

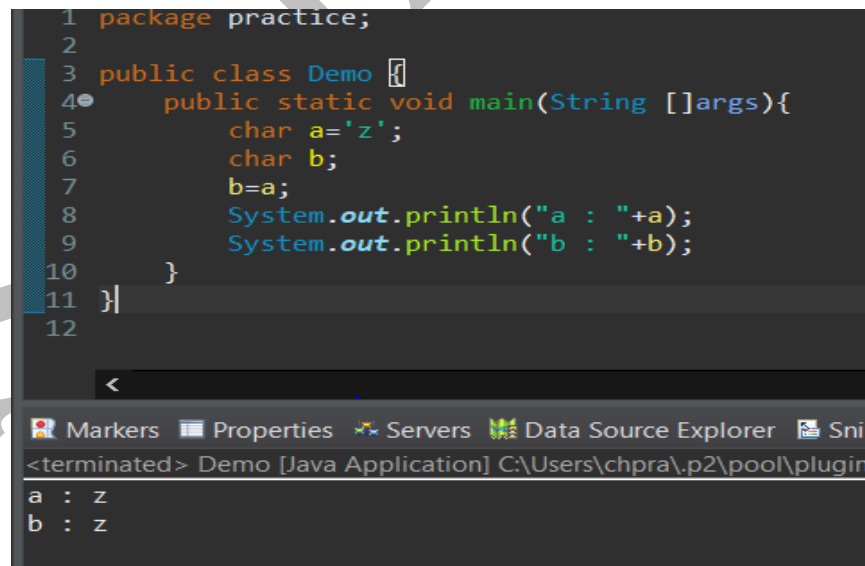
## PRIMITIVE DATATYPES TYPECASTING

### Conversion of data from char-to-char datatype:

#### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        char a='z';  
        char b;  
        b=a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

#### Output:

A screenshot of an IDE window. The top pane shows the Java code from the previous block, with line numbers 1 through 12. The bottom pane shows the output of the program, which is "a : z" followed by "b : z" on the next line. The IDE interface includes tabs for "Markers", "Properties", "Servers", "Data Source Explorer", and "Snippets". The title bar of the bottom pane reads "<terminated> Demo [Java Application] C:\Users\chpra\.p2\pool\plugin".

```
1 package practice;  
2  
3 public class Demo {  
4     public static void main(String []args){  
5         char a='z';  
6         char b;  
7         b=a;  
8         System.out.println("a : "+a);  
9         System.out.println("b : "+b);  
10    }  
11 }  
12  
  
<terminated> Demo [Java Application] C:\Users\chpra\.p2\pool\plugin  
a : z  
b : z
```

#### Conclusion:

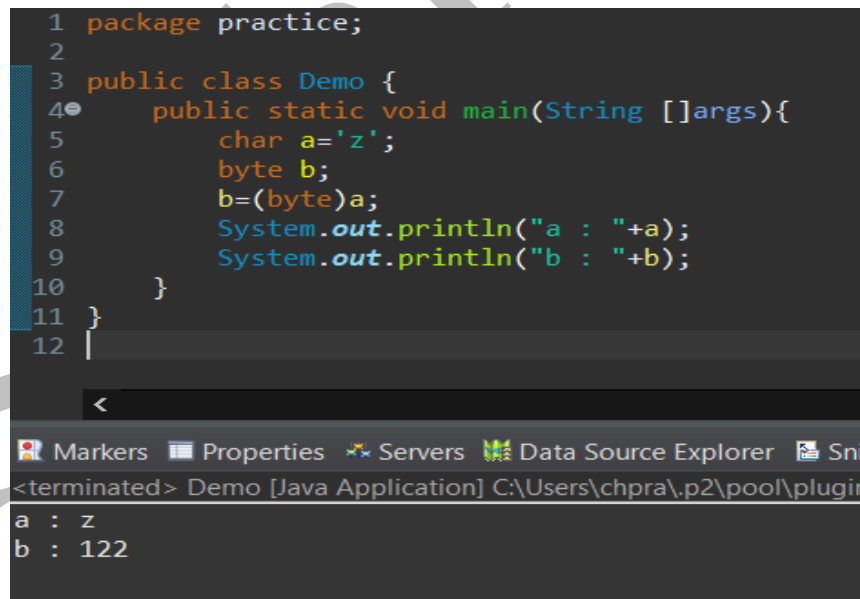
char-to-char conversion is not required.

## Conversion of data from char-to-byte datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        char a='z';  
        byte b;  
        b=(byte)a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

A screenshot of a Java IDE window. The top pane shows the source code for a class named 'Demo' in the 'practice' package. The code defines a 'main' method that declares a 'char' variable 'a' with the value 'z', a 'byte' variable 'b', and performs an explicit cast from 'a' to 'b' using '(byte)a'. It then prints the values of 'a' and 'b'. The bottom pane shows the IDE's output window, which displays the results of the program execution: 'a : z' and 'b : 122'. The IDE interface includes a toolbar with icons for Markers, Properties, Servers, and Data Source Explorer, and a status bar at the bottom indicating the current file and project path.

```
1 package practice;  
2  
3 public class Demo {  
4     public static void main(String []args){  
5         char a='z';  
6         byte b;  
7         b=(byte)a;  
8         System.out.println("a : "+a);  
9         System.out.println("b : "+b);  
10    }  
11 }  
12 |
```

<terminated> Demo [Java Application] C:\Users\chpra\.p2\pool\plugin

a : z  
b : 122

### Conclusion:

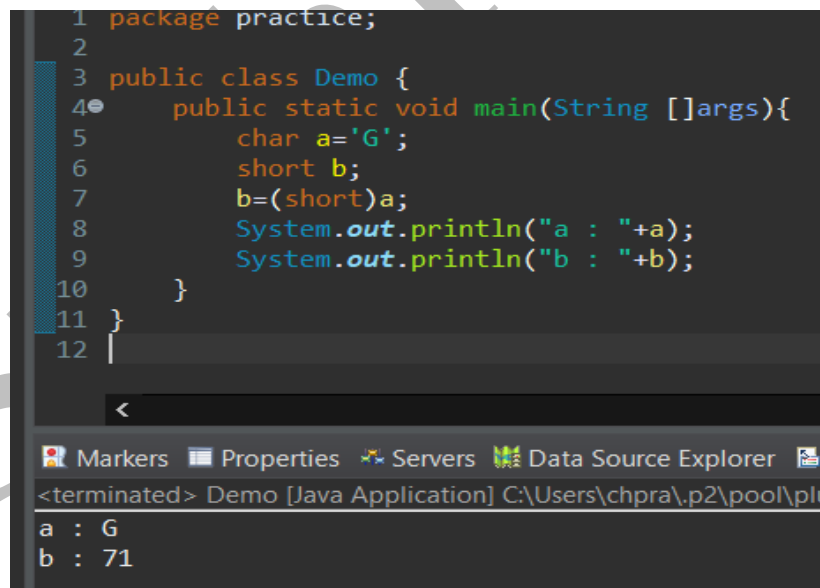
char-to-byte conversion is done through explicit type casting.

## Conversion of data from char-to-short datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        char a='G';  
        short b;  
        b=(short)a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

A screenshot of a Java IDE. The top pane shows the source code for a class named 'Demo' in the 'practice' package. The code defines a 'main' method where a character 'G' is cast to a short integer, resulting in the value 71. The bottom pane shows the IDE's interface with tabs for 'Markers', 'Properties', 'Servers', and 'Data Source Explorer'. Below these is a console window titled '<terminated> Demo [Java Application] C:\Users\chpra\p2\pool\plu' which displays the output: 'a : G' and 'b : 71'.

```
1 package practice;  
2  
3 public class Demo {  
4     public static void main(String []args){  
5         char a='G';  
6         short b;  
7         b=(short)a;  
8         System.out.println("a : "+a);  
9         System.out.println("b : "+b);  
10    }  
11 }  
12 |
```

<terminated> Demo [Java Application] C:\Users\chpra\p2\pool\plu  
a : G  
b : 71

### Conclusion:

char-to-short conversion is done through explicit type casting.

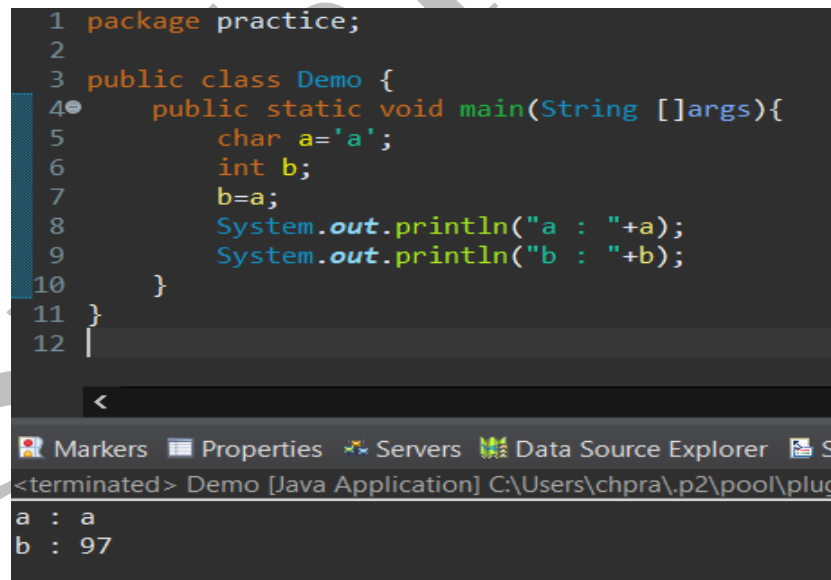
## Conversion of data from char-to-int datatype:

### Program:

```
package practice;

public class Demo {
    public static void main(String []args){
        char a='a';
        int b;
        b=a;
        System.out.println("a : "+a);
        System.out.println("b : "+b);
    }
}
```

### Output:

A screenshot of a Java IDE window. The top pane shows the source code for a class named 'Demo' in the 'practice' package. The code defines a 'main' method where a character 'a' is assigned the value 'a', an integer 'b' is declared, 'b' is assigned the value of 'a' (demonstrating implicit casting), and both are printed. The bottom pane shows the output of the program: 'a : a' and 'b : 97'. The IDE interface includes a toolbar with icons for Markers, Properties, Servers, and Data Source Explorer, and a status bar at the bottom indicating the file path and application name.

```
1 package practice;
2
3 public class Demo {
4     public static void main(String []args){
5         char a='a';
6         int b;
7         b=a;
8         System.out.println("a : "+a);
9         System.out.println("b : "+b);
10    }
11 }
12 |
```

<terminated> Demo [Java Application] C:\Users\chpra\.p2\pool\plug

a : a  
b : 97

### Conclusion:

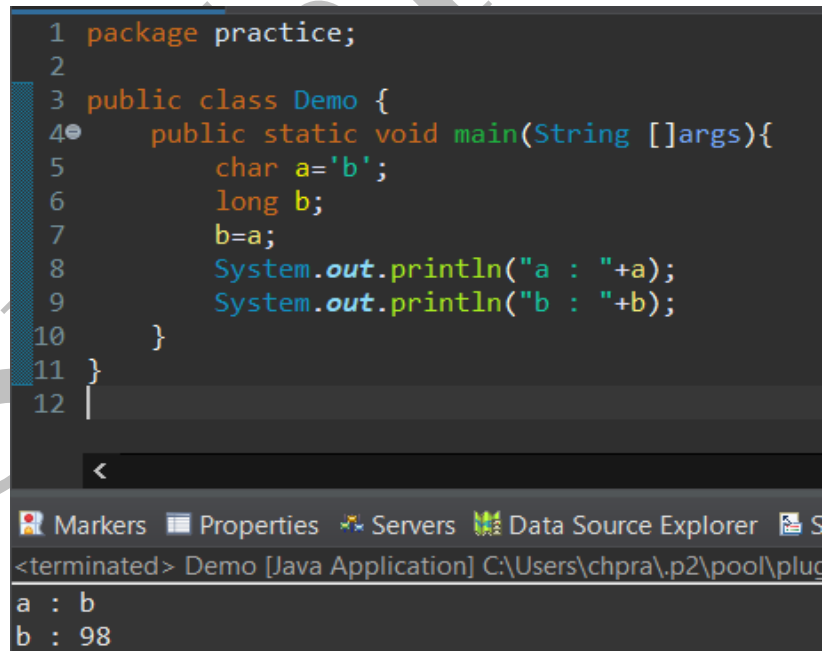
char-to-int conversion is done through implicit type casting.

## Conversion of data from char-to-long datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        char a='b';  
        long b;  
        b=a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

A screenshot of an IDE window showing the Java code from the previous block. The code is displayed in a dark-themed editor with line numbers 1 through 12. Below the editor, the IDE's interface includes tabs for 'Markers', 'Properties', 'Servers', and 'Data Source Explorer'. A console window at the bottom shows the output of the program: 'a : b' and 'b : 98'. The title bar of the console window reads '<terminated> Demo [Java Application] C:\Users\chpra\.p2\pool\plug...'.

```
1 package practice;  
2  
3 public class Demo {  
4     public static void main(String []args){  
5         char a='b';  
6         long b;  
7         b=a;  
8         System.out.println("a : "+a);  
9         System.out.println("b : "+b);  
10    }  
11 }  
12 |
```

<terminated> Demo [Java Application] C:\Users\chpra\.p2\pool\plug...

a : b  
b : 98

### Conclusion:

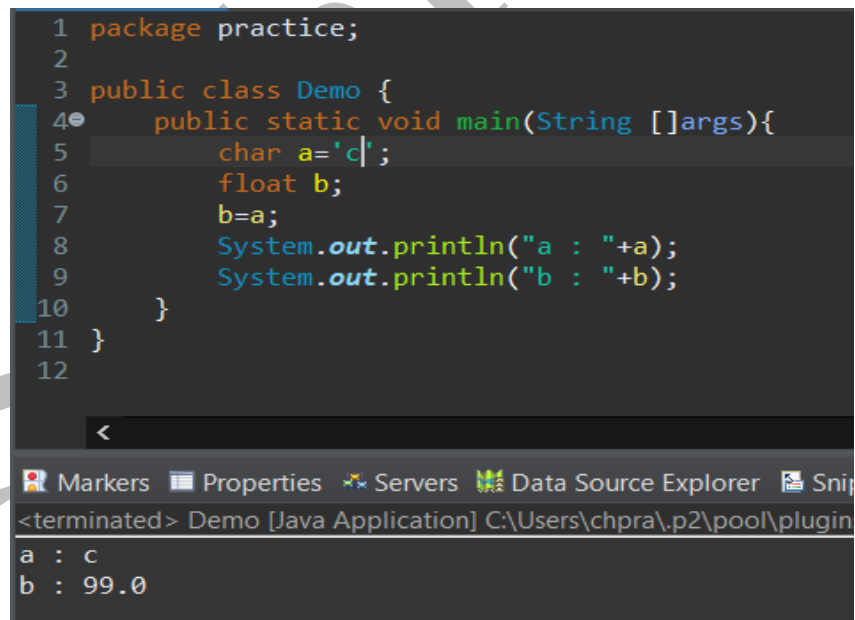
char-to-long conversion is done through implicit type casting.

## Conversion of data from char-to-float datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        char a='c';  
        float b;  
        b=a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:



The screenshot shows an IDE window with the following code:

```
1 package practice;  
2  
3 public class Demo {  
4     public static void main(String []args){  
5         char a='c';  
6         float b;  
7         b=a;  
8         System.out.println("a : "+a);  
9         System.out.println("b : "+b);  
10    }  
11 }  
12
```

Below the code editor, the output is displayed:

```
<terminated> Demo [Java Application] C:\Users\chpra\.p2\pool\plugin  
a : c  
b : 99.0
```

### Conclusion:

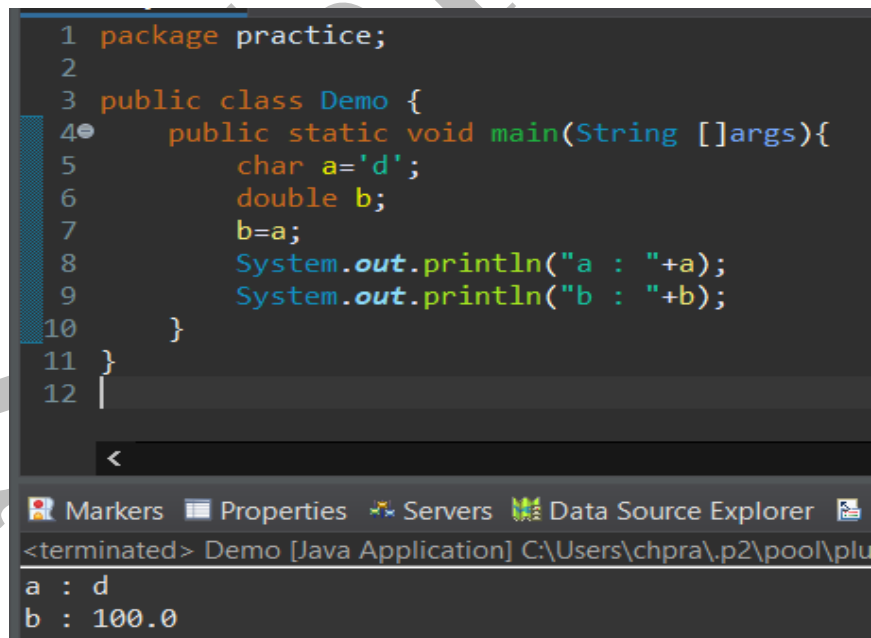
char-to-float conversion is done through implicit type casting.

## Conversion of data from char-to-double datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        char a='d';  
        double b;  
        b=a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:



The screenshot shows an IDE window with the following Java code:

```
1 package practice;  
2  
3 public class Demo {  
4     public static void main(String []args){  
5         char a='d';  
6         double b;  
7         b=a;  
8         System.out.println("a : "+a);  
9         System.out.println("b : "+b);  
10    }  
11 }  
12 |
```

Below the code editor, the output is displayed:

```
<terminated> Demo [Java Application] C:\Users\chpra\.p2\pool\plu  
a : d  
b : 100.0
```

### Conclusion:

char-to-double conversion is done through implicit type casting.

## Conversion of data from char-to-boolean datatype:

### Program://Implicit type casting

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        char a='d';  
        boolean b;  
        b=a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### }//Explicit type casting

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        char a='d';  
        boolean b;  
        b=(boolean)a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

Compilation error for type casting.

### Conclusion:

char-to-boolean conversion is not possible.

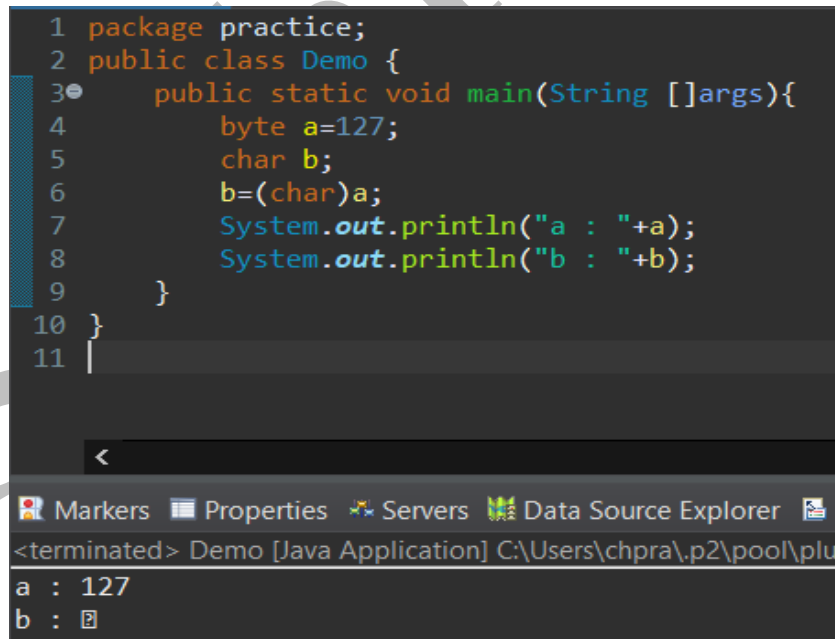


## Conversion of data from byte-to-char datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        byte a=127;  
        char b;  
        b=(char)a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:



The screenshot shows an IDE window with a Java file named 'Demo.java'. The code is as follows:

```
1 package practice;  
2 public class Demo {  
3     public static void main(String []args){  
4         byte a=127;  
5         char b;  
6         b=(char)a;  
7         System.out.println("a : "+a);  
8         System.out.println("b : "+b);  
9     }  
10 }  
11 |
```

Below the code editor, the output console shows the following results:

```
<terminated> Demo [Java Application] C:\Users\chpra\.p2\pool\plug  
a : 127  
b : 7
```

### Conclusion:

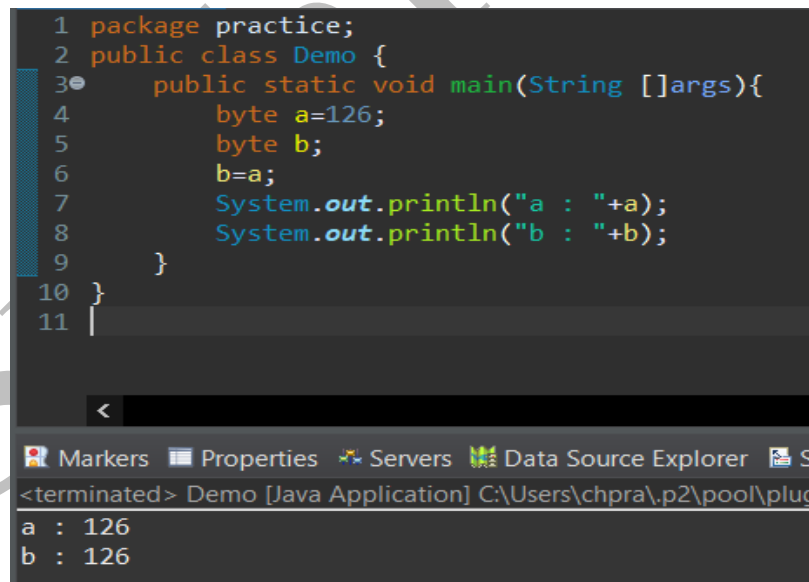
byte-to-char conversion is done through explicit type casting.

## Conversion of data from byte-to-byte datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        byte a=126;  
        byte b;  
        b=a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

A screenshot of a Java IDE. The top part shows a code editor with the following code:

```
1 package practice;  
2 public class Demo {  
3     public static void main(String []args){  
4         byte a=126;  
5         byte b;  
6         b=a;  
7         System.out.println("a : "+a);  
8         System.out.println("b : "+b);  
9     }  
10 }  
11 |
```

The bottom part shows the IDE's console window with the output:

```
<terminated> Demo [Java Application] C:\Users\chpra\.p2\pool\plug  
a : 126  
b : 126
```

### Conclusion:

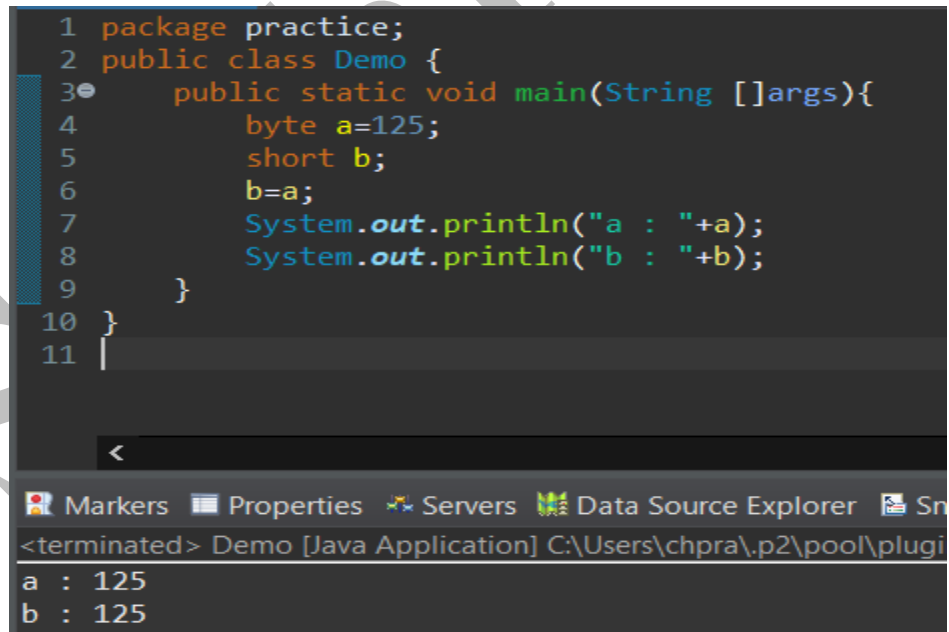
byte-to-byte conversion is not required.

## Conversion of data from byte-to-short datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        byte a=125;  
        short b;  
        b=a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

A screenshot of an IDE window showing the Java code from the previous block. The code is displayed in a dark-themed editor with line numbers 1 through 11. Below the editor, the IDE's interface includes tabs for 'Markers', 'Properties', 'Servers', 'Data Source Explorer', and 'Snippets'. The 'Markers' tab is active, showing a message: '<terminated> Demo [Java Application] C:\Users\chpra\.p2\pool\plugi'. At the bottom of the IDE, the output of the program is shown: 'a : 125' and 'b : 125'.

```
1 package practice;  
2 public class Demo {  
3     public static void main(String []args){  
4         byte a=125;  
5         short b;  
6         b=a;  
7         System.out.println("a : "+a);  
8         System.out.println("b : "+b);  
9     }  
10 }  
11 |
```

<terminated> Demo [Java Application] C:\Users\chpra\.p2\pool\plugi

a : 125  
b : 125

### Conclusion:

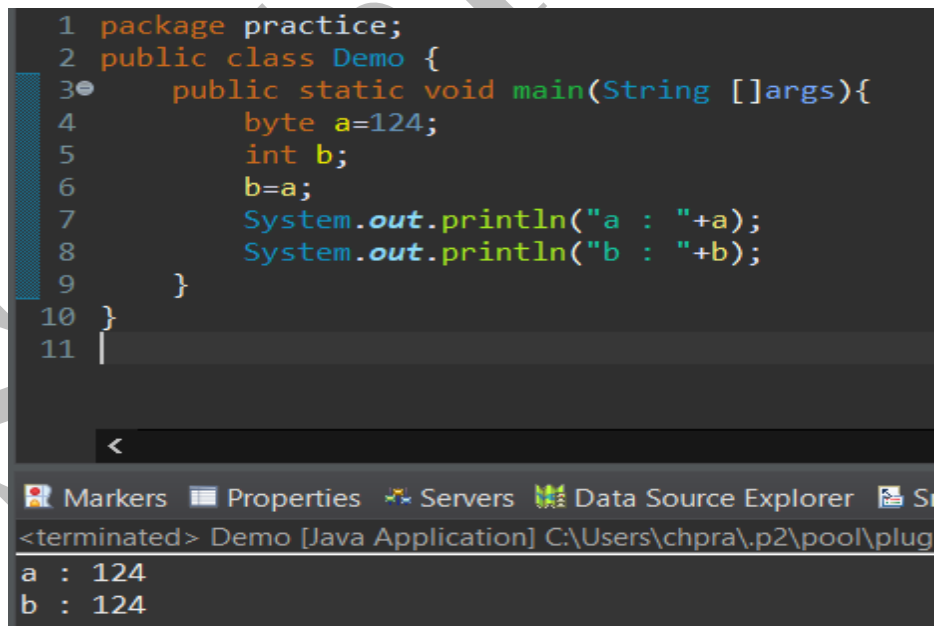
byte-to-short conversion is done through implicit type casting.

## Conversion of data from byte-to-int datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        byte a=124;  
        int b;  
        b=a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:



The screenshot shows a Java IDE with the following code in the editor:

```
1 package practice;  
2 public class Demo {  
3     public static void main(String []args){  
4         byte a=124;  
5         int b;  
6         b=a;  
7         System.out.println("a : "+a);  
8         System.out.println("b : "+b);  
9     }  
10 }  
11 |
```

Below the editor, the console output is displayed:

```
<terminated> Demo [Java Application] C:\Users\chpra\.p2\pool\plug  
a : 124  
b : 124
```

### Conclusion:

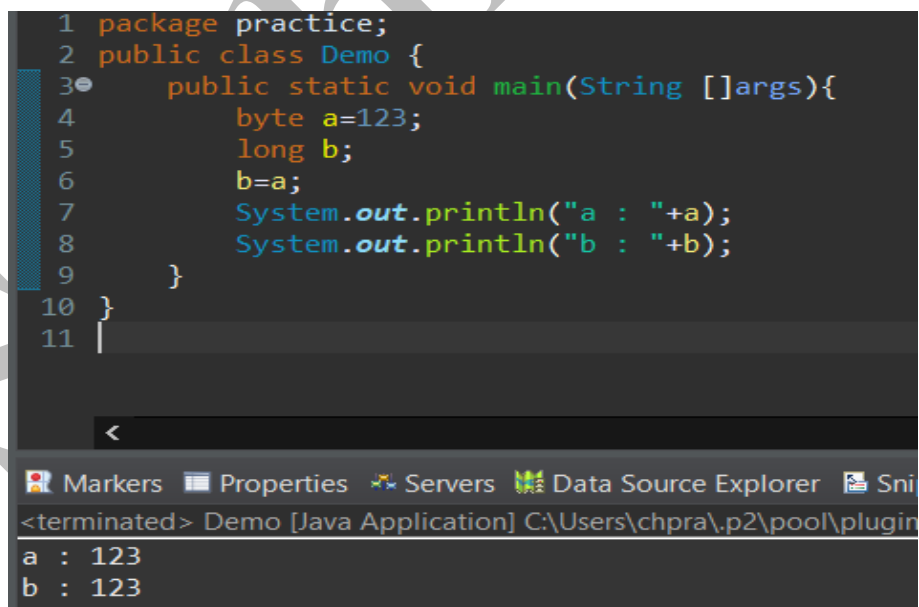
byte-to-int conversion is done through implicit type casting.

## Conversion of data from byte-to-long datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        byte a=123;  
        long b;  
        b=a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:



The screenshot shows an IDE window with the following code:

```
1 package practice;  
2 public class Demo {  
3     public static void main(String []args){  
4         byte a=123;  
5         long b;  
6         b=a;  
7         System.out.println("a : "+a);  
8         System.out.println("b : "+b);  
9     }  
10 }  
11 |
```

Below the code editor, the output is displayed:

```
<terminated> Demo [Java Application] C:\Users\chpra\.p2\pool\plugin  
a : 123  
b : 123
```

### Conclusion:

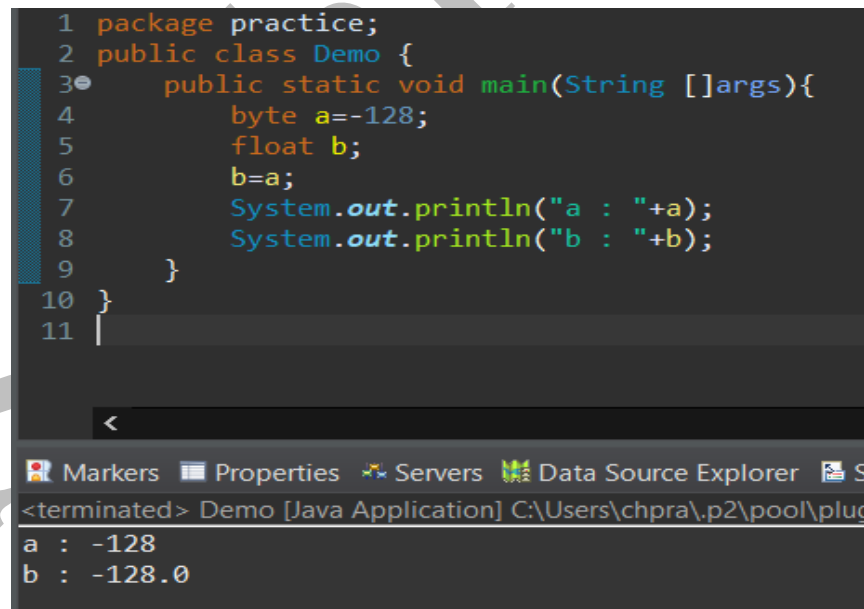
byte-to-long conversion is done through implicit type casting.

## Conversion of data from byte-to-float datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        byte a=-128;  
        float b;  
        b=a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

A screenshot of an IDE window showing the Java code from the previous block. The code is highlighted with syntax coloring. Below the code editor, the output console is visible, showing the execution results. The output is:  
a : -128  
b : -128.0  
The IDE interface includes a toolbar with icons for Markers, Properties, Servers, and Data Source Explorer. The title bar of the window reads "<terminated> Demo [Java Application] C:\Users\chpra\p2\pool\plug".

```
1 package practice;  
2 public class Demo {  
3     public static void main(String []args){  
4         byte a=-128;  
5         float b;  
6         b=a;  
7         System.out.println("a : "+a);  
8         System.out.println("b : "+b);  
9     }  
10 }  
11 |  
  
<  
Markers Properties Servers Data Source Explorer  
<terminated> Demo [Java Application] C:\Users\chpra\p2\pool\plug  
a : -128  
b : -128.0
```

### Conclusion:

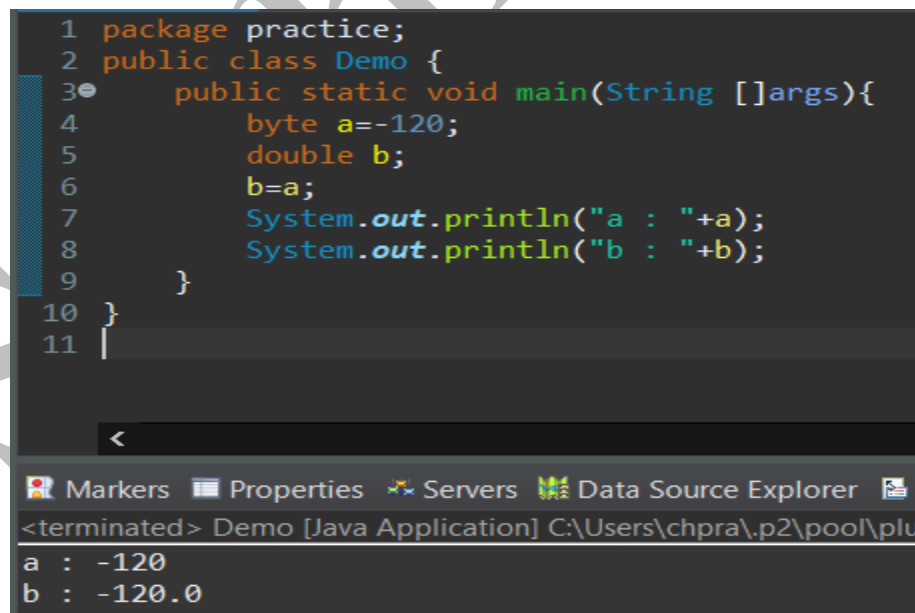
byte-to-float conversion is done through implicit type casting.

## Conversion of data from byte-to-double datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        byte a=-120;  
        double b;  
        b=a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

A screenshot of an IDE window showing the Java code from the previous block. The code is displayed in a dark-themed editor with line numbers 1 through 11. Below the editor, the IDE's interface includes tabs for 'Markers', 'Properties', 'Servers', and 'Data Source Explorer'. A console window at the bottom shows the output of the program: 'a : -120' and 'b : -120.0'. The title bar of the IDE window indicates the file path: 'C:\Users\chpra\p2\pool\plu...'.

```
1 package practice;  
2 public class Demo {  
3     public static void main(String []args){  
4         byte a=-120;  
5         double b;  
6         b=a;  
7         System.out.println("a : "+a);  
8         System.out.println("b : "+b);  
9     }  
10 }  
11 |
```

Markers Properties Servers Data Source Explorer

<terminated> Demo [Java Application] C:\Users\chpra\p2\pool\plu

a : -120  
b : -120.0

### Conclusion:

byte-to-double conversion is done through implicit type casting.

## Conversion of data from byte-to-boolean datatype:

### Program://Implicit type casting

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        byte a=-120;  
        boolean b;  
        b=a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### }//Explicit type casting

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        byte a=-120;  
        boolean b;  
        b=(boolean)a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

Compilation error for type casting.

### Conclusion:

byte-to-boolean conversion is not possible.

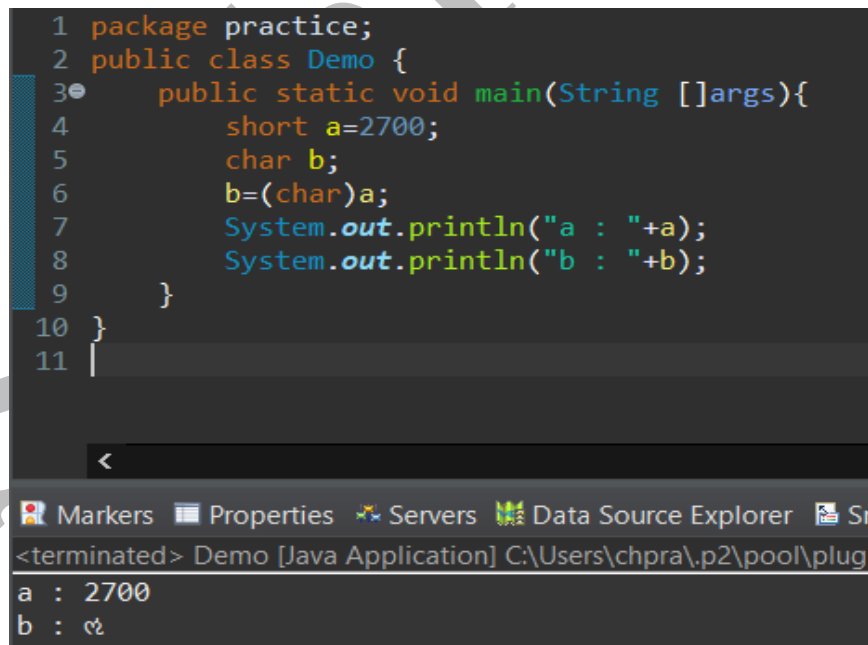


## Conversion of data from short-to-char datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        short a=2700;  
        char b;  
        b=(char)a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:



The screenshot shows an IDE window with the following code:

```
1 package practice;  
2 public class Demo {  
3     public static void main(String []args){  
4         short a=2700;  
5         char b;  
6         b=(char)a;  
7         System.out.println("a : "+a);  
8         System.out.println("b : "+b);  
9     }  
10 }  
11 |
```

Below the code editor, the output is displayed:

```
<terminated> Demo [Java Application] C:\Users\chpra\.p2\pool\plug  
a : 2700  
b :  
```

### Conclusion:

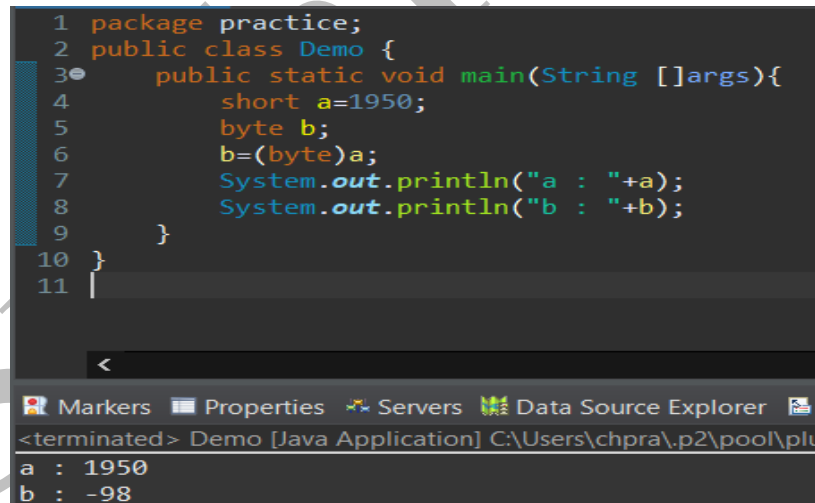
short-to-char conversion is done through explicit type casting.

## Conversion of data from short-to-byte datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        short a=1950;  
        byte b;  
        b=(byte)a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

A screenshot of an IDE window showing the Java code from the previous block. The code is highlighted with syntax coloring. Below the code editor, the IDE's output console is visible, showing the execution results. The output shows 'a : 1950' and 'b : -98'. The IDE interface includes tabs for 'Markers', 'Properties', 'Servers', and 'Data Source Explorer'. The title bar of the IDE window indicates the file path: 'C:\Users\chpra\p2\pool\plu'.

```
1 package practice;  
2 public class Demo {  
3     public static void main(String []args){  
4         short a=1950;  
5         byte b;  
6         b=(byte)a;  
7         System.out.println("a : "+a);  
8         System.out.println("b : "+b);  
9     }  
10 }  
11 |  
  
<terminated> Demo [Java Application] C:\Users\chpra\p2\pool\plu  
a : 1950  
b : -98
```

### Conclusion:

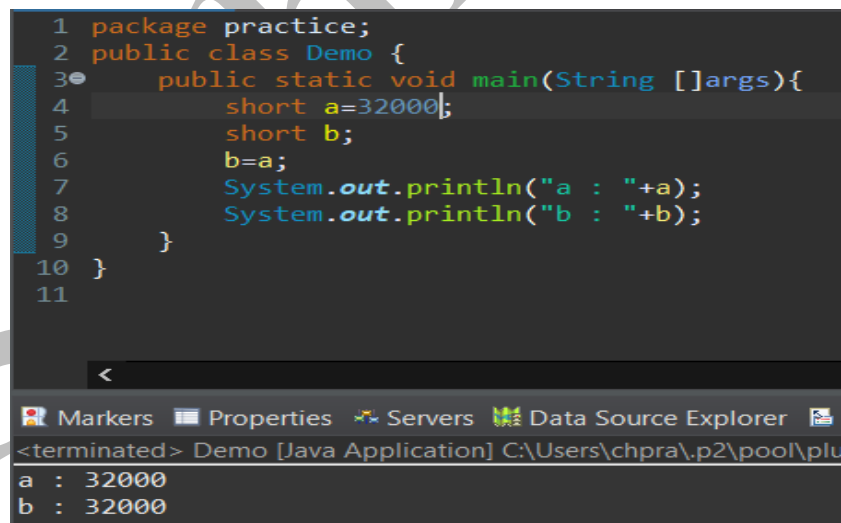
short-to-byte conversion is done through explicit type casting.

## Conversion of data from short-to-short datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        short a=32000;  
        short b;  
        b=a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

A screenshot of an IDE window showing the Java code from the previous block. The code is highlighted with syntax coloring. Below the code editor, the output console is visible, showing the execution results.

```
1 package practice;  
2 public class Demo {  
3     public static void main(String []args){  
4         short a=32000;  
5         short b;  
6         b=a;  
7         System.out.println("a : "+a);  
8         System.out.println("b : "+b);  
9     }  
10 }  
11
```

Markers Properties Servers Data Source Explorer

<terminated> Demo [Java Application] C:\Users\chpra.p2\pool\plu

a : 32000  
b : 32000

### Conclusion:

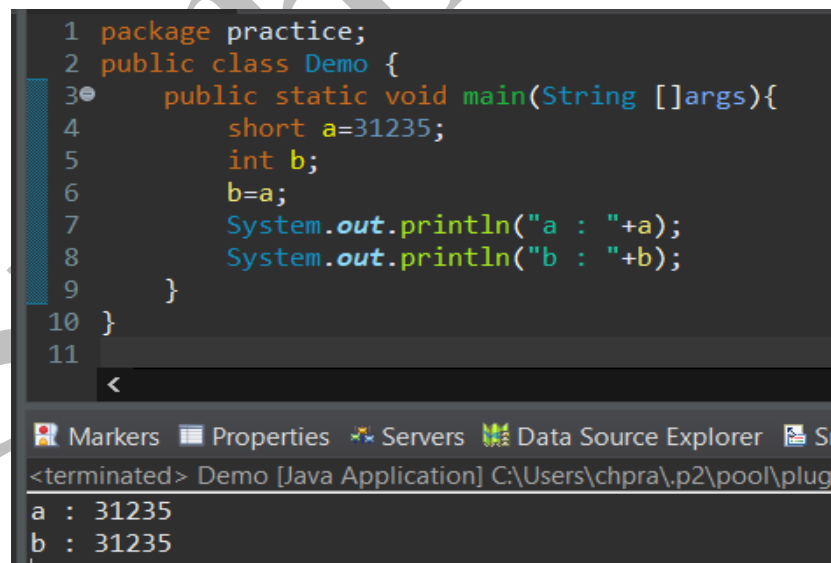
short-to-short conversion is not required.

## Conversion of data from short-to-int datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        short a=31235;  
        int b;  
        b=a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

A screenshot of an IDE window. The top pane shows the Java code from the previous block, with line numbers 1 through 11. The bottom pane shows the output of the program, which is "a : 31235" followed by "b : 31235". The IDE interface includes a toolbar with icons for Markers, Properties, Servers, Data Source Explorer, and a file explorer showing the project path C:\Users\chpra\.p2\pool\plug.

```
1 package practice;  
2 public class Demo {  
3     public static void main(String []args){  
4         short a=31235;  
5         int b;  
6         b=a;  
7         System.out.println("a : "+a);  
8         System.out.println("b : "+b);  
9     }  
10 }  
11  
<  
  
Markers Properties Servers Data Source Explorer  
<terminated> Demo [Java Application] C:\Users\chpra\.p2\pool\plug  
a : 31235  
b : 31235
```

### Conclusion:

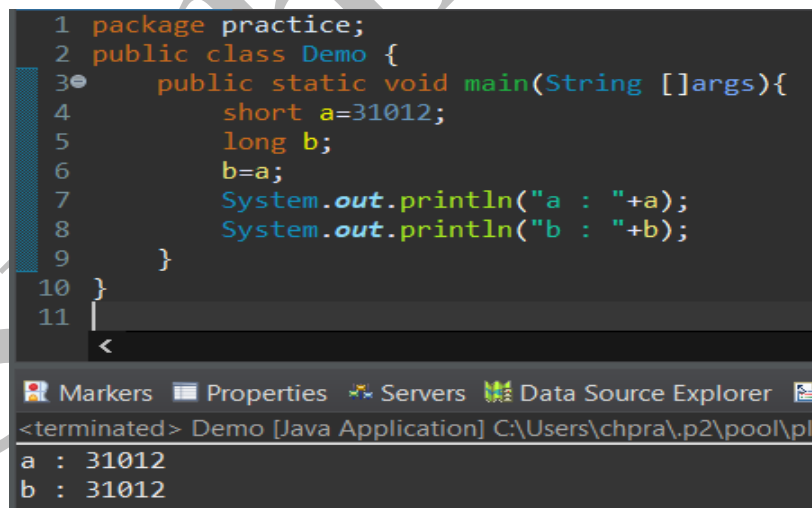
short-to-int conversion is done through implicit type casting.

## Conversion of data from short-to-long datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        short a=31012;  
        long b;  
        b=a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

A screenshot of an IDE window showing the Java code from the previous block. The code is highlighted with syntax coloring. Below the code editor, the IDE's console window is visible, showing the output of the program: "a : 31012" and "b : 31012". The IDE interface includes tabs for Markers, Properties, Servers, and Data Source Explorer.

```
1 package practice;  
2 public class Demo {  
3     public static void main(String []args){  
4         short a=31012;  
5         long b;  
6         b=a;  
7         System.out.println("a : "+a);  
8         System.out.println("b : "+b);  
9     }  
10 }  
11
```

Markers Properties Servers Data Source Explorer

<terminated> Demo [Java Application] C:\Users\chpra\.p2\pool\pl

a : 31012  
b : 31012

### Conclusion:

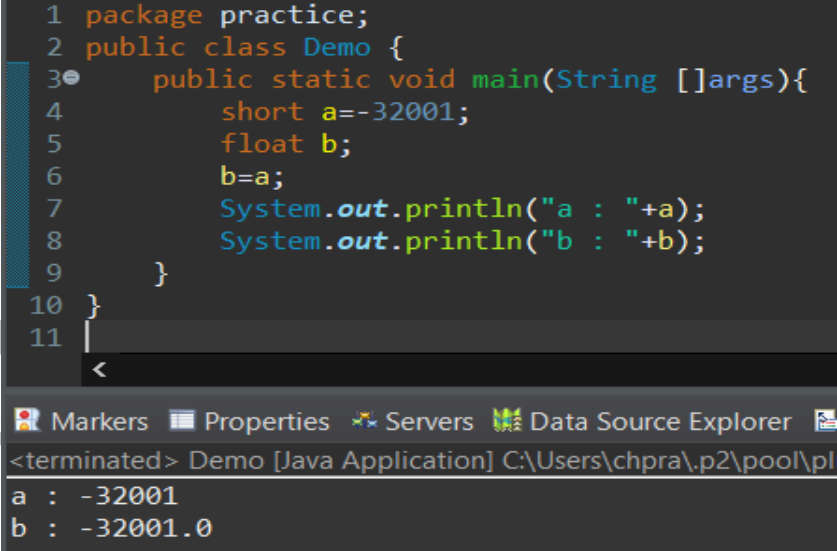
short-to-long conversion is done through implicit type casting.

## Conversion of data from short-to-float datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        short a=-32001;  
        float b;  
        b=a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:



The screenshot shows a Java IDE with a dark theme. The editor displays the code from the previous block. Below the editor, the 'Run' button is visible. The console output shows the program's execution results.

```
1 package practice;  
2 public class Demo {  
3     public static void main(String []args){  
4         short a=-32001;  
5         float b;  
6         b=a;  
7         System.out.println("a : "+a);  
8         System.out.println("b : "+b);  
9     }  
10 }  
11 |  
<
```

Markers Properties Servers Data Source Explorer

<terminated> Demo [Java Application] C:\Users\chpra\.p2\pool\pl

a : -32001  
b : -32001.0

### Conclusion:

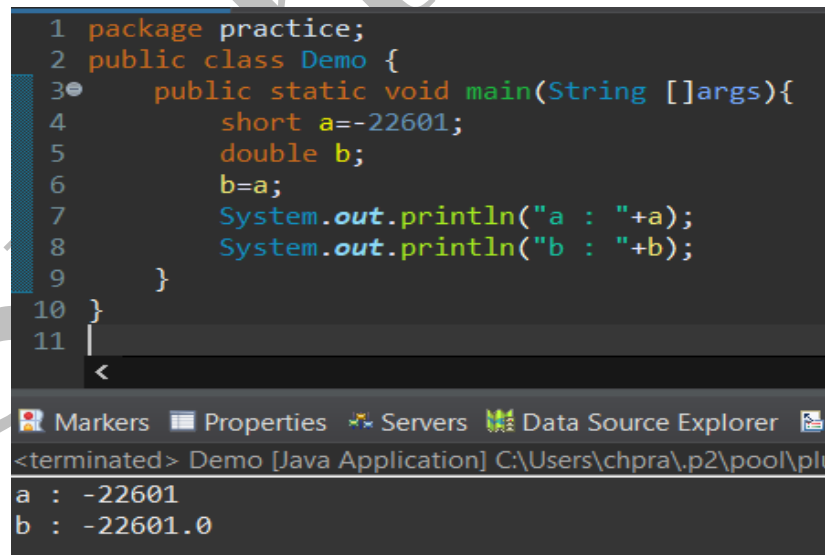
short-to-float conversion is done through implicit type casting.

## Conversion of data from short-to-double datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        short a=-22601;  
        double b;  
        b=a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

A screenshot of a Java IDE window. The top pane shows the source code for a class named 'Demo' in the 'practice' package. The code defines a 'main' method where a 'short' variable 'a' is assigned the value -22601, a 'double' variable 'b' is declared, 'b' is assigned the value of 'a', and both are printed. The bottom pane shows the IDE's interface with tabs for 'Markers', 'Properties', 'Servers', and 'Data Source Explorer'. Below these is a console window titled '<terminated> Demo [Java Application] C:\Users\chpra\p2\pool\pl...' which displays the output: 'a : -22601' and 'b : -22601.0'.

```
1 package practice;  
2 public class Demo {  
3     public static void main(String []args){  
4         short a=-22601;  
5         double b;  
6         b=a;  
7         System.out.println("a : "+a);  
8         System.out.println("b : "+b);  
9     }  
10 }  
11
```

<terminated> Demo [Java Application] C:\Users\chpra\p2\pool\pl...

a : -22601  
b : -22601.0

### Conclusion:

short-to-double conversion is done through implicit type casting.

## Conversion of data from short-to-boolean datatype:

### Program://Implicit type casting

```
package practice;  
public class Demo {  
    public static void main(String []args){  
        short a=3200;  
        boolean b;  
        b=a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### }//Explicit type casting

```
package practice;  
public class Demo {  
    public static void main(String []args){  
        short a=3200;  
        boolean b;  
        b=(boolean)a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

Compilation error for type casting.

### Conclusion:

short-to-boolean conversion is not possible.

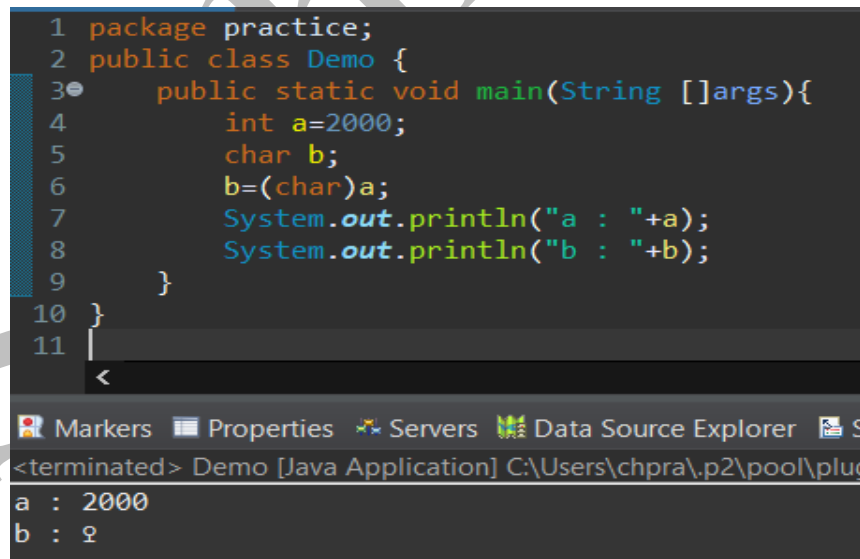


## Conversion of data from int-to-char datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        int a=2000;  
        char b;  
        b=(char)a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:



The screenshot shows a Java IDE with the following code in the editor:

```
1 package practice;  
2 public class Demo {  
3     public static void main(String []args){  
4         int a=2000;  
5         char b;  
6         b=(char)a;  
7         System.out.println("a : "+a);  
8         System.out.println("b : "+b);  
9     }  
10 }  
11
```

The output window at the bottom shows the following results:

```
<terminated> Demo [Java Application] C:\Users\chpra\.p2\pool\plug  
a : 2000  
b :  
```

### Conclusion:

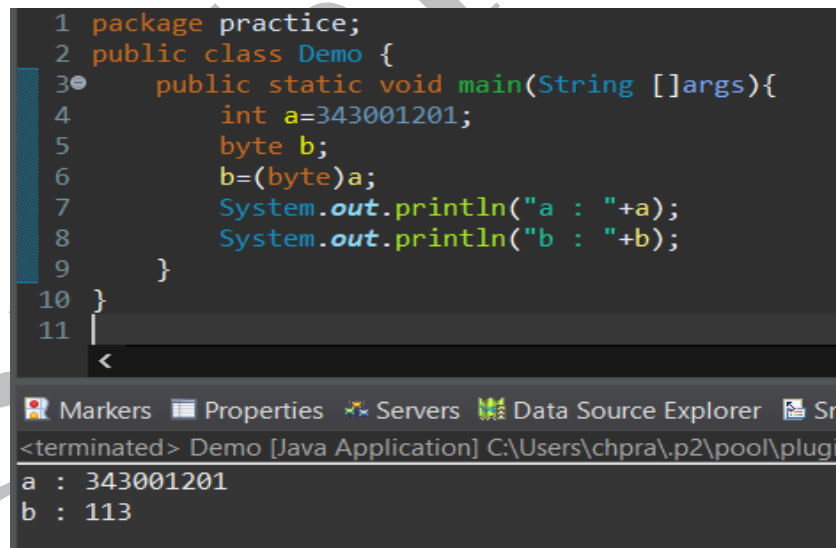
int-to-char conversion is done through explicit type casting.

## Conversion of data from int-to-byte datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        int a=343001201;  
        byte b;  
        b=(byte)a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

A screenshot of a Java IDE window. The top pane shows the source code for a class named 'Demo' in the 'practice' package. The code defines a 'main' method that declares an 'int' variable 'a' with the value 343001201, a 'byte' variable 'b', and performs an explicit cast from 'a' to 'b' using '(byte)a'. It then prints the values of 'a' and 'b'. The bottom pane shows the output of the program, which is 'a : 343001201' followed by 'b : 113'. The IDE interface includes a toolbar with icons for Markers, Properties, Servers, Data Source Explorer, and a console window at the bottom.

```
1 package practice;  
2 public class Demo {  
3     public static void main(String []args){  
4         int a=343001201;  
5         byte b;  
6         b=(byte)a;  
7         System.out.println("a : "+a);  
8         System.out.println("b : "+b);  
9     }  
10 }  
11 |  
<  
  
Markers Properties Servers Data Source Explorer  
<terminated> Demo [Java Application] C:\Users\chpra\p2\pool\plugi  
a : 343001201  
b : 113
```

### Conclusion:

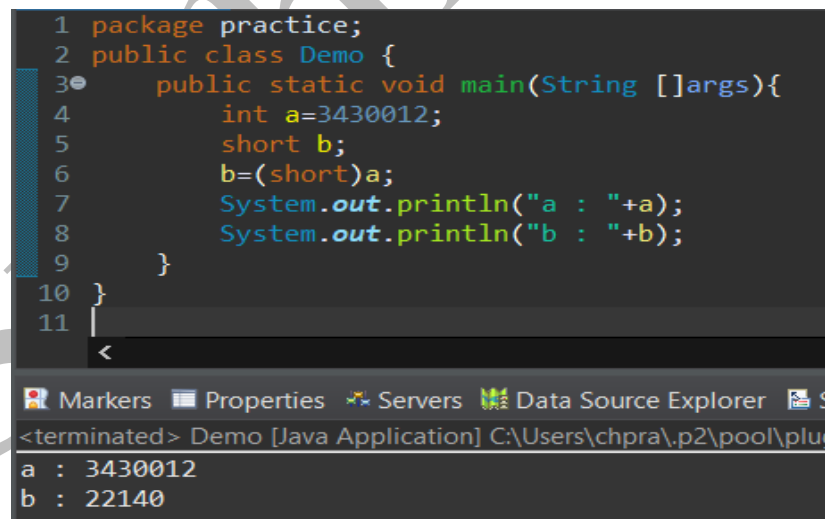
int-to-byte conversion is done through explicit type casting.

## Conversion of data from int-to-short datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        int a=3430012;  
        short b;  
        b=(short)a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:



The screenshot shows a Java IDE with the following code in the editor:

```
1 package practice;  
2 public class Demo {  
3     public static void main(String []args){  
4         int a=3430012;  
5         short b;  
6         b=(short)a;  
7         System.out.println("a : "+a);  
8         System.out.println("b : "+b);  
9     }  
10 }  
11
```

The output console at the bottom shows the following output:

```
<terminated> Demo [Java Application] C:\Users\chpra\p2\pool\plu  
a : 3430012  
b : 22140
```

### Conclusion:

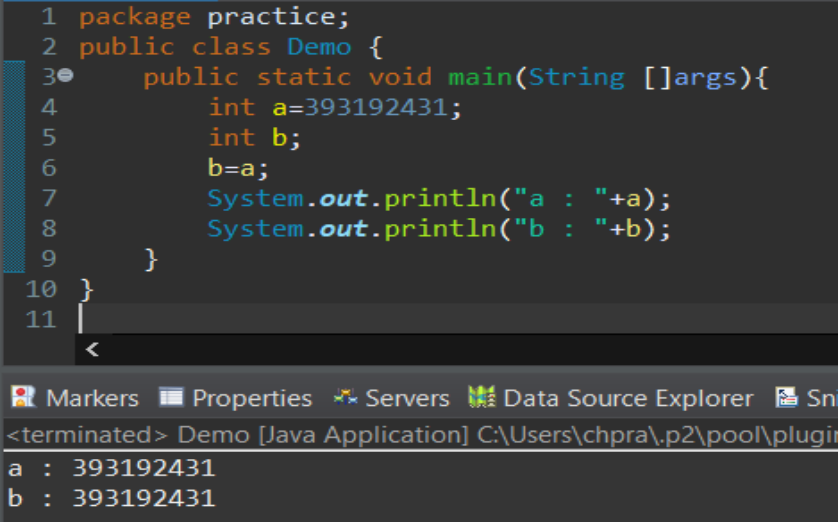
int-to-short conversion is done through explicit type casting.

## Conversion of data from int-to-int datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        int a=393192431;  
        int b;  
        b=a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:



The screenshot shows an IDE window with a Java file named 'Demo.java'. The code is as follows:

```
1 package practice;  
2 public class Demo {  
3     public static void main(String []args){  
4         int a=393192431;  
5         int b;  
6         b=a;  
7         System.out.println("a : "+a);  
8         System.out.println("b : "+b);  
9     }  
10 }  
11
```

Below the code editor, the IDE's console window shows the output of the program:

```
<terminated> Demo [Java Application] C:\Users\chpra\.p2\pool\plugin  
a : 393192431  
b : 393192431
```

### Conclusion:

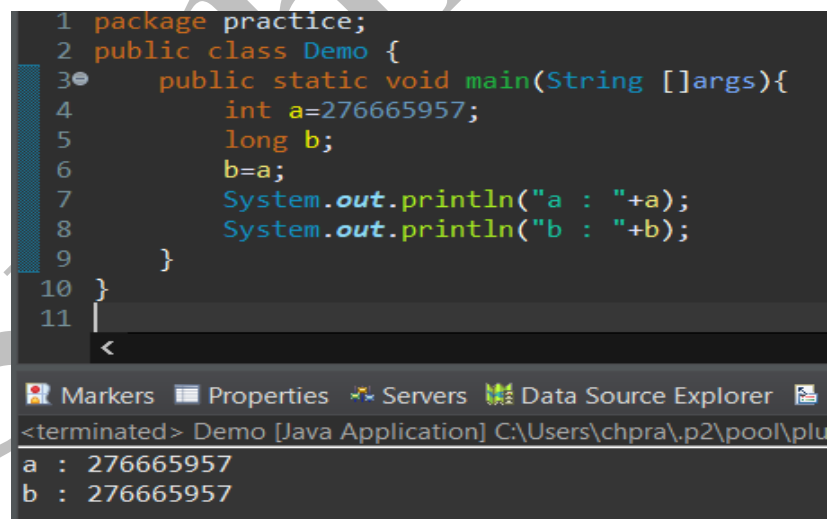
int-to-int conversion is not required.

## Conversion of data from int-to-long datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        int a=276665957;  
        long b;  
        b=a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

A screenshot of an IDE window showing the Java code from the previous block. The code is highlighted with syntax coloring. Below the code editor, the output console is visible, showing the execution results. The output is:

```
a : 276665957  
b : 276665957
```

The IDE interface includes tabs for Markers, Properties, Servers, and Data Source Explorer. The title bar of the window indicates the application is a Java Application named 'Demo' located at 'C:\Users\chpra\.p2\pool\plu'.

### Conclusion:

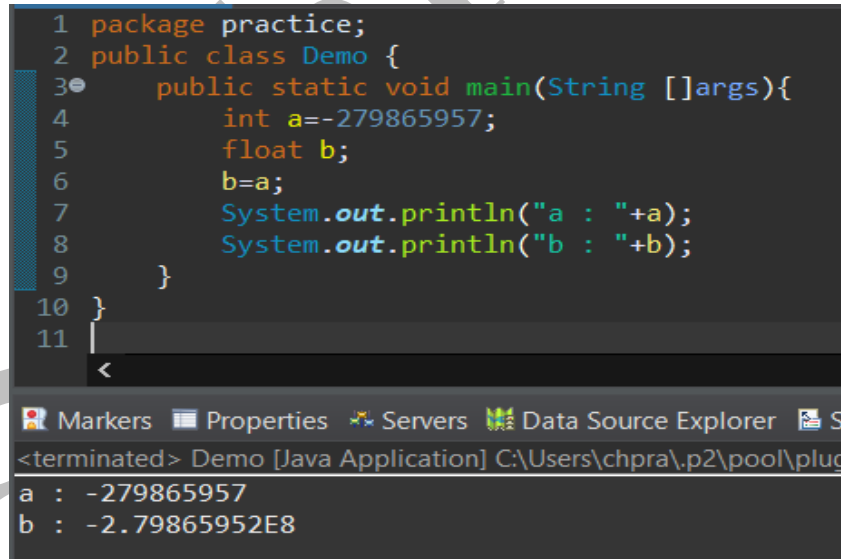
int-to-long conversion is done through implicit type casting.

## Conversion of data from int-to-float datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        int a=-279865957;  
        float b;  
        b=a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

A screenshot of an IDE window. The top pane shows the Java code from the previous block, with line numbers 1 through 11. The bottom pane shows the output of the program: 'a : -279865957' and 'b : -2.79865952E8'. The IDE interface includes tabs for 'Markers', 'Properties', 'Servers', and 'Data Source Explorer' at the top of the bottom pane. The title bar of the bottom pane reads '<terminated> Demo [Java Application] C:\Users\chpra\.p2\pool\plug...'.

```
1 package practice;  
2 public class Demo {  
3     public static void main(String []args){  
4         int a=-279865957;  
5         float b;  
6         b=a;  
7         System.out.println("a : "+a);  
8         System.out.println("b : "+b);  
9     }  
10 }  
11 |  
<  
  
Markers Properties Servers Data Source Explorer  
<terminated> Demo [Java Application] C:\Users\chpra\.p2\pool\plug...  
a : -279865957  
b : -2.79865952E8
```

### Conclusion:

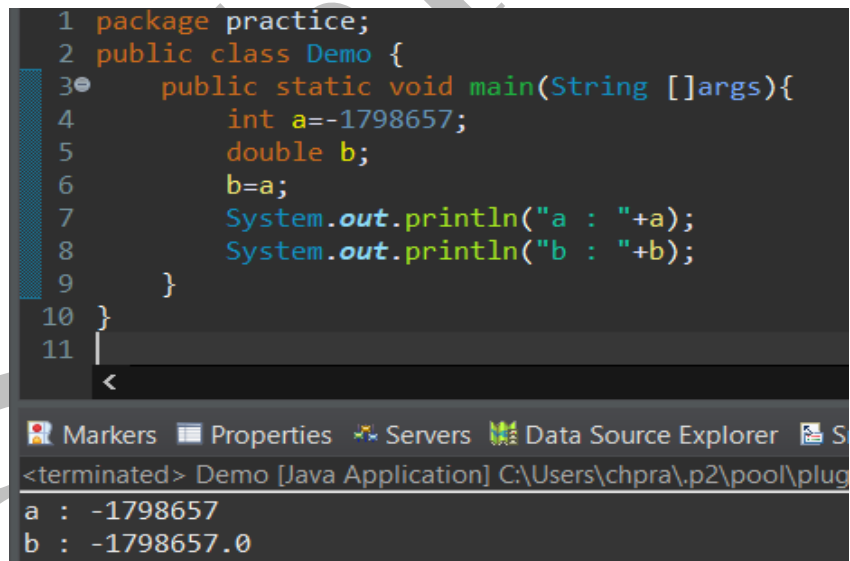
int-to-float conversion is done through implicit type casting.

## Conversion of data from int-to-double datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        int a=-1798657;  
        double b;  
        b=a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

A screenshot of an IDE window showing the Java code from the previous block. The code is displayed in a dark-themed editor with line numbers 1 through 11. Below the editor, the IDE's output console is visible, showing the execution results. The output shows the value of 'a' as -1798657 and the value of 'b' as -1798657.0, demonstrating the implicit conversion from int to double.

```
1 package practice;  
2 public class Demo {  
3     public static void main(String []args){  
4         int a=-1798657;  
5         double b;  
6         b=a;  
7         System.out.println("a : "+a);  
8         System.out.println("b : "+b);  
9     }  
10 }  
11  
<  
Markers Properties Servers Data Source Explorer  
<terminated> Demo [Java Application] C:\Users\chpra\p2\pool\plug  
a : -1798657  
b : -1798657.0
```

### Conclusion:

int-to-double conversion is done through implicit type casting.

## Conversion of data from int-to-boolean datatype:

### Program://Implicit type casting

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        int a=1798657;  
        boolean b;  
        b=a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### }//Explicit type casting

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        int a=1798657;  
        boolean b;  
        b=(boolean)a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

Compilation error for type casting.

### Conclusion:

int-to-boolean conversion is not possible.

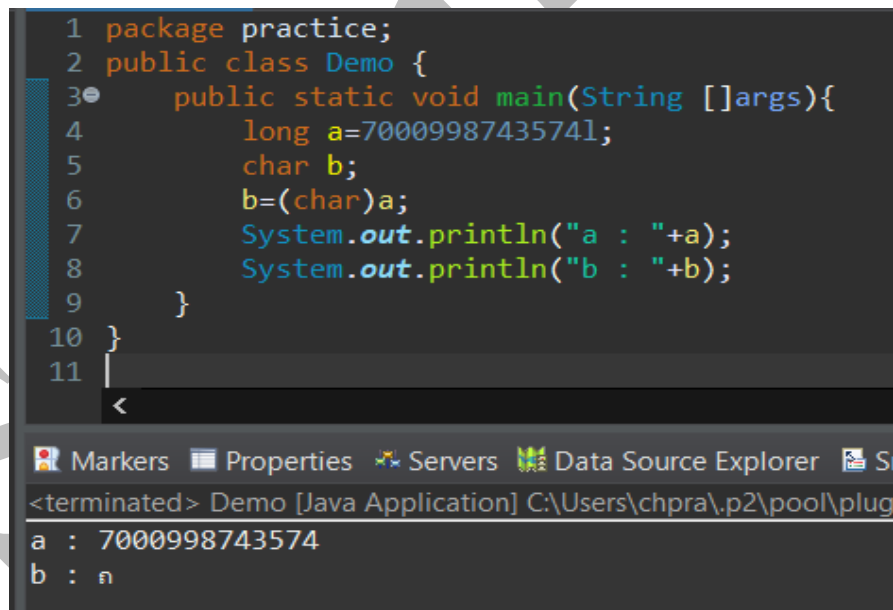


## Conversion of data from long-to-char datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        long a=7000998743574l;  
        char b;  
        b=(char)a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

A screenshot of a Java IDE window. The top pane shows the source code for a class named 'Demo' in the 'practice' package. The code defines a 'main' method that declares a 'long' variable 'a' with the value 7000998743574l, a 'char' variable 'b', and performs an explicit cast from 'a' to 'b' using '(char)a'. It then prints the values of 'a' and 'b'. The bottom pane shows the IDE's interface with tabs for 'Markers', 'Properties', 'Servers', 'Data Source Explorer', and 'S'. Below these tabs, a console window displays the output of the program: 'a : 7000998743574' and 'b :  ' (a null character).

```
1 package practice;  
2 public class Demo {  
3     public static void main(String []args){  
4         long a=7000998743574l;  
5         char b;  
6         b=(char)a;  
7         System.out.println("a : "+a);  
8         System.out.println("b : "+b);  
9     }  
10 }  
11 |  
<  
  
Markers Properties Servers Data Source Explorer S  
<terminated> Demo [Java Application] C:\Users\chpra\p2\pool\plug  
a : 7000998743574  
b :  
```

### Conclusion:

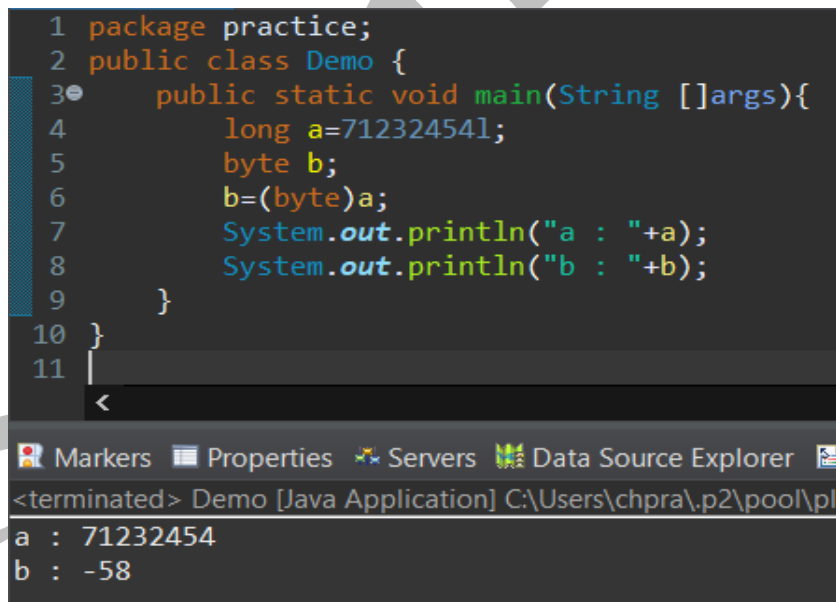
long-to-char conversion is done through explicit type casting.

## Conversion of data from long-to-byte datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        long a=71232454l;  
        byte b;  
        b=(byte)a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:



The screenshot shows a Java IDE with the following code in the editor:

```
1 package practice;  
2 public class Demo {  
3     public static void main(String []args){  
4         long a=71232454l;  
5         byte b;  
6         b=(byte)a;  
7         System.out.println("a : "+a);  
8         System.out.println("b : "+b);  
9     }  
10 }  
11
```

The output console at the bottom shows the following output:

```
<terminated> Demo [Java Application] C:\Users\chpra\p2\pool\pl  
a : 71232454  
b : -58
```

### Conclusion:

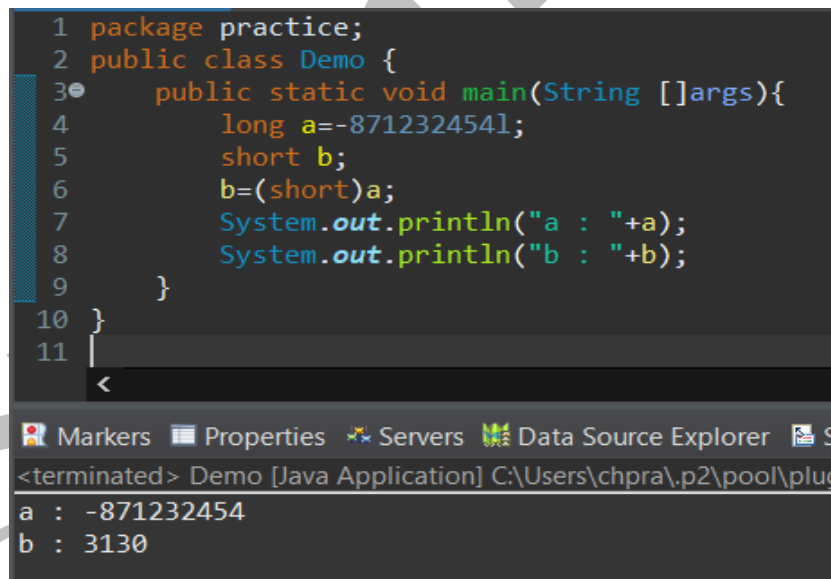
long-to-byte conversion is done through explicit type casting.

## Conversion of data from long-to-short datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        long a=-871232454l;  
        short b;  
        b=(short)a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

A screenshot of an IDE window. The top pane shows the Java code from the previous block, with line numbers 1 through 11. The bottom pane shows the output of the program: "a : -871232454" and "b : 3130". The IDE interface includes a toolbar with icons for Markers, Properties, Servers, and Data Source Explorer, and a status bar at the bottom indicating the application is terminated.

```
1 package practice;  
2 public class Demo {  
3     public static void main(String []args){  
4         long a=-871232454l;  
5         short b;  
6         b=(short)a;  
7         System.out.println("a : "+a);  
8         System.out.println("b : "+b);  
9     }  
10 }  
11
```

<terminated> Demo [Java Application] C:\Users\chpra\.p2\pool\plug  
a : -871232454  
b : 3130

### Conclusion:

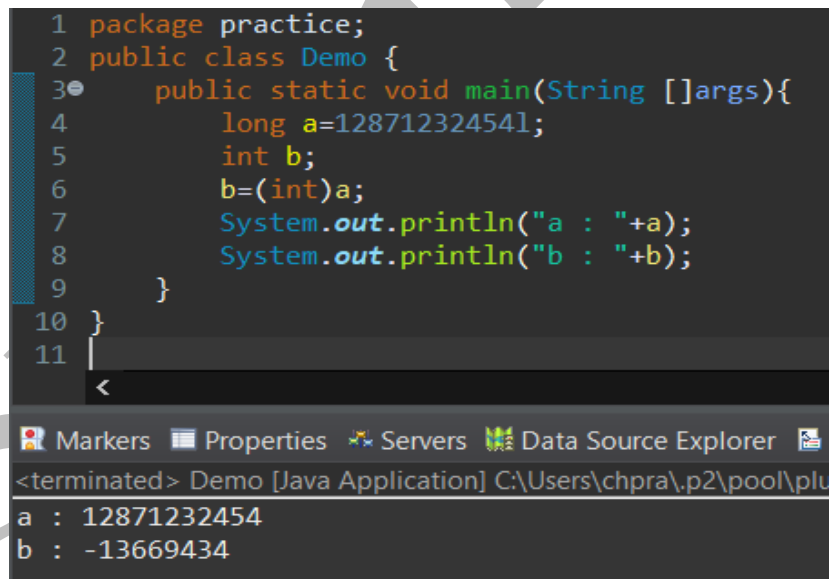
long-to-short conversion is done through explicit type casting.

## Conversion of data from long-to-int datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        long a=12871232454l;  
        int b;  
        b=(int)a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

A screenshot of an IDE window showing the Java code from the previous block. The code is displayed in a dark-themed editor with line numbers 1 through 11. Below the editor, the IDE's interface includes tabs for 'Markers', 'Properties', 'Servers', and 'Data Source Explorer'. The console output shows the program's execution results: 'a : 12871232454' and 'b : -13669434'. The output is displayed in a separate window titled '<terminated> Demo [Java Application] C:\Users\chpra\.p2\pool\plu'.

```
1 package practice;  
2 public class Demo {  
3     public static void main(String []args){  
4         long a=12871232454l;  
5         int b;  
6         b=(int)a;  
7         System.out.println("a : "+a);  
8         System.out.println("b : "+b);  
9     }  
10 }  
11
```

<terminated> Demo [Java Application] C:\Users\chpra\.p2\pool\plu  
a : 12871232454  
b : -13669434

### Conclusion:

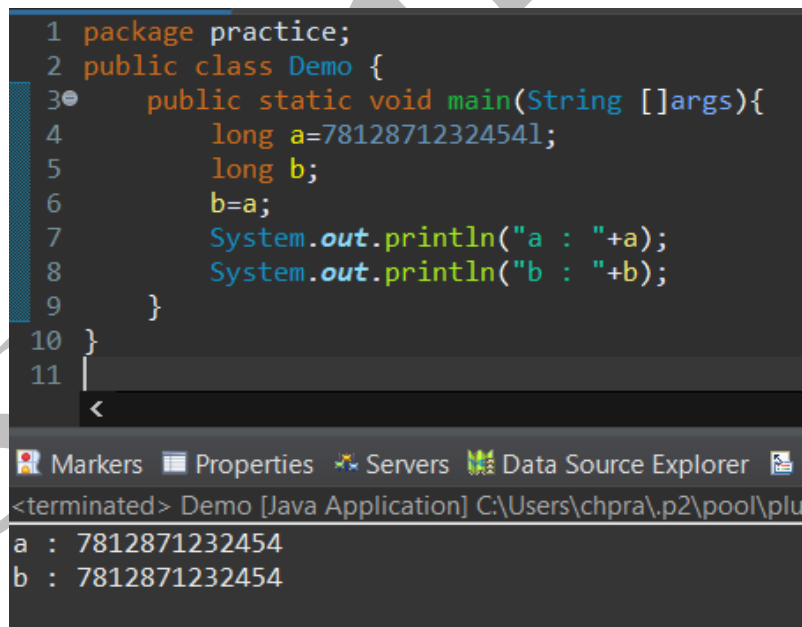
long-to-int conversion is done through explicit type casting.

## Conversion of data from long-to-long datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        long a=7812871232454l;  
        long b;  
        b=a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

A screenshot of a Java IDE. The top part shows the source code for a class named 'Demo' in the 'practice' package. The code defines a 'main' method where a long variable 'a' is assigned the value 7812871232454l, then 'a' is assigned to 'b', and both are printed. The bottom part of the screenshot shows the IDE's console window with the output: 'a : 7812871232454' and 'b : 7812871232454'. The IDE interface includes tabs for 'Markers', 'Properties', 'Servers', and 'Data Source Explorer'.

### Conclusion:

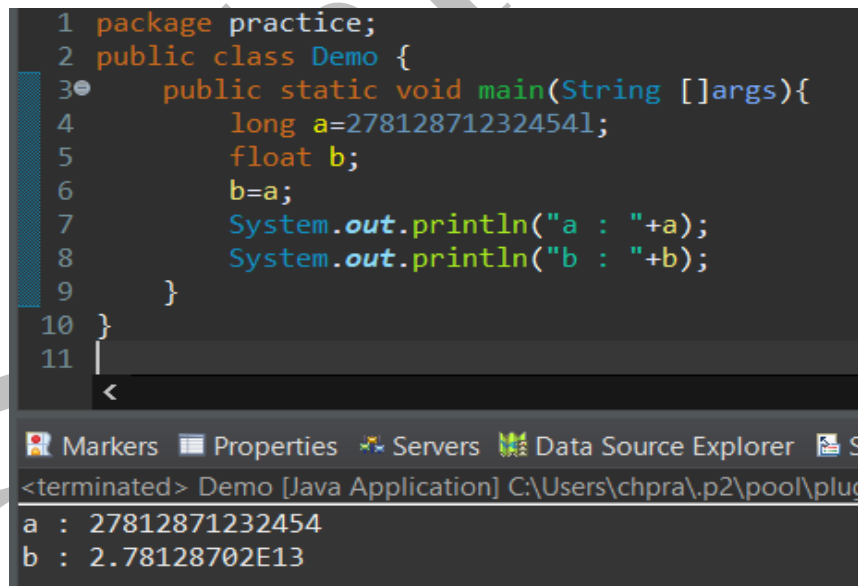
long-to-long conversion is not required.

## Conversion of data from long-to-float datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        long a=27812871232454l;  
        float b;  
        b=a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

A screenshot of a Java IDE window. The top pane shows the source code for a class named 'Demo' in the 'practice' package. The code defines a 'main' method that declares a 'long' variable 'a' with the value 27812871232454l, a 'float' variable 'b', and assigns 'b' the value of 'a'. It then prints the values of 'a' and 'b'. The bottom pane shows the IDE's interface with tabs for 'Markers', 'Properties', 'Servers', 'Data Source Explorer', and 'S'. Below these tabs, a console window displays the output of the program: 'a : 27812871232454' and 'b : 2.78128702E13'. The title bar of the IDE window indicates the application is 'Demo [Java Application]' located at 'C:\Users\chpra\.p2\pool\plug'.

### Conclusion:

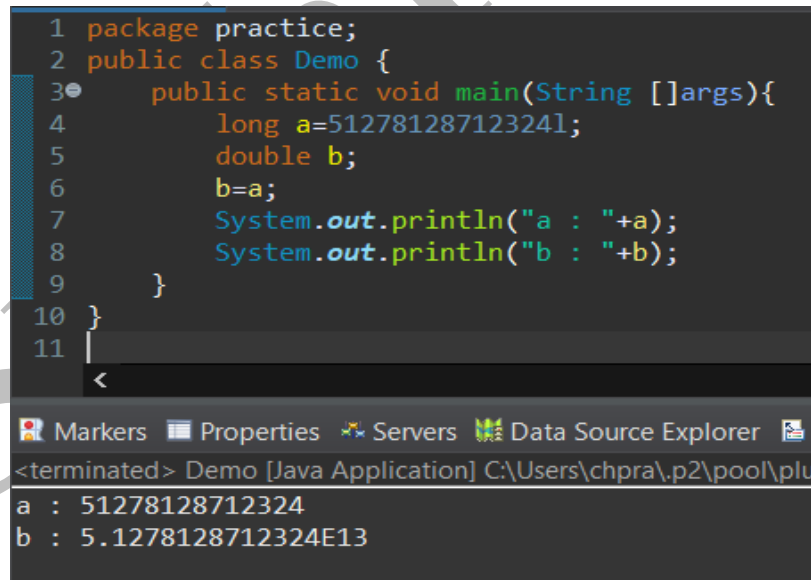
long-to-float conversion is done through implicit type casting.

## Conversion of data from long-to-double datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        long a=51278128712324l;  
        double b;  
        b=a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

A screenshot of an IDE window showing the Java code from the previous block. The code is highlighted with syntax coloring. Below the code editor, the IDE's console window is visible, showing the output of the program. The output consists of two lines: "a : 51278128712324" and "b : 5.1278128712324E13". The IDE interface includes a toolbar with icons for Markers, Properties, Servers, and Data Source Explorer. The console window title is "<terminated> Demo [Java Application] C:\Users\chpra\.p2\pool\plug".

```
1 package practice;  
2 public class Demo {  
3     public static void main(String []args){  
4         long a=51278128712324l;  
5         double b;  
6         b=a;  
7         System.out.println("a : "+a);  
8         System.out.println("b : "+b);  
9     }  
10 }  
11
```

Markers Properties Servers Data Source Explorer

<terminated> Demo [Java Application] C:\Users\chpra\.p2\pool\plug

a : 51278128712324  
b : 5.1278128712324E13

### Conclusion:

long-to-double conversion is done through implicit type casting.

## Conversion of data from long-to-boolean datatype:

### Program://Implicit type casting

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        long a=7512781287123241;  
        boolean b;  
        b=a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### }//Explicit type casting

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        long a=7512781287123241;  
        boolean b;  
        b=(boolean)a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

Compilation error for type casting.

### Conclusion:

int-to-boolean conversion is not possible.

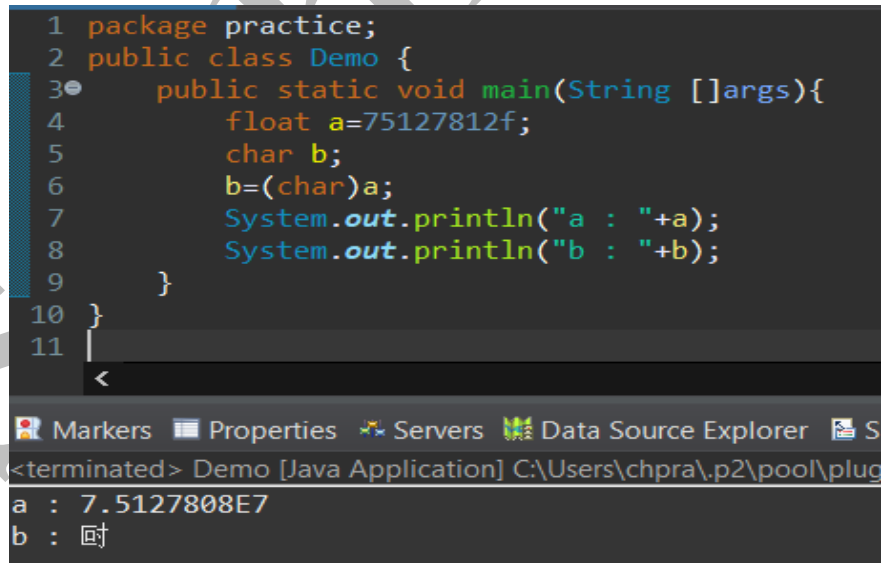


## Conversion of data from float-to-char datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        float a=75127812f;  
        char b;  
        b=(char)a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

A screenshot of a Java IDE window. The top pane shows the source code for a class named 'Demo' in the 'practice' package. The code defines a 'main' method that declares a float variable 'a' with the value 75127812f, a char variable 'b', and performs an explicit cast from float to char. It then prints the values of 'a' and 'b'. The bottom pane shows the IDE's interface with tabs for 'Markers', 'Properties', 'Servers', 'Data Source Explorer', and 'S'. Below these, a status bar indicates the application is terminated. The output console at the bottom displays the results: 'a : 7.5127808E7' and 'b : ' (an empty character).

```
1 package practice;  
2 public class Demo {  
3     public static void main(String []args){  
4         float a=75127812f;  
5         char b;  
6         b=(char)a;  
7         System.out.println("a : "+a);  
8         System.out.println("b : "+b);  
9     }  
10 }  
11 |  
<  
  
Markers Properties Servers Data Source Explorer S  
<terminated> Demo [Java Application] C:\Users\chpra.p2\pool\plug  
a : 7.5127808E7  
b : 
```

### Conclusion:

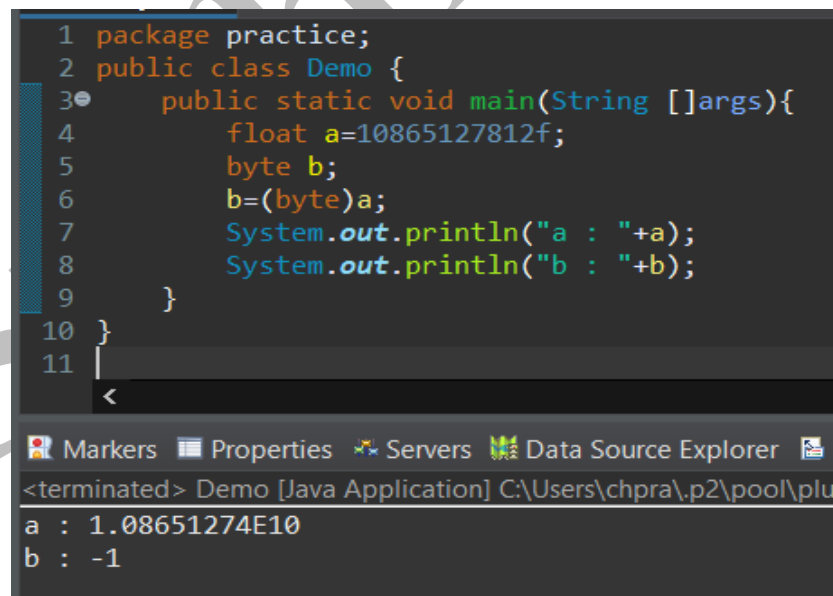
float-to-char conversion is done through explicit type casting.

## Conversion of data from float-to-byte datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        float a=10865127812f;  
        byte b;  
        b=(byte)a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:



```
1 package practice;  
2 public class Demo {  
3     public static void main(String []args){  
4         float a=10865127812f;  
5         byte b;  
6         b=(byte)a;  
7         System.out.println("a : "+a);  
8         System.out.println("b : "+b);  
9     }  
10 }  
11
```

<terminated> Demo [Java Application] C:\Users\chpra\p2\pool\plu  
a : 1.08651274E10  
b : -1

### Conclusion:

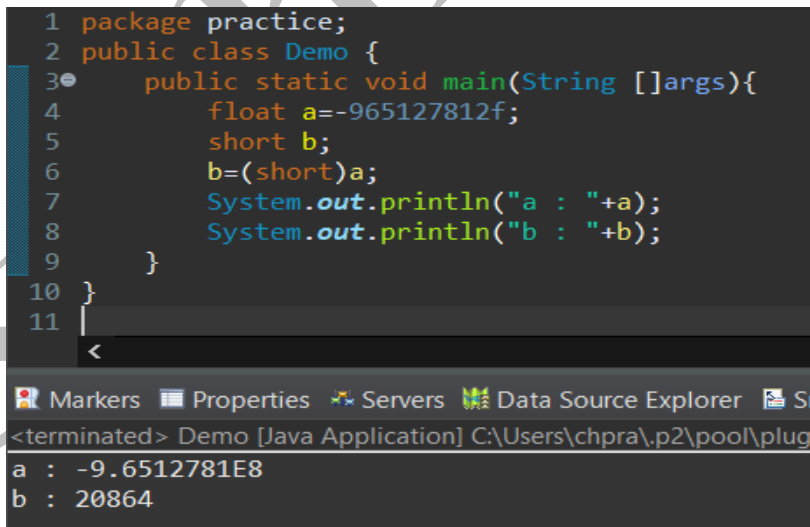
float-to-byte conversion is done through explicit type casting.

## Conversion of data from float-to-short datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        float a=-965127812f;  
        short b;  
        b=(short)a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:



The screenshot shows a Java IDE with the following code in the editor:

```
1 package practice;  
2 public class Demo {  
3     public static void main(String []args){  
4         float a=-965127812f;  
5         short b;  
6         b=(short)a;  
7         System.out.println("a : "+a);  
8         System.out.println("b : "+b);  
9     }  
10 }  
11
```

The output window at the bottom shows the following output:

```
<terminated> Demo [Java Application] C:\Users\chpra\.p2\pool\plug  
a : -9.6512781E8  
b : 20864
```

### Conclusion:

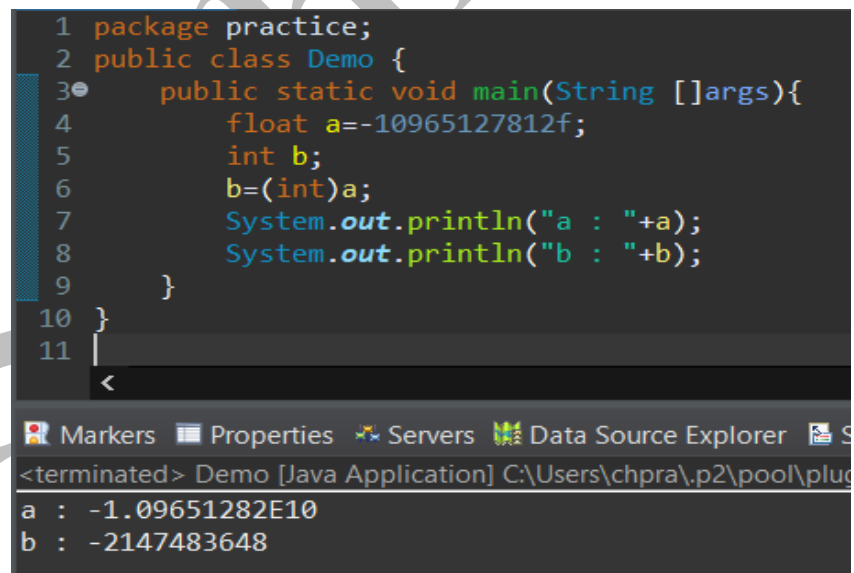
float-to-short conversion is done through explicit type casting.

## Conversion of data from float-to-int datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        float a=-10965127812f;  
        int b;  
        b=(int)a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:



The screenshot shows a Java IDE with the following code in the editor:

```
1 package practice;  
2 public class Demo {  
3     public static void main(String []args){  
4         float a=-10965127812f;  
5         int b;  
6         b=(int)a;  
7         System.out.println("a : "+a);  
8         System.out.println("b : "+b);  
9     }  
10 }  
11 |
```

The output window at the bottom shows the following results:

```
<terminated> Demo [Java Application] C:\Users\chpra\p2\pool\plug  
a : -1.09651282E10  
b : -2147483648
```

### Conclusion:

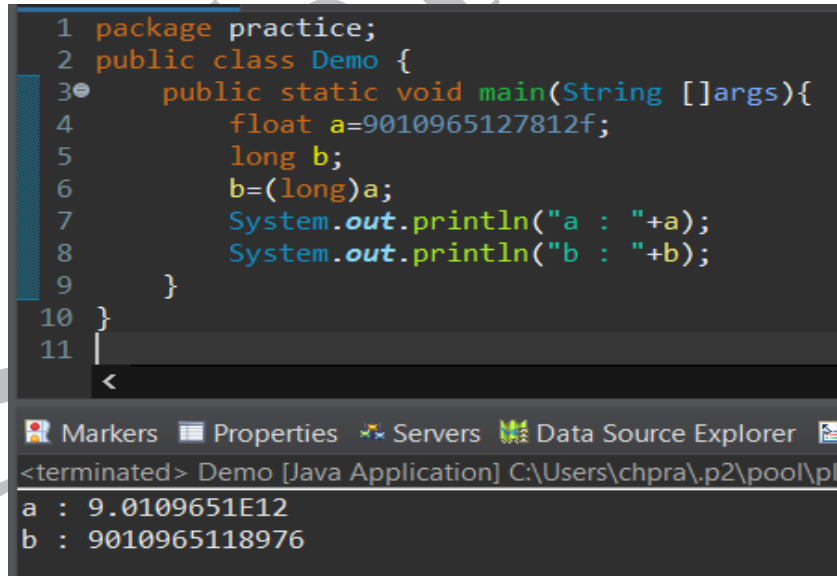
float-to-int conversion is done through explicit type casting.

## Conversion of data from float-to-long datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        float a=9010965127812f;  
        long b;  
        b=(long)a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:



The screenshot shows an IDE window with the following code in the editor:

```
1 package practice;  
2 public class Demo {  
3     public static void main(String []args){  
4         float a=9010965127812f;  
5         long b;  
6         b=(long)a;  
7         System.out.println("a : "+a);  
8         System.out.println("b : "+b);  
9     }  
10 }  
11
```

Below the editor, the console output is displayed:

```
<terminated> Demo [Java Application] C:\Users\chpra\.p2\pool\pl  
a : 9.0109651E12  
b : 9010965118976
```

### Conclusion:

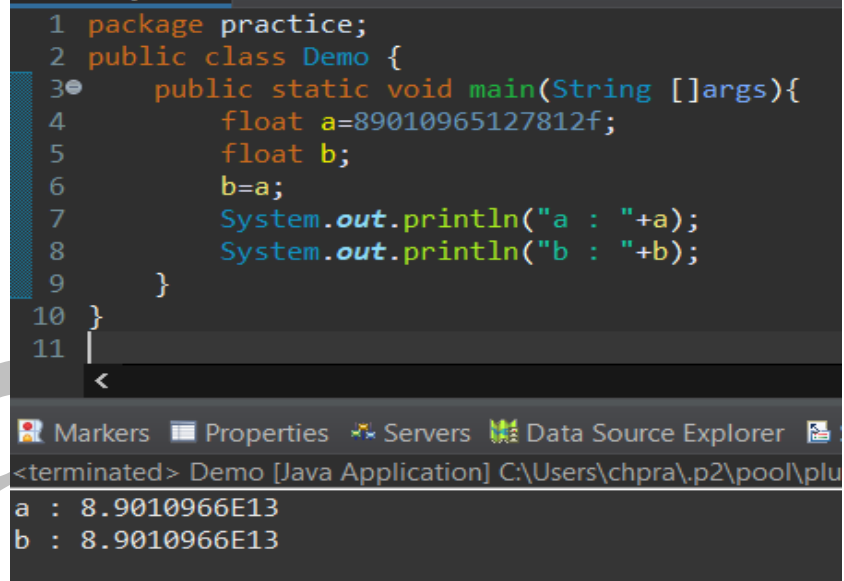
float-to-long conversion is done through explicit type casting.

## Conversion of data from float-to-float datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        float a=89010965127812f;  
        float b;  
        b=a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:



The screenshot shows a Java IDE with the following code in the editor:

```
1 package practice;  
2 public class Demo {  
3     public static void main(String []args){  
4         float a=89010965127812f;  
5         float b;  
6         b=a;  
7         System.out.println("a : "+a);  
8         System.out.println("b : "+b);  
9     }  
10 }  
11
```

The IDE's console window at the bottom shows the output of the program:

```
<terminated> Demo [Java Application] C:\Users\chpra\.p2\pool\plu  
a : 8.9010966E13  
b : 8.9010966E13
```

### Conclusion:

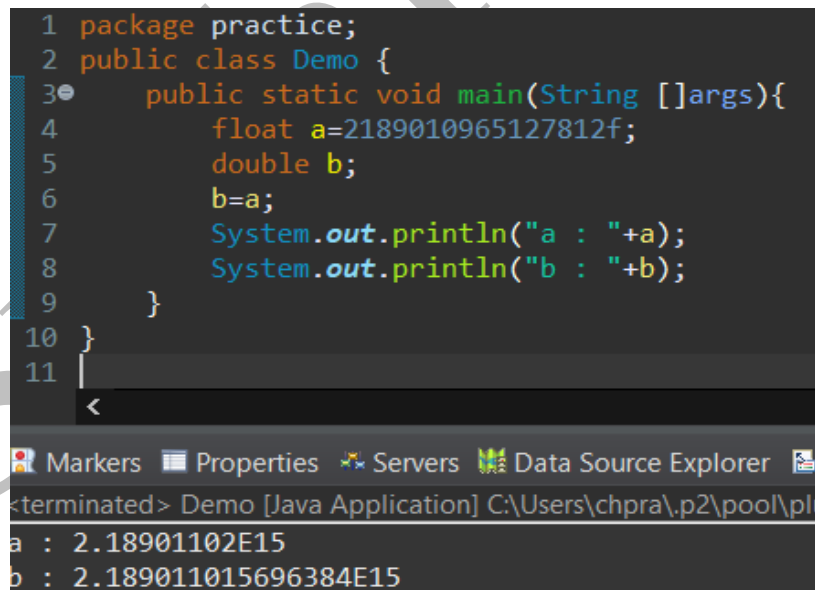
float-to-float conversion is not required.

## Conversion of data from float-to-double datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        float a=2189010965127812f;  
        double b;  
        b=a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:



The screenshot shows a Java IDE with the following code in the editor:

```
1 package practice;  
2 public class Demo {  
3     public static void main(String []args){  
4         float a=2189010965127812f;  
5         double b;  
6         b=a;  
7         System.out.println("a : "+a);  
8         System.out.println("b : "+b);  
9     }  
10 }  
11
```

Below the code editor, the IDE's console window displays the output of the program:

```
<terminated> Demo [Java Application] C:\Users\chpra\.p2\pool\pl  
a : 2.18901102E15  
b : 2.189011015696384E15
```

### Conclusion:

float-to-double conversion is done through implicit type casting.

## Conversion of data from float-to-boolean datatype:

### Program://Implicit type casting

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        float a=10965127812f;  
        boolean b;  
        b=a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### }//Explicit type casting

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        float a=10965127812f;  
        boolean b;  
        b=(boolean)a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

Compilation error for type casting.

### Conclusion:

float-to-boolean conversion is not possible.

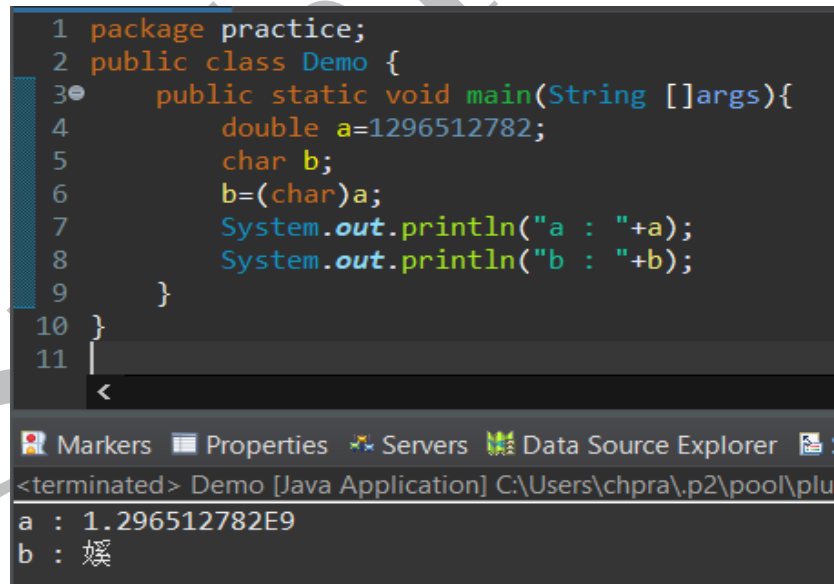


## Conversion of data from double-to-char datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        double a=1296512782;  
        char b;  
        b=(char)a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

A screenshot of a Java IDE window. The top pane shows the source code for a class named 'Demo' in the 'practice' package. The code defines a 'main' method that declares a 'double' variable 'a' with the value 1296512782, a 'char' variable 'b', and casts 'a' to 'b' using '(char)a'. It then prints the values of 'a' and 'b'. The bottom pane shows the IDE's interface with tabs for 'Markers', 'Properties', 'Servers', and 'Data Source Explorer'. Below these is a console window titled '<terminated> Demo [Java Application] C:\Users\chpra\.p2\pool\plu' which displays the output: 'a : 1.296512782E9' and 'b : 爰'.

### Conclusion:

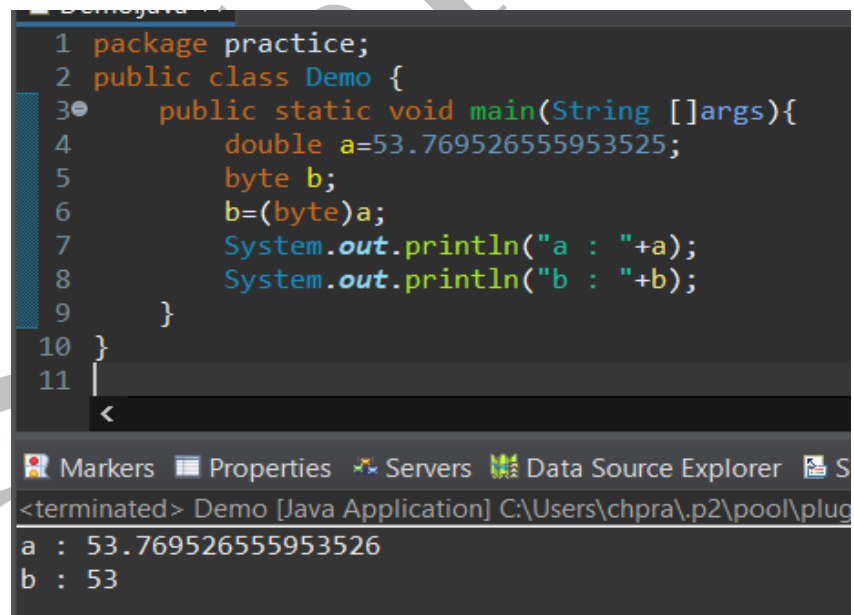
double-to-char conversion is done through explicit type casting.

## Conversion of data from double-to-byte datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        double a=53.769526555953525;  
        byte b;  
        b=(byte)a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

A screenshot of a Java IDE window titled 'Demo.java'. The code editor shows the same code as in the 'Program' block. Below the code editor, the 'Markers', 'Properties', 'Servers', 'Data Source Explorer', and 'S' tabs are visible. The console output shows the execution results: 'a : 53.769526555953526' and 'b : 53'.

```
1 package practice;  
2 public class Demo {  
3     public static void main(String []args){  
4         double a=53.769526555953525;  
5         byte b;  
6         b=(byte)a;  
7         System.out.println("a : "+a);  
8         System.out.println("b : "+b);  
9     }  
10 }  
11 |  
<  
  
Markers Properties Servers Data Source Explorer S  
<terminated> Demo [Java Application] C:\Users\chpra\p2\pool\plug  
a : 53.769526555953526  
b : 53
```

### Conclusion:

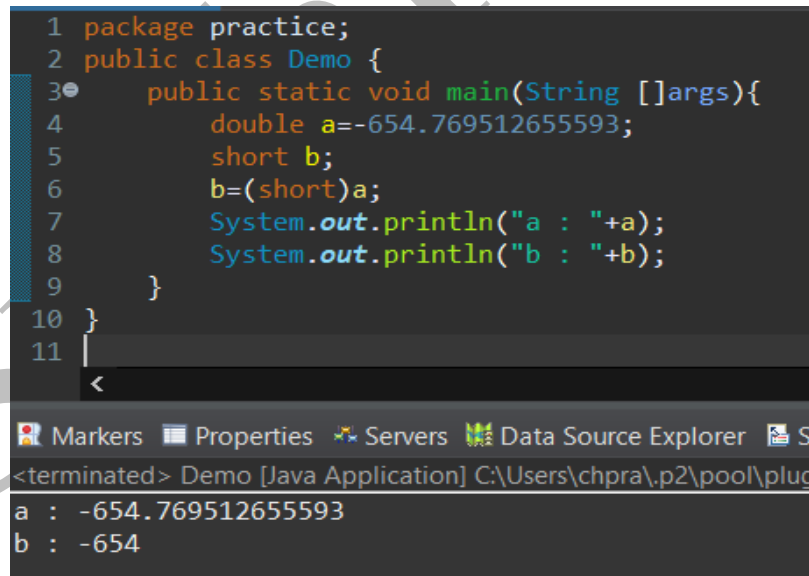
double-to-byte conversion is done through explicit type casting.

## Conversion of data from double-to-short datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        double a=-654.769512655593;  
        short b;  
        b=(short)a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

A screenshot of an IDE window. The top part shows the Java code from the previous block. The bottom part shows the output of the program in a console window. The output is:  
<terminated> Demo [Java Application] C:\Users\chpra\.p2\pool\plug  
a : -654.769512655593  
b : -654  

```
1 package practice;  
2 public class Demo {  
3     public static void main(String []args){  
4         double a=-654.769512655593;  
5         short b;  
6         b=(short)a;  
7         System.out.println("a : "+a);  
8         System.out.println("b : "+b);  
9     }  
10 }  
11 |  
<
```

Markers Properties Servers Data Source Explorer

<terminated> Demo [Java Application] C:\Users\chpra\.p2\pool\plug

a : -654.769512655593  
b : -654

### Conclusion:

double-to-short conversion is done through explicit type casting.

## Conversion of data from double-to-int datatype:

### Program:

```
package practice;

public class Demo {

    public static void main(String []args){

        double a=1769.088765547;

        int b;

        b=(int)a;

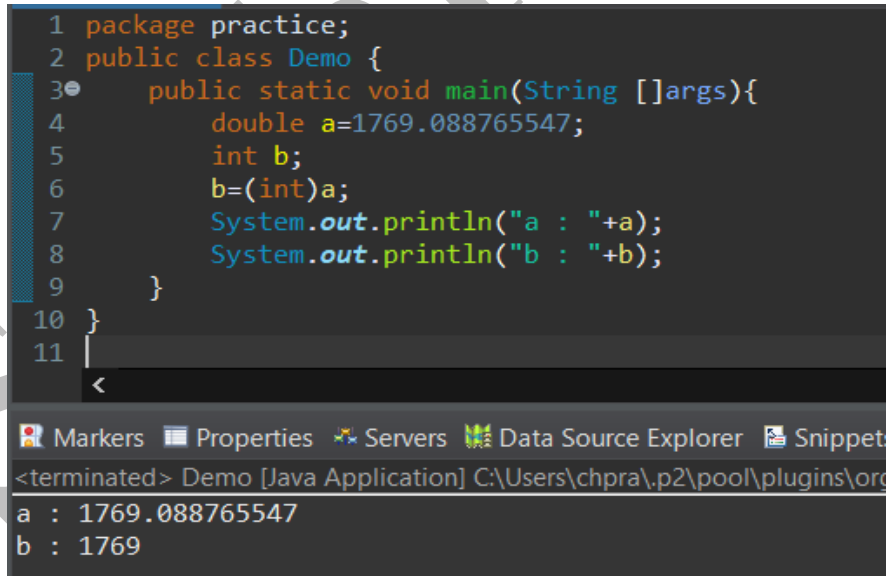
        System.out.println("a : "+a);

        System.out.println("b : "+b);

    }

}
```

### Output:

A screenshot of an IDE window showing the Java code from the previous block. The code is highlighted with syntax coloring. Below the code editor, the output console is visible, showing the execution results. The output shows the value of 'a' as 1769.088765547 and the value of 'b' as 1769, demonstrating the truncation of the decimal part during the explicit cast to an integer.

```
1 package practice;
2 public class Demo {
3     public static void main(String []args){
4         double a=1769.088765547;
5         int b;
6         b=(int)a;
7         System.out.println("a : "+a);
8         System.out.println("b : "+b);
9     }
10 }
11
```

Markers Properties Servers Data Source Explorer Snippet

<terminated> Demo [Java Application] C:\Users\chpra\.p2\pool\plugins\org

a : 1769.088765547  
b : 1769

### Conclusion:

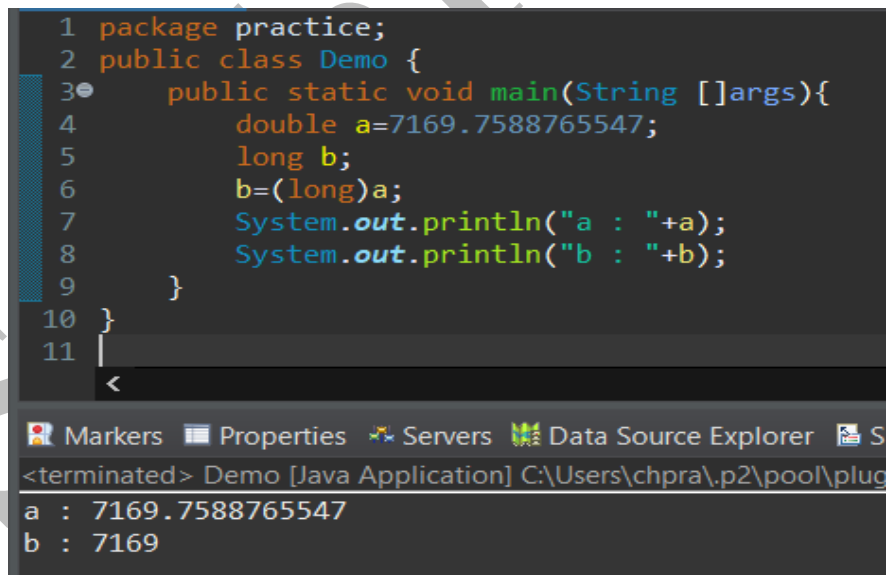
double-to-int conversion is done through explicit type casting.

## Conversion of data from double-to-long datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        double a=7169.7588765547;  
        long b;  
        b=(long)a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

A screenshot of an IDE window showing the Java code from the previous block. The code is in a dark-themed editor with line numbers 1 through 11. Below the editor, there is a console window showing the output of the program. The output consists of two lines: "a : 7169.7588765547" and "b : 7169". The IDE interface includes tabs for "Markers", "Properties", "Servers", "Data Source Explorer", and "S". The console window title is "<terminated> Demo [Java Application] C:\Users\chpra\.p2\pool\plug".

```
1 package practice;  
2 public class Demo {  
3     public static void main(String []args){  
4         double a=7169.7588765547;  
5         long b;  
6         b=(long)a;  
7         System.out.println("a : "+a);  
8         System.out.println("b : "+b);  
9     }  
10 }  
11 |  
<  
  
Markers Properties Servers Data Source Explorer S  
<terminated> Demo [Java Application] C:\Users\chpra\.p2\pool\plug  
a : 7169.7588765547  
b : 7169
```

### Conclusion:

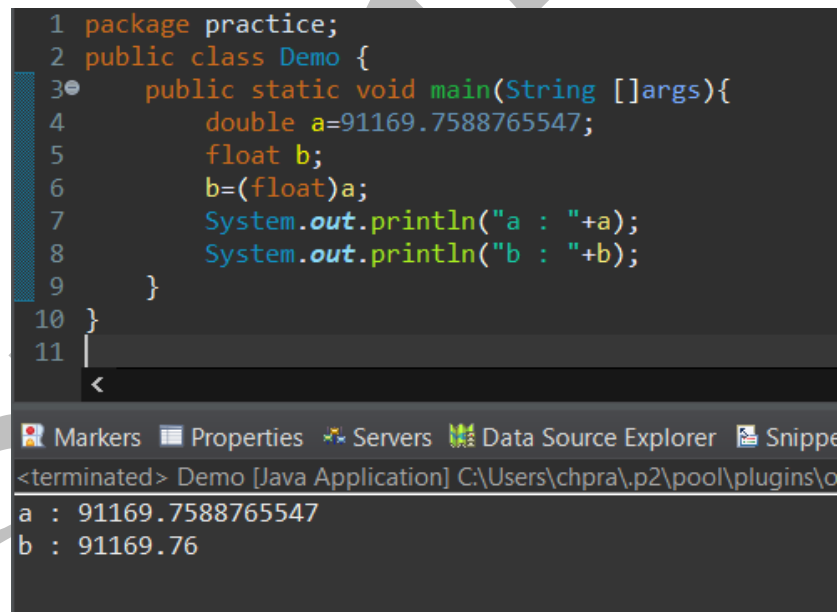
double-to-long conversion is done through explicit type casting.

## Conversion of data from double-to-float datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        double a=91169.7588765547;  
        float b;  
        b=(float)a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

A screenshot of an IDE window. The top pane shows the Java code from the previous block, with line numbers 1 through 11. The bottom pane shows the output of the program: "a : 91169.7588765547" and "b : 91169.76". The IDE interface includes tabs for Markers, Properties, Servers, Data Source Explorer, and Snippets.

```
1 package practice;  
2 public class Demo {  
3     public static void main(String []args){  
4         double a=91169.7588765547;  
5         float b;  
6         b=(float)a;  
7         System.out.println("a : "+a);  
8         System.out.println("b : "+b);  
9     }  
10 }  
11 |  
<  
  
Markers Properties Servers Data Source Explorer Snippets  
<terminated> Demo [Java Application] C:\Users\chpra\.p2\pool\plugins\c  
a : 91169.7588765547  
b : 91169.76
```

### Conclusion:

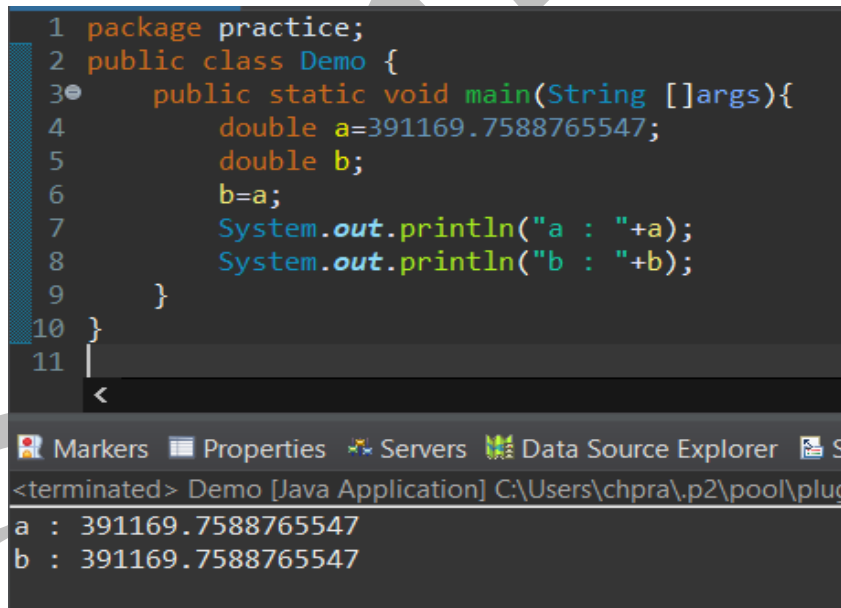
double-to-float conversion is done through explicit type casting.

## Conversion of data from double-to-double datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        double a=391169.7588765547;  
        double b;  
        b=a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

A screenshot of a Java IDE window. The top pane shows the source code for a class named 'Demo' in the 'practice' package. The code defines a 'main' method that declares a 'double' variable 'a' with the value 391169.7588765547, declares another 'double' variable 'b', assigns 'a' to 'b', and prints both values. The bottom pane shows the output of the program, which is the same value for both 'a' and 'b'. The IDE interface includes a toolbar with icons for Markers, Properties, Servers, Data Source Explorer, and a console window at the bottom.

```
1 package practice;  
2 public class Demo {  
3     public static void main(String []args){  
4         double a=391169.7588765547;  
5         double b;  
6         b=a;  
7         System.out.println("a : "+a);  
8         System.out.println("b : "+b);  
9     }  
10 }  
11
```

<terminated> Demo [Java Application] C:\Users\chpra\.p2\pool\plug  
a : 391169.7588765547  
b : 391169.7588765547

### Conclusion:

double-to-double conversion is not required.

## Conversion of data from double-to-boolean datatype:

### Program://Implicit type casting

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        double a=1096512782;  
        boolean b;  
        b=a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### }//Explicit type casting

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        double a=1096512782;  
        boolean b;  
        b=(boolean)a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

Compilation error for type casting.

### Conclusion:

double-to-boolean conversion is not possible.



## Conversion of data from boolean-to-char datatype:

### Program:

```
package practice;  
public class Demo {  
    public static void main(String []args){  
        boolean a=true;  
        char b;  
        b=a;  
        // explicit type casting  
        b=(char)a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

Compilation error for type casting.

### Conclusion:

boolean-to-char conversion is not possible.

## Conversion of data from boolean-to-byte datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        boolean a=true;  
        byte b;  
        b=a;  
        // explicit type casting  
        b=(byte)a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

Compilation error for type casting.

### Conclusion:

boolean-to-byte conversion is not possible.

## Conversion of data from boolean-to-short datatype:

### Program:

```
package practice;  
public class Demo {  
    public static void main(String []args){  
        boolean a=true;  
        short b;  
        b=a;  
        // explicit type casting  
        b=(short)a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

Compilation error for type casting.

### Conclusion:

boolean-to-short conversion is not possible.

## Conversion of data from boolean-to-int datatype:

### Program:

```
package practice;  
public class Demo {  
    public static void main(String []args){  
        boolean a=true;  
        int b;  
        b=a;  
        // explicit type casting  
        b=(int)a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

Compilation error for type casting.

### Conclusion:

boolean-to-int conversion is not possible.

## Conversion of data from boolean-to-long datatype:

### Program:

```
package practice;  
public class Demo {  
    public static void main(String []args){  
        boolean a=true;  
        long b;  
        b=a;  
        // explicit type casting  
        b=(long)a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

Compilation error for type casting.

### Conclusion:

boolean-to-long conversion is not possible.

## Conversion of data from boolean-to-float datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        boolean a=true;  
        float b;  
        b=a;  
        // explicit type casting  
        b=(float)a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

Compilation error for type casting.

### Conclusion:

boolean-to-float conversion is not possible.

## Conversion of data from boolean-to-double datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        boolean a=true;  
        double b;  
        b=a;  
        // explicit type casting  
        b=(double)a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

Compilation error for type casting.

### Conclusion:

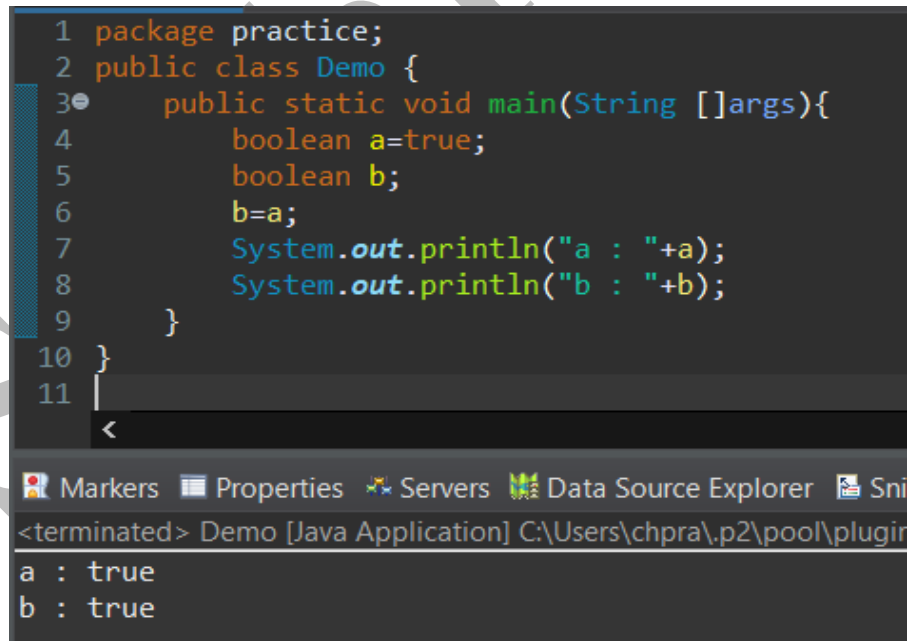
boolean-to-double conversion is not possible.

## Conversion of data from boolean-to-boolean datatype:

### Program:

```
package practice;  
  
public class Demo {  
    public static void main(String []args){  
        boolean a=true;  
        boolean b;  
        b=a;  
        System.out.println("a : "+a);  
        System.out.println("b : "+b);  
    }  
}
```

### Output:

A screenshot of an IDE window. The top pane shows the Java code from the previous block, with line numbers 1 through 11. The bottom pane shows the output of the program, which is "a : true" followed by "b : true" on the next line. The IDE's interface includes tabs for Markers, Properties, Servers, Data Source Explorer, and Snippets. The title bar of the bottom pane reads "<terminated> Demo [Java Application] C:\Users\chpra\p2\pool\plugin".

```
1 package practice;  
2 public class Demo {  
3     public static void main(String []args){  
4         boolean a=true;  
5         boolean b;  
6         b=a;  
7         System.out.println("a : "+a);  
8         System.out.println("b : "+b);  
9     }  
10 }  
11 |  
<  
  
Markers Properties Servers Data Source Explorer Snippets  
<terminated> Demo [Java Application] C:\Users\chpra\p2\pool\plugin  
a : true  
b : true
```

### Conclusion:

boolean-to-boolean conversion is not required.



char→byte→short→int→long→float→double→boolean

----- implicit conversion

boolean→double→float→long→int→short→byte→char

-----explicit conversion

	char	byte	short	Int	long	Float	double	boolean
char	CNR	✓ Explicit	✓ Explicit	✓ Implicit	✓ Implicit	✓ Implicit	✓ Implicit	✗ Not possible
byte	✓ Explicit	CNR	✓ Implicit	✓ Implicit	✓ Implicit	✓ Implicit	✓ Implicit	✗ Not possible
short	✓ Explicit	✓ Explicit	CNR	✓ Implicit	✓ Implicit	✓ Implicit	✓ Implicit	✗ Not possible
int	✓ Explicit	✓ Explicit	✓ Explicit	CNR	✓ Implicit	✓ Implicit	✓ Implicit	✗ Not possible
long	✓ Explicit	✓ Explicit	✓ Explicit	✓ Explicit	CNR	✓ Implicit	✓ Implicit	✗ Not possible
float	✓ Explicit	✓ Explicit	✓ Explicit	✓ Explicit	✓ Explicit	CNR	✓ Implicit	✗ Not possible
double	✓ Explicit	✓ Explicit	✓ Explicit	✓ Explicit	✓ Explicit	✓ Explicit	CNR	✗ Not possible
boolean	✗ Not possible	✗ Not possible	✗ Not possible	✗ Not possible	✗ Not possible	✗ Not possible	✗ Not possible	CNR

CNR: Conversion not required