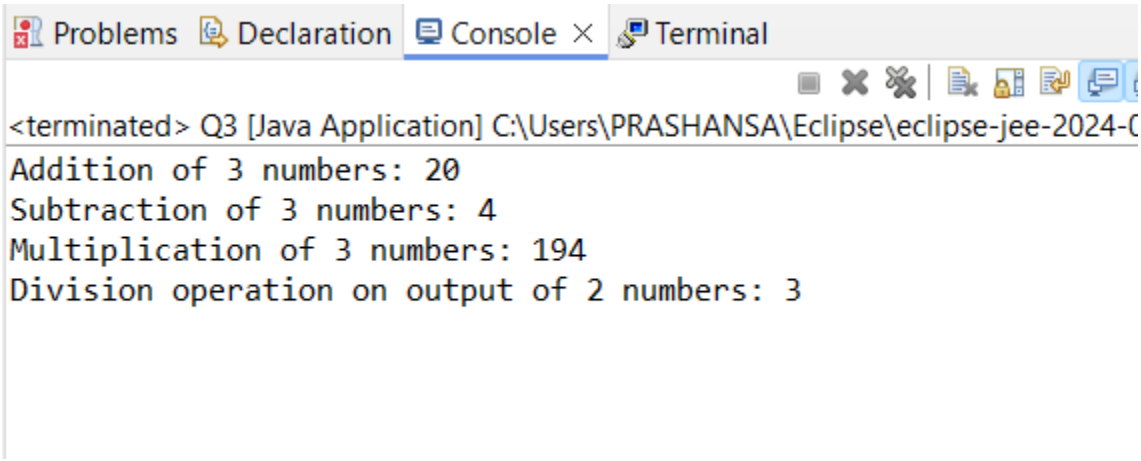
```
        double data = 3452.345d;

        int value = (int)data;

        System.out.println("After narrowing conversion is: "+ value);
    }
}
```



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

```
public class Q3 {
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        int a= 10;
        float b= (float) 2.5;
        float c= (float) 7.7678;
        long result1;

        result1= (long) (a+b+c);

        System.out.println("Addition of 3 numbers: "+ result1);

        long result2= (long) (a+b-c);

        System.out.println("Subtraction of 3 numbers: "+ result2);
```

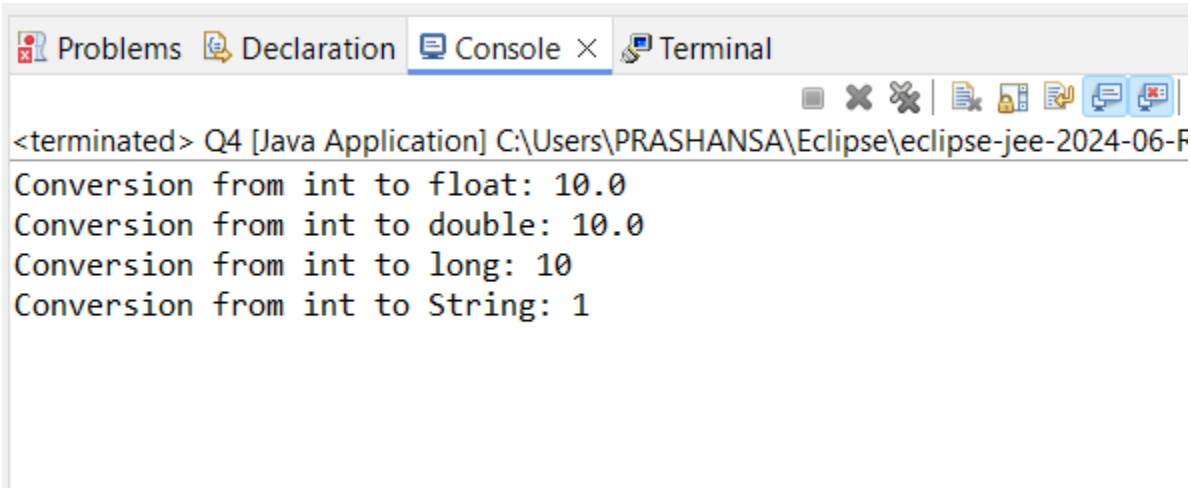
```
            long result3= (long) (a*b*c);

            System.out.println("Multiplication of 3 numbers: "+ result3);

            long result4= (long) (a*b/c);

            System.out.println("Division operation on output of 2 numbers: "+ result4);

        }
}
```



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

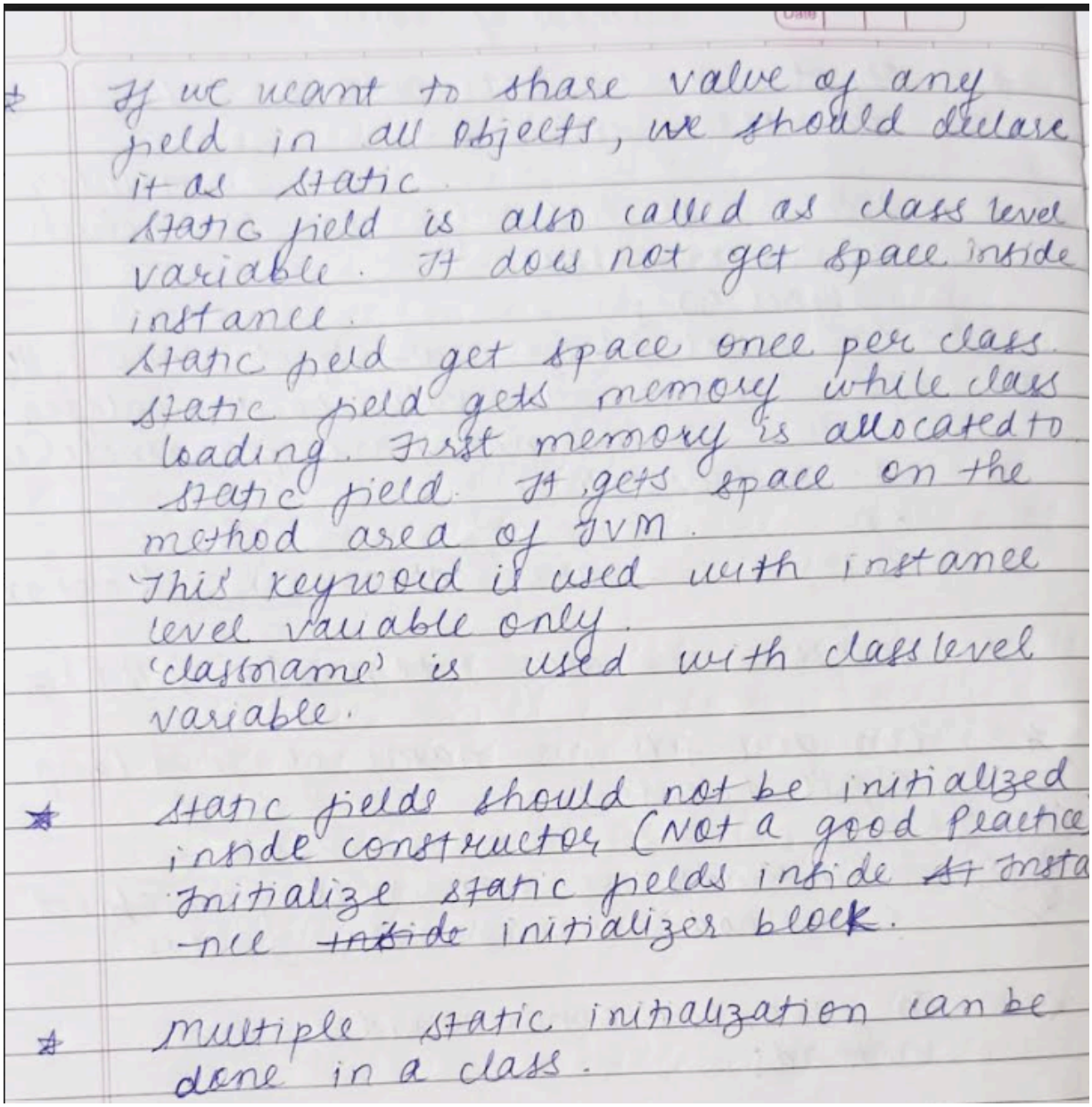
```
public class Q4 {
        public static void main(String[] args) {
            // TODO Auto-generated method stub

            int num1= 10;

            int num6= 1;

            float num2= num1;
            System.out.println("Conversion from int to float: "+ num2);

            double num3= num1;
            System.out.println("Conversion from int to double: "+ num3);

            long num4= num1;
```

```
                System.out.println("Conversion from int to long: "+ num4);

                //boolean num5= (Boolean)num6;      //Cannot convert int to boolean

                String num7= String.valueOf(num6);

                System.out.println("Conversion from int to String: "+ num7);
        }
}
```



Interview Questions:

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



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

NO, static methods in java cannot be overridden because static methods are associated with the class itself, rather than an instance of the class. when a subclass inherits a static method from its parent class, it is not possible to override the behaviour of that static method. They cannot be overridden because they do not act on a specific instance of an object. In Java, static methods can be overloaded but not overridden. They can have diff" parameters while having the same name in the same class or subclass. They cannot be overridden because they act on the class itself, not an object.

If we want to share value of any field in all objects, we should declare it as static.
Static field is also called as class level variable. It does not get space inside instance.
Static field get space once per class.
Static field gets memory while class loading. First memory is allocated to static field. It gets space on the method area of JVM.
This keyword is used with instance level variable only.
'classname' is used with class level variable.

Static fields should not be initialized inside constructor (Not a good practice)
Initialize static fields inside ~~#~~ instance ~~inside~~ initializer block.

Multiple static initialization can be done in a class.

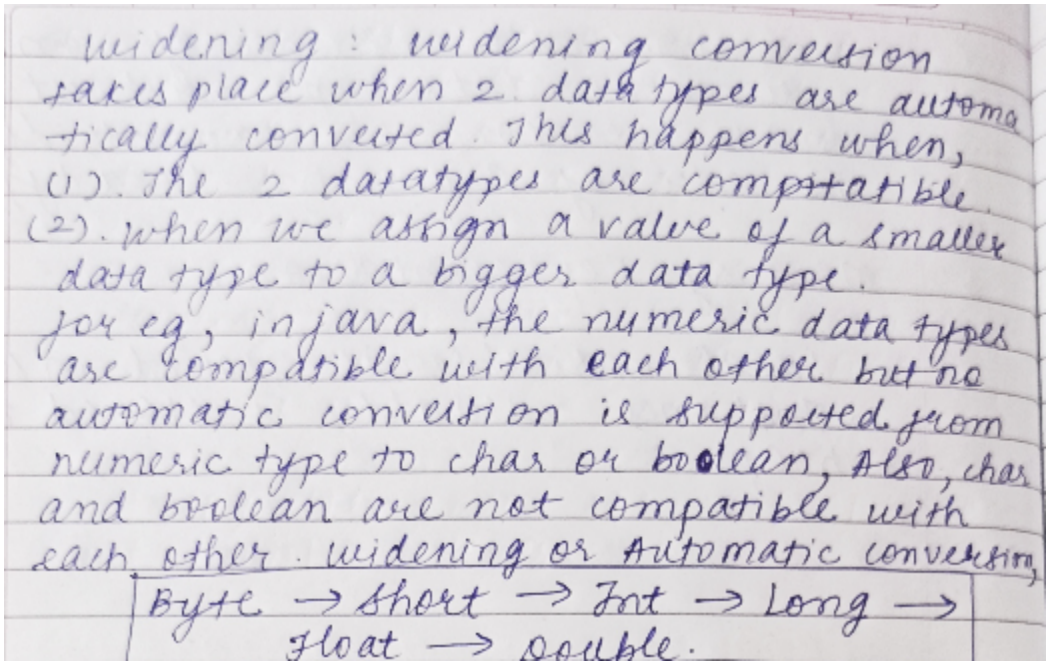
Method having a body is called concrete method.

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

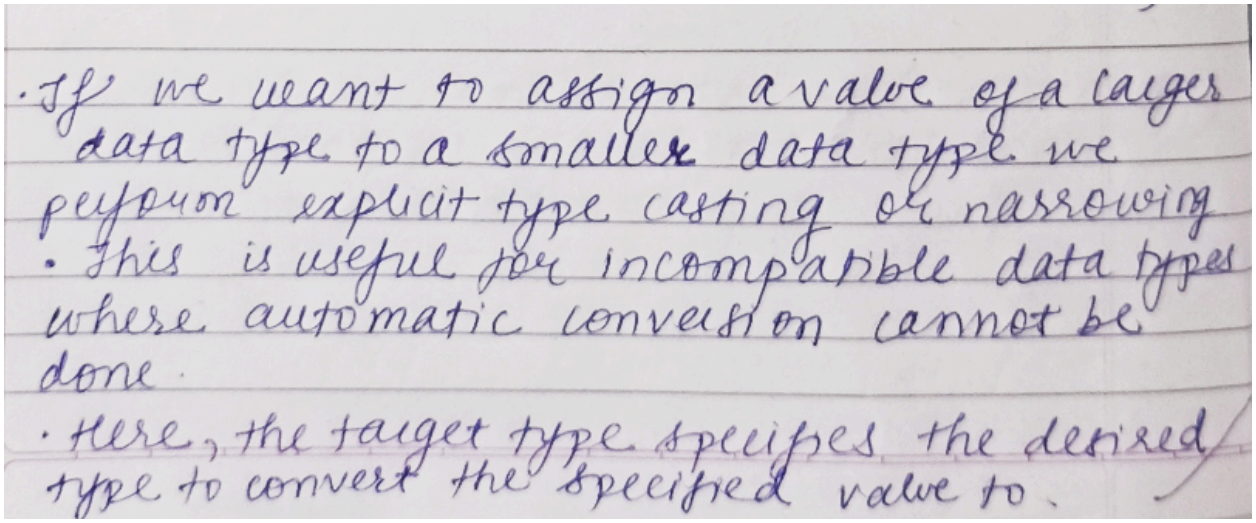
The final keyword in Java is a non-access modifier that prevents classes, methods, and variables from being changed or inherited. Its used to make code more secure, efficient and predictable.
Here, are some ways the final keyword is used in Java,
Final variables : once initialized, the value of a final variable cannot be changed. This is useful for creating constants.
Final methods : A final method cannot be overridden by subclasses. This can be used to enforce security or immutability in classes.
Final classes : A final class cannot be subclassed by other classes. This is often used to prevent classes from being extended or modified, especially utility classes.
Thread safety : when a final variable is initialized in a constructor, it is guaranteed to be initialized before other threads can access the object.
compiler optimization : The final variable can help the compiler peform optimizations, such as inlining constant values or methods.
security : The final keyword can help ~~the compiler peform optimizations~~, prevent certain types of attacks, like man-in-the-middle attacks, by preventing certain methods or variables from being modified.

3. What are narrowing and widening conversions in Java?

Widening conversion:

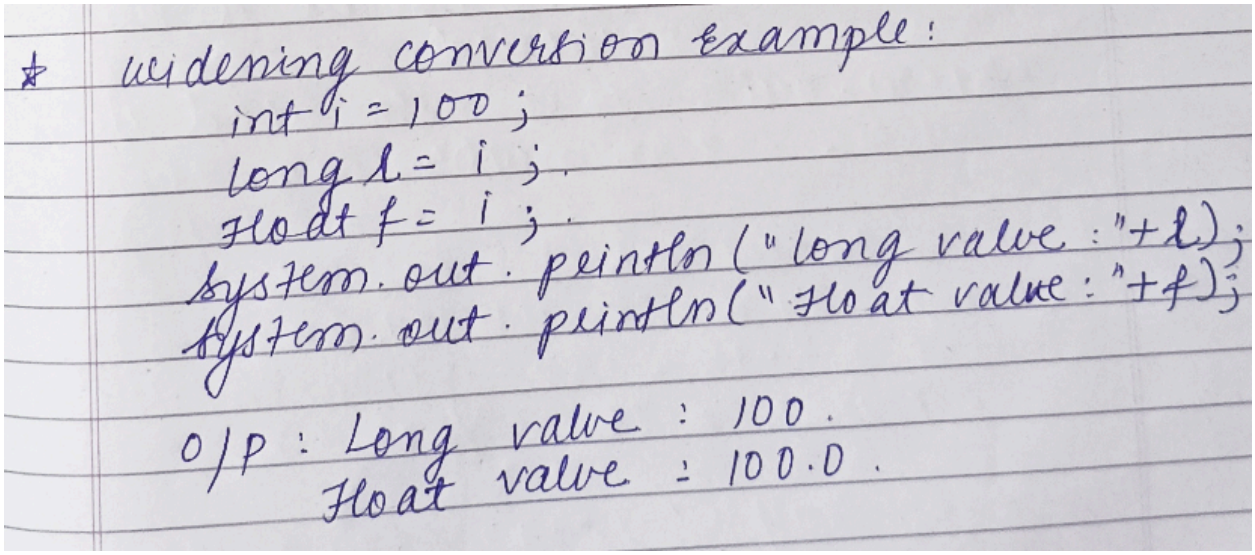


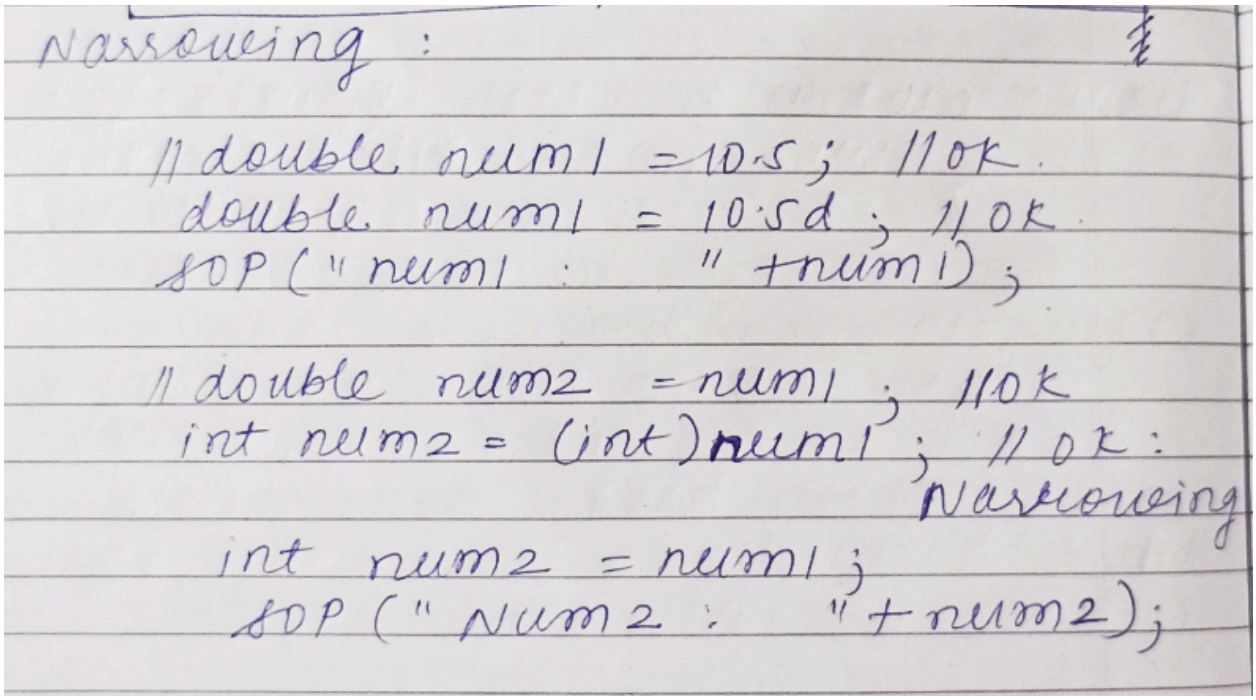
Narrowing conversion:



Double → Float → Long → Int → Short
→ Byte.
Narrowing or Explicit conversion

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

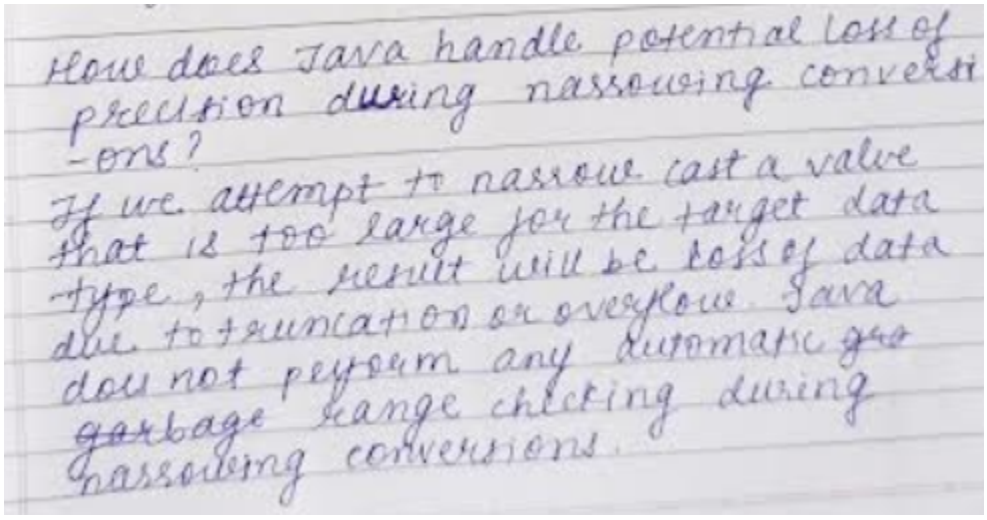




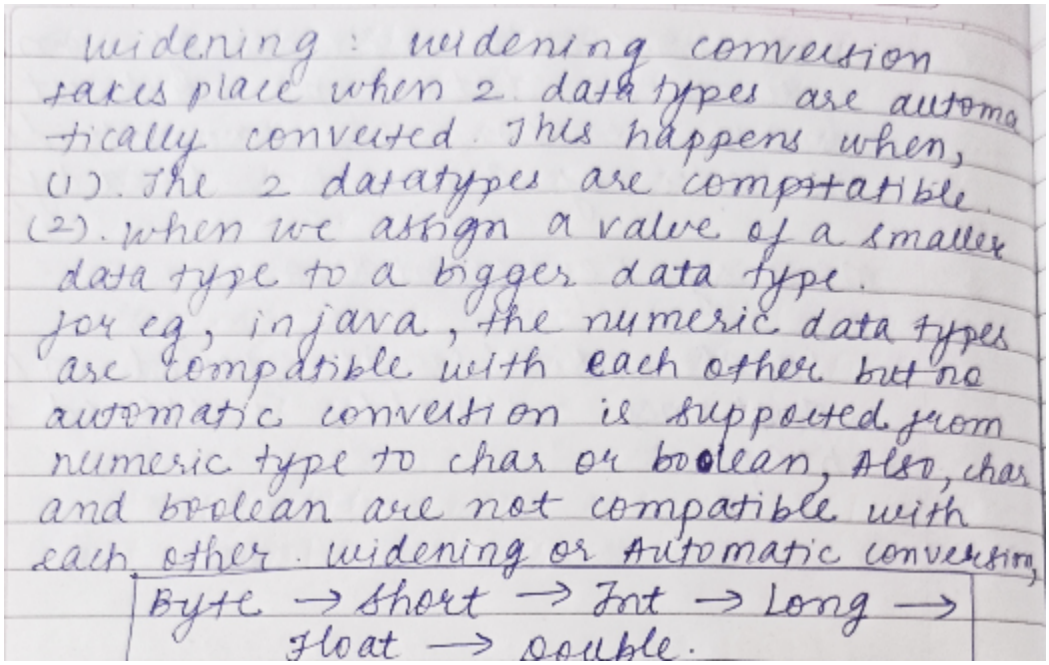
O/P: Num1: 10

Num2: 10

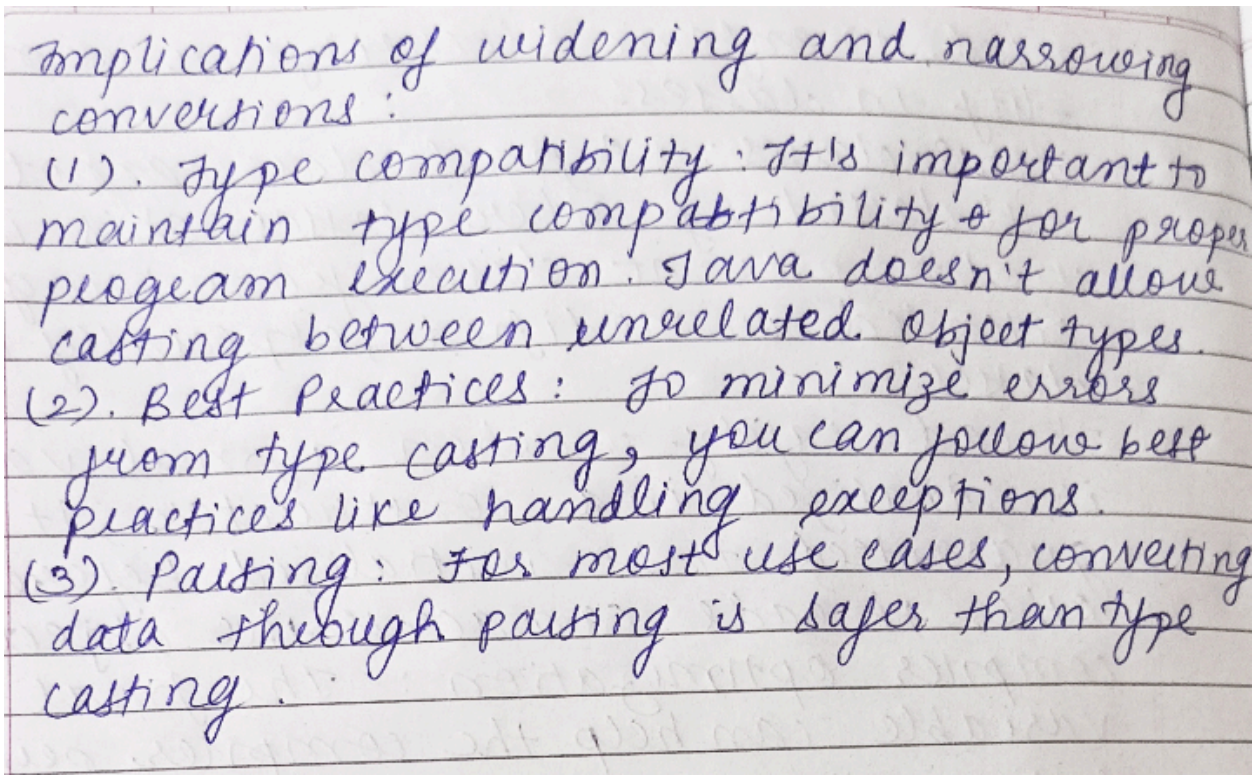
5. How does Java handle potential loss of precision during narrowing conversions?


How does Java handle potential loss of precision during narrowing conversions?
If we attempt to narrow cast a value that is too large for the target data type, the result will be loss of data due to truncation or overflow. Java does not perform any automatic range checking during narrowing conversions.

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


widening : widening conversion takes place when 2 data types are automatically converted. This happens when,
(1). The 2 datatypes are comptatible.
(2). when we assign a value of a smaller data type to a bigger data type.
for eg, in java, the numeric data types are compatible with each other but no automatic conversion is supported from numeric type to char or boolean. Also, char and boolean are not compatible with each other. widening or Automatic conversion,
Byte → short → Int → Long → float → double.

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


Implications of widening and narrowing conversions:
(1). Type compatibility: It's important to maintain type compatibility for proper program execution. Java doesn't allow casting between unrelated object types.
(2). Best Practices: To minimize errors from type casting, you can follow best practices like handling exceptions.
(3). Parsing: For most use cases, converting data through parsing is safer than type casting.