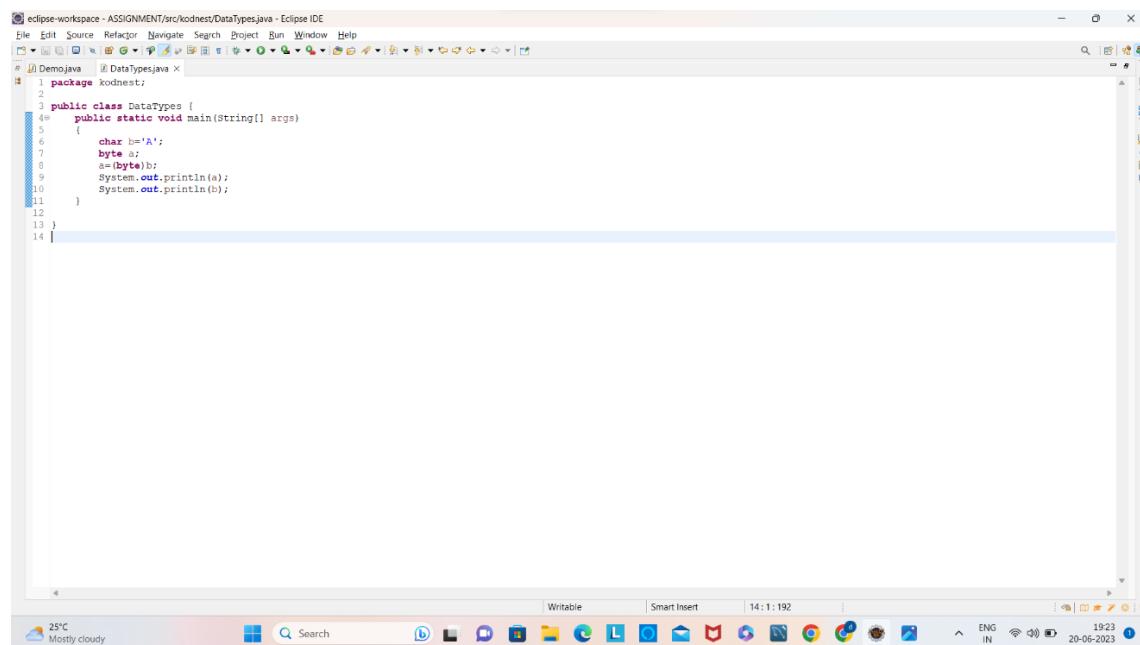


ASSIGNMENT

CONVERSION OF DATA TYPES

char to byte:

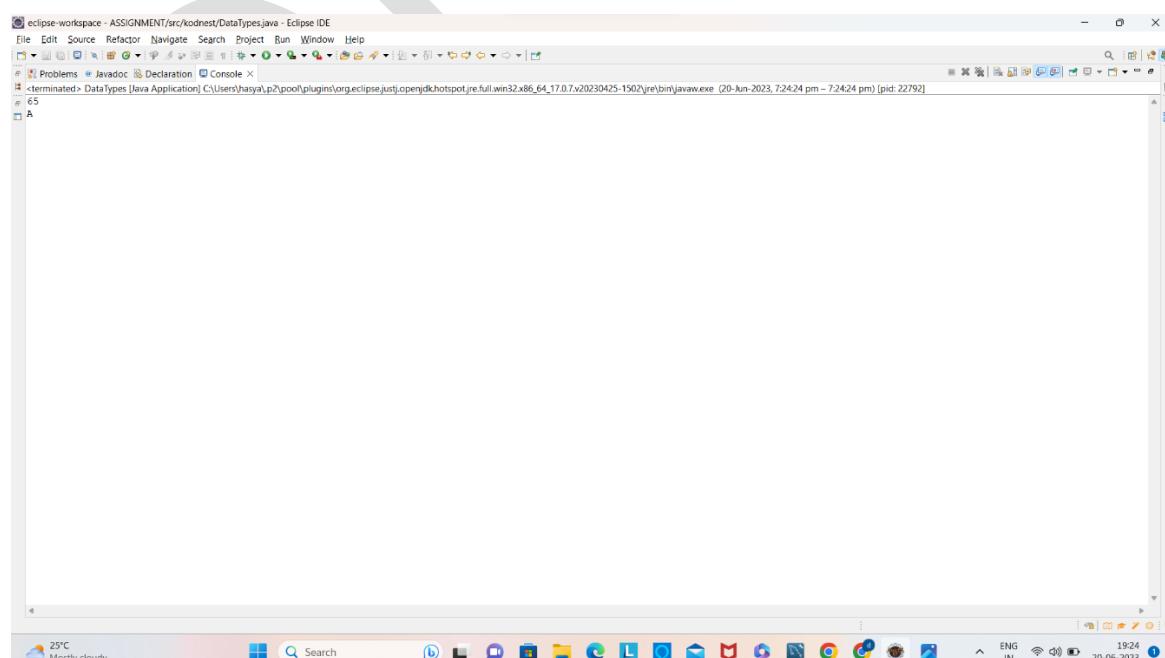
Program:



```
eclipse-workspace - ASSIGNMENT/src/kodnest/DataTypes.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Demojava DataTypes.java
1 package kodnest;
2
3 public class DataTypes {
4     public static void main(String[] args)
5     {
6         char b='A';
7         byte a;
8         a=(byte)b;
9         System.out.println(a);
10        System.out.println(b);
11    }
12
13 }
14
```

The screenshot shows the Eclipse IDE interface with a Java file named 'DataTypes.java' open. The code demonstrates implicit conversion from a character to a byte. It defines a class 'DataTypes' with a main method. Inside the main method, a character variable 'b' is assigned the value 'A'. A byte variable 'a' is then assigned the value of 'b' using an assignment operator with a cast. Finally, both variables are printed using System.out.println.

Output:



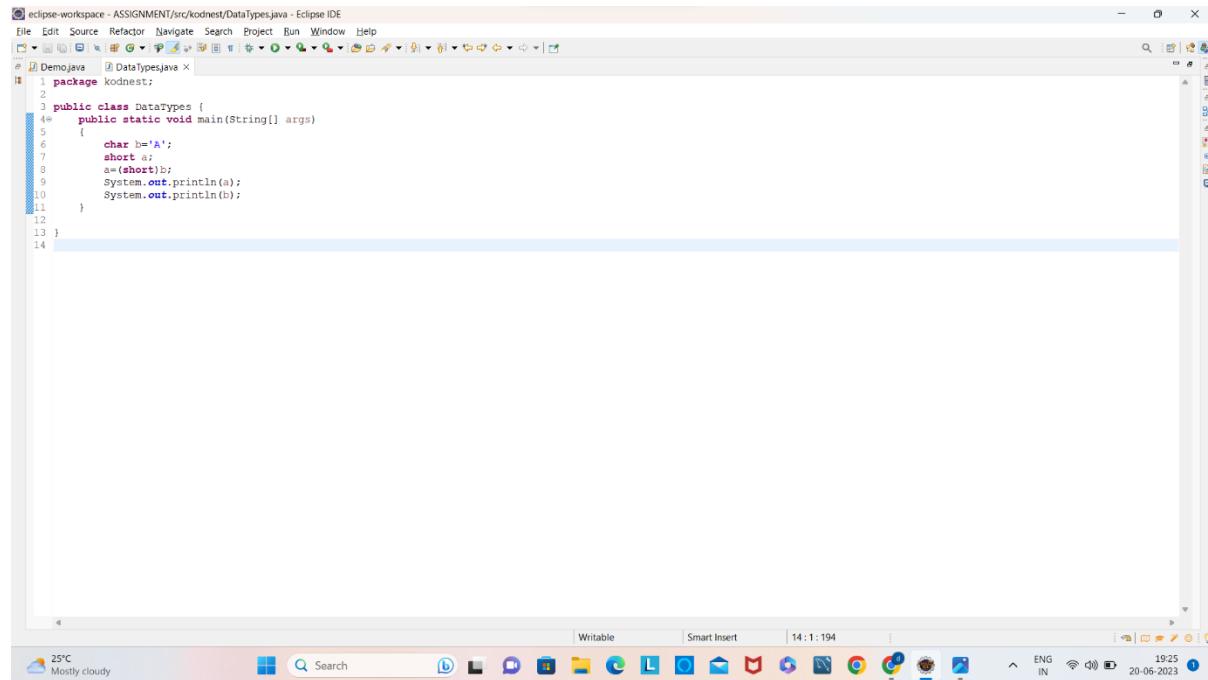
```
eclipse-workspace - ASSIGNMENT/src/kodnest/DataTypes.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Problems Javadoc Declaration Console X
terminated> DataTypes [Java Application] C:\Users\hanya\p2\pool\plugins\org.eclipse.jdt.core\1.17.0.7v20230425-1502\jre\bin\javaw.exe (20-Jun-2023, 7:24:24 pm - 7:24:24 pm) [pid: 22792]
65
A
```

The screenshot shows the Eclipse IDE interface after running the Java application. The 'Console' tab is selected, displaying the output of the program. The output shows the character 'A' and its byte representation, both printed on separate lines.

Conclusion: char to byte is possible implicitly

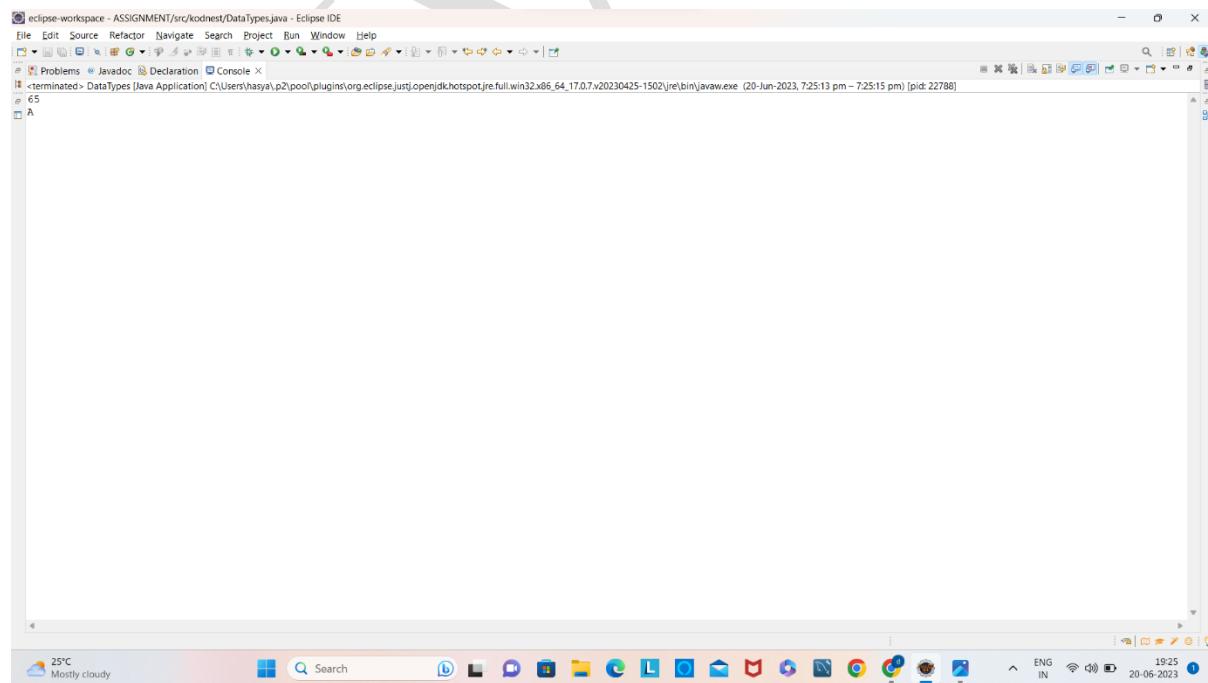
char to short:

Program:



```
eclipse-workspace - ASSIGNMENT/src/kodnest/DataTypes.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Demo.java DataTypes.java X
1 package kodnest;
2
3 public class DataTypes {
4     public static void main(String[] args) {
5         char b='A';
6         short a;
7         a=(short)b;
8         System.out.println(a);
9         System.out.println(b);
10    }
11 }
12
13 }
```

Output:

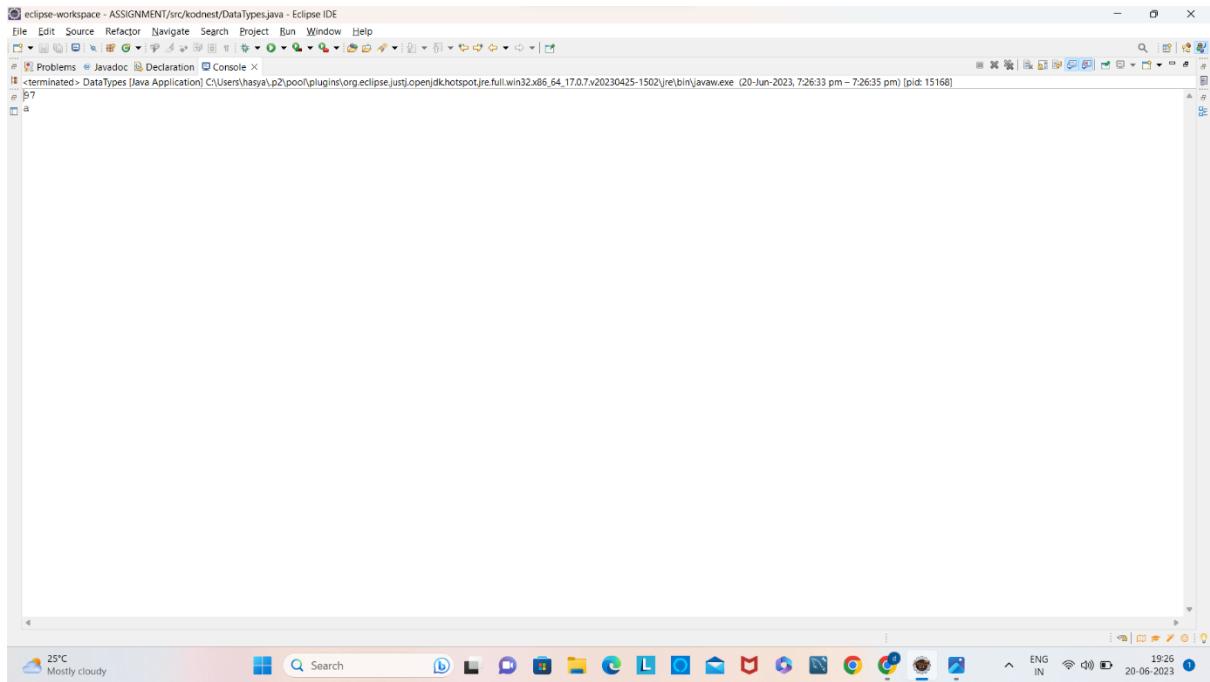


```
eclipse-workspace - ASSIGNMENT/src/kodnest/DataTypes.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Problems Javadoc Declaration Console X
terminated> DataTypes [Java Application] C:\Users\hasya\p2\pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.7.v20230425-1502\jre\bin\javaw.exe (20-Jun-2023, 7:25:13 pm - 7:25:15 pm) [pid: 22788]
65
A
```

Conclusion: char to short is possible implicitly.

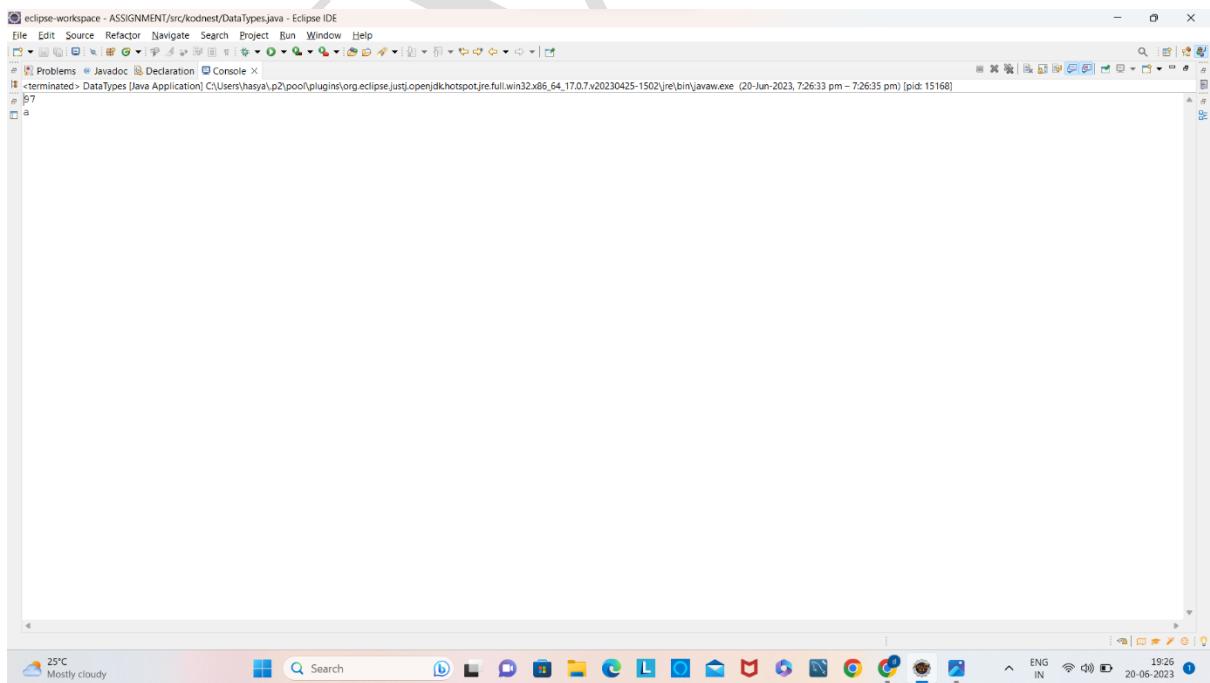
char to int:

Program:



```
char a = 'A';
System.out.println(a);
```

Output:

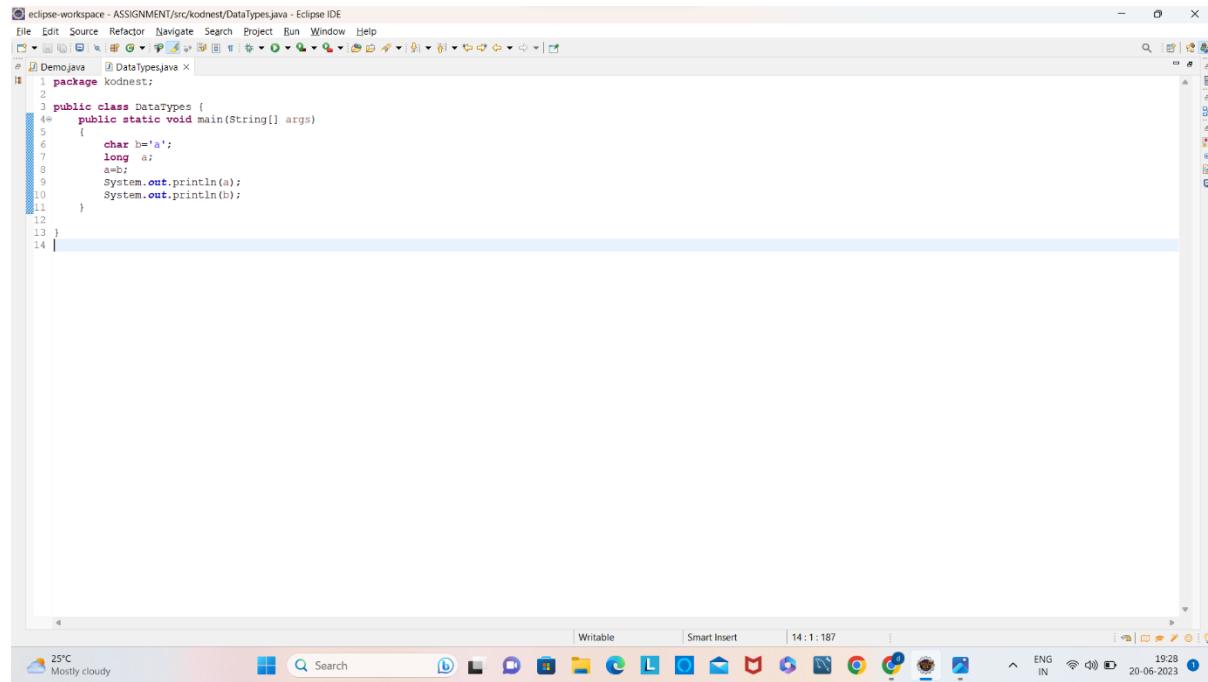


```
A
```

Conclusion: char to int is possible implicitly.

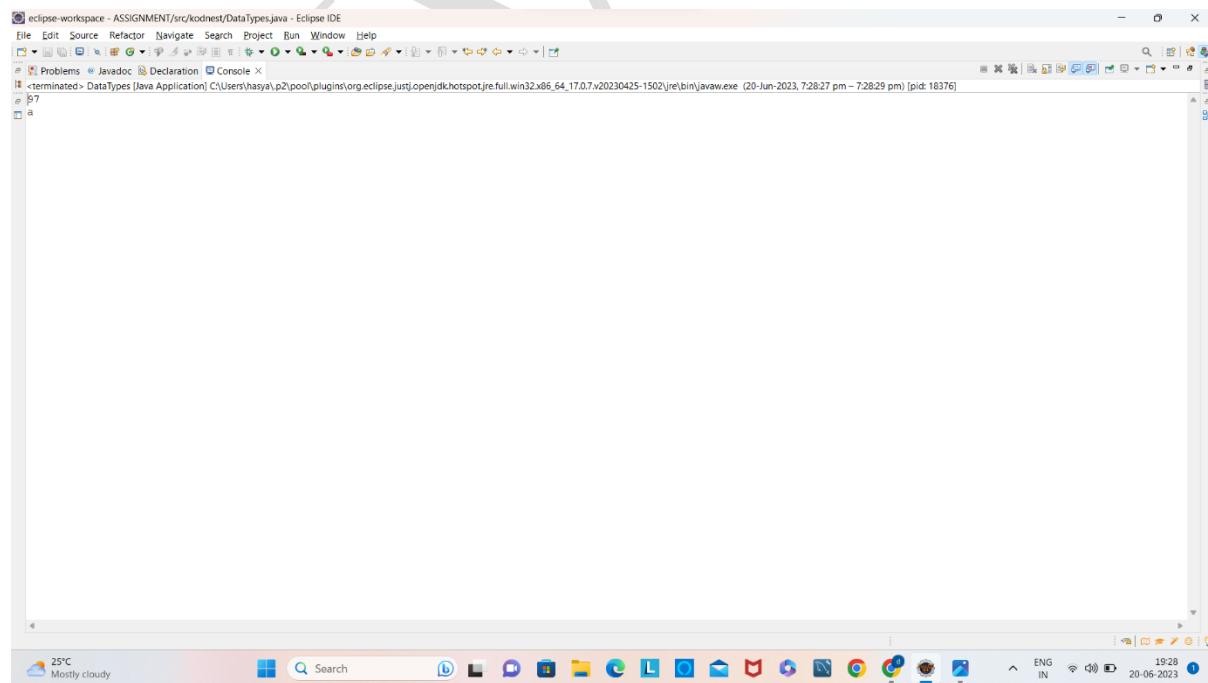
char to long:

Program:



```
1 package kodnest;
2
3 public class DataTypes {
4     public static void main(String[] args) {
5         char b='a';
6         long a;
7         a=b;
8         System.out.println(a);
9         System.out.println(b);
10    }
11 }
12
13 }
```

Output:

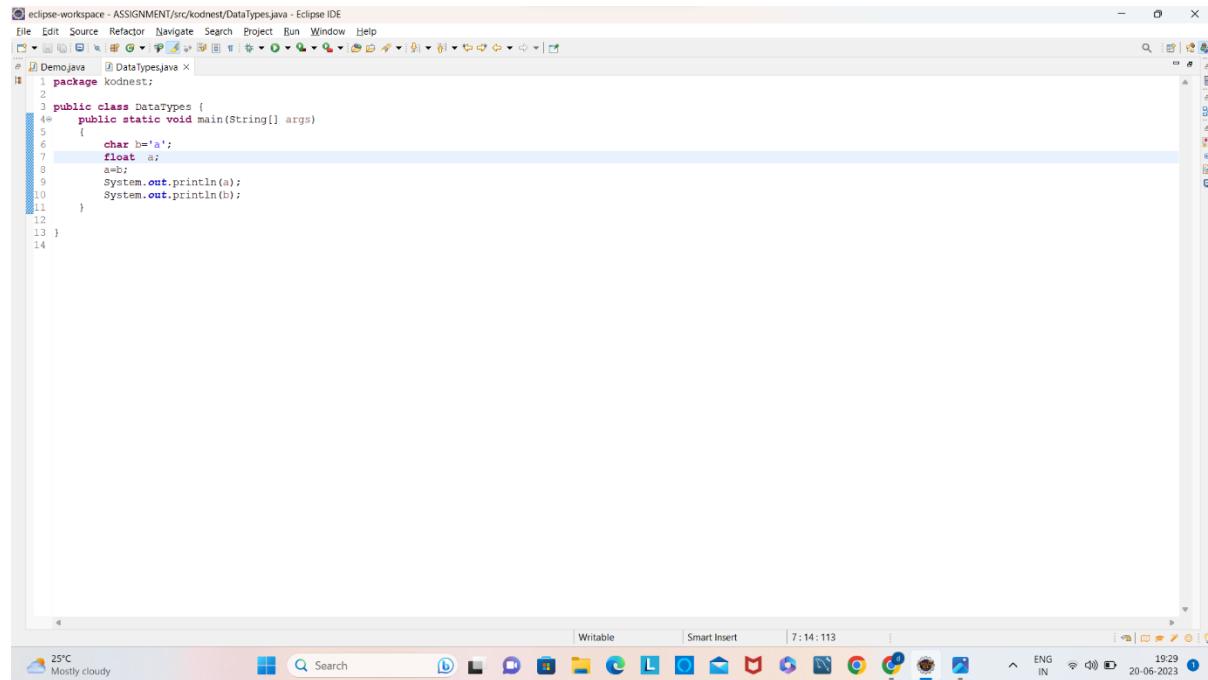


```
25°C Mostly cloudy Search 14 : 1 : 187 ENG IN 19:28 20-06-2023
eclipse-workspace - ASSIGNMENT/src/kodnest/DataTypes.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Problems Javadoc Declaration Console
terminated > DataTypes [Java Application] C:\Users\hasyl\p2\pool\plugins\org.eclipse.jdt.core\jre\full\win32\x86_64\17.0.7\20230425-1502\jre\bin\javaw.exe (20-Jun-2023, 7:28:27 pm - 7:28:29 pm) [pid: 18376]
a
b
```

Conclusion: char to long is possible implicitly.

char to float:

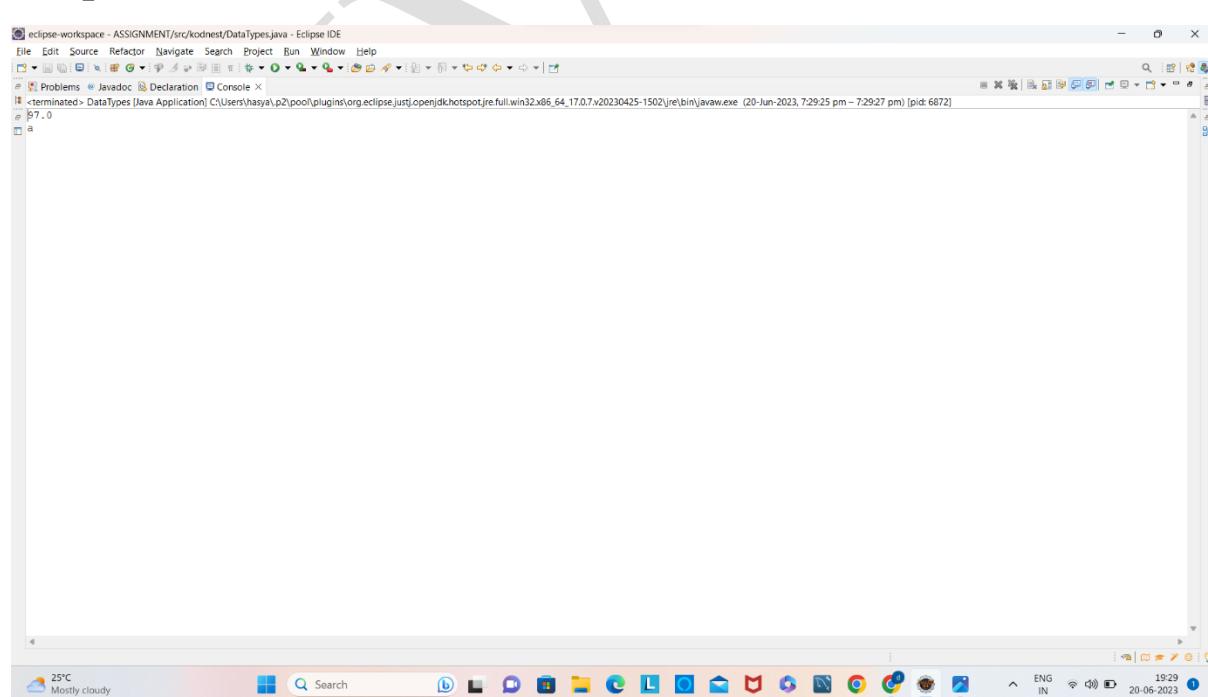
Program:



The screenshot shows the Eclipse IDE interface with a Java file named DataTypes.java open. The code defines a class DataTypes with a main method. In the main method, a character 'a' is assigned to a variable 'b', and then 'b' is cast to a float and printed. The output window shows the result of the execution.

```
1 package kodnest;
2
3 public class DataTypes {
4     public static void main(String[] args) {
5         char b='a';
6         float a;
7         a=b;
8         System.out.println(a);
9         System.out.println(b);
10    }
11
12 }
13 }
```

Output:



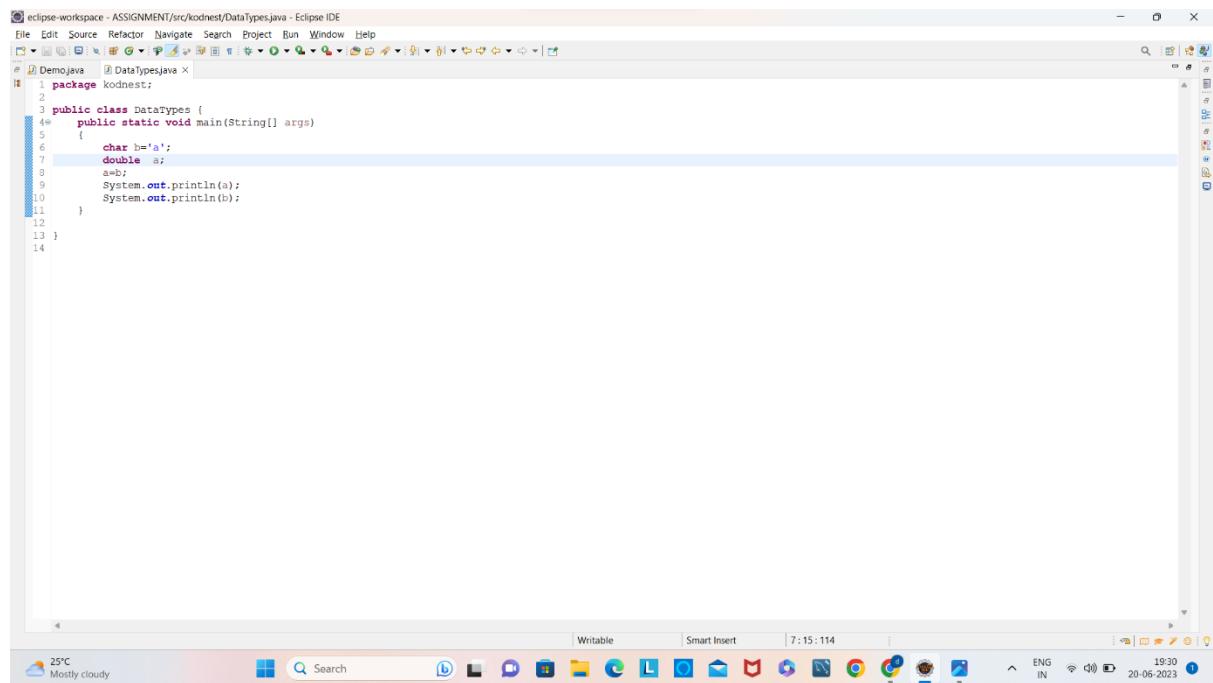
The screenshot shows the Eclipse IDE interface with the Java application running. The output window displays the results of the program execution, showing the value of 'a' as 97.0, which is the ASCII value of the character 'a'.

```
a
97.0
```

Conclusion: char to float is possible implicitly.

char to double:

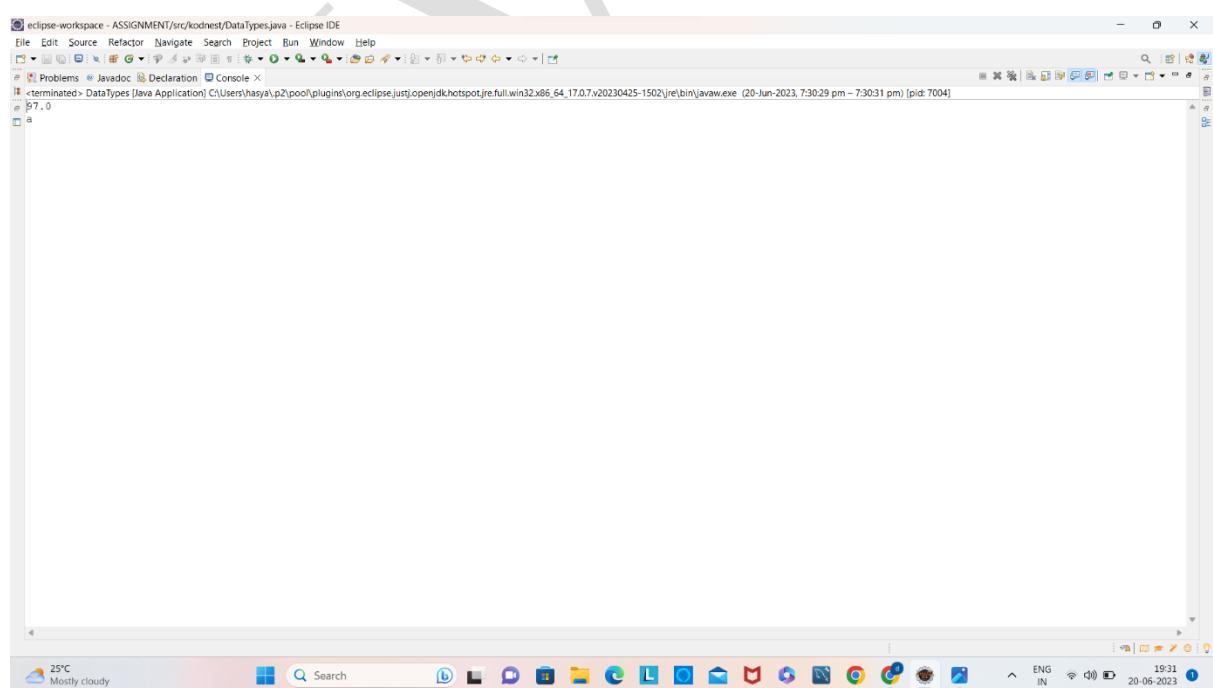
Program:



The screenshot shows the Eclipse IDE interface with a Java file named `DataTypes.java` open. The code defines a class `DataTypes` with a `main` method. Inside the `main` method, a character `'a'` is assigned to variable `a`, and its value is printed. Then, the character `'a'` is converted to a double value and assigned to variable `b`, which is then printed.

```
1 package kodnest;
2
3 public class DataTypes {
4     public static void main(String[] args) {
5         char a='a';
6         double b;
7         a=b;
8         System.out.println(a);
9         System.out.println(b);
10    }
11
12 }
13 }
```

Output:



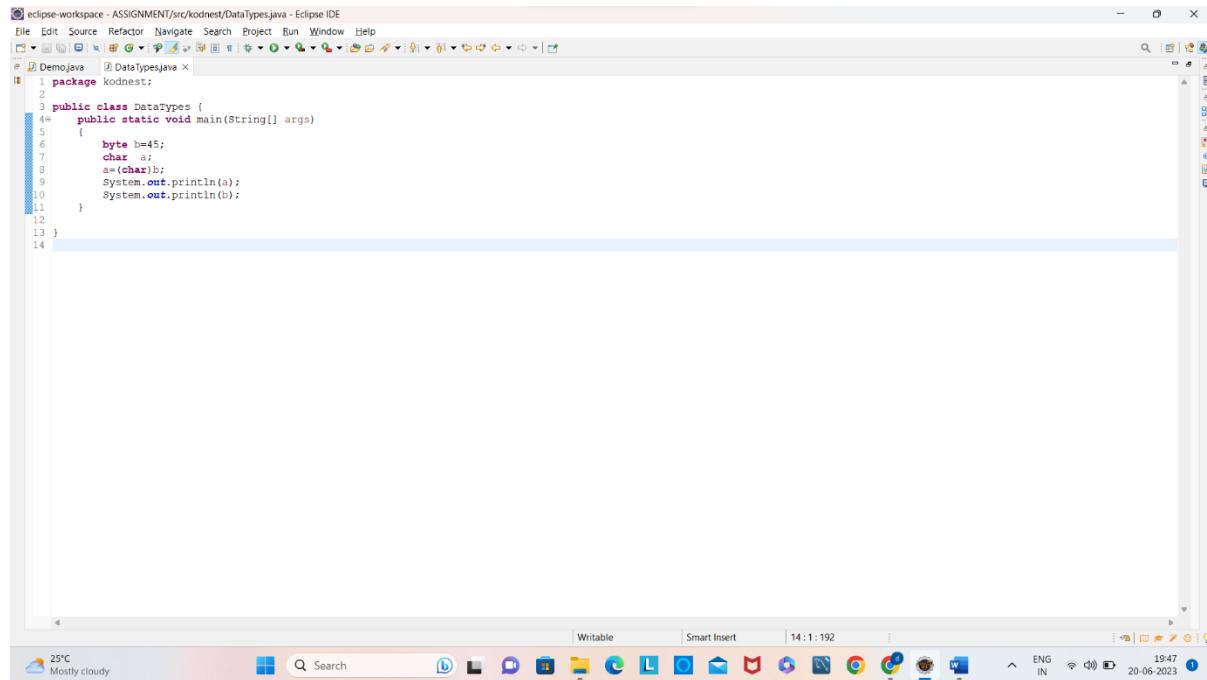
The screenshot shows the Eclipse IDE interface after running the Java application. The console output window displays the results of the program execution. The character `'a'` is printed as `a`, and its conversion to a double value is printed as `97.0`.

```
25°C Mostly cloudy 20-06-2023 19:30
[Running] DataTypes [Java Application] C:\Users\hasyl\p2\pool\plugins\org.eclipse.jdt.core\jre\full\win32\x86_64\17.0.7\20230425-1502\jre\bin\javaw.exe (20-Jun-2023, 7:30:29 pm - 7:30:31 pm) [pid: 7004]
a
97.0
```

Conclusion: char to double is possible implicitly.

byte to char:

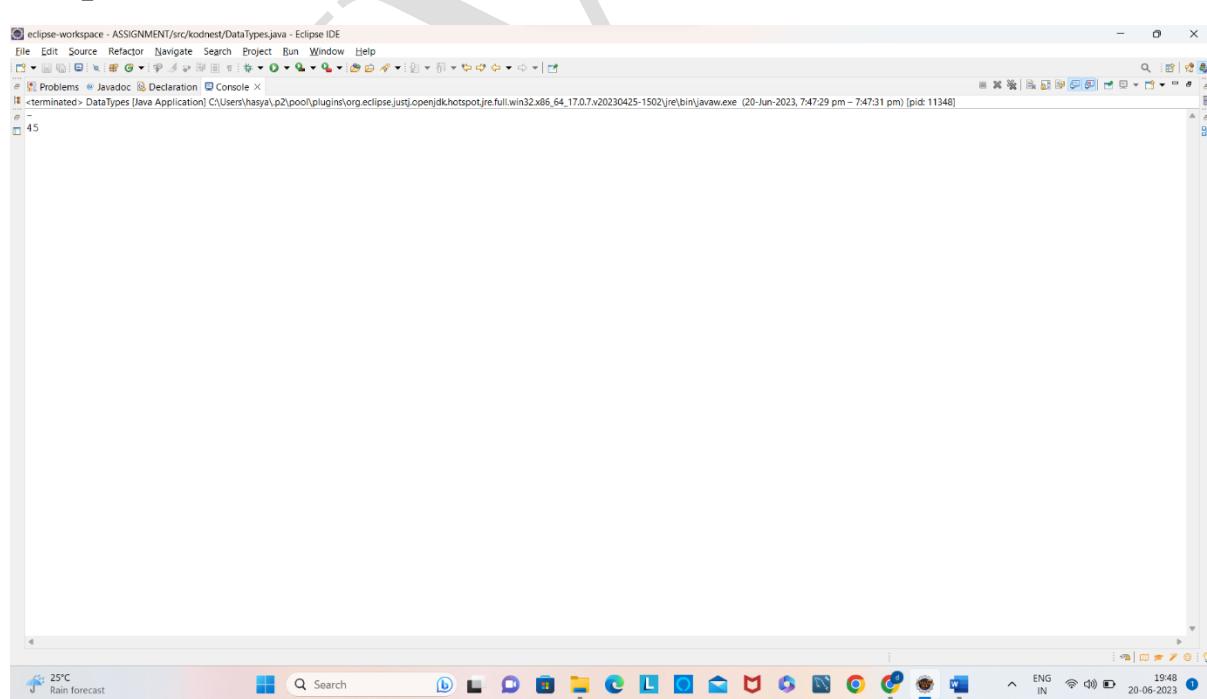
Program:



The screenshot shows the Eclipse IDE interface with a Java file named `DataTypes.java` open. The code defines a class `DataTypes` with a `main` method. Inside the `main` method, a `byte` variable `b=5;` is declared and assigned the value 5. A `char` variable `a=(char)b;` is then assigned the value of `b`. Finally, `System.out.println(a);` and `System.out.println(b);` are printed to the console. The code is as follows:

```
1 package kodnest;
2
3 public class DataTypes {
4     public static void main(String[] args) {
5         byte b=5;
6         char a;
7         a=(char)b;
8         System.out.println(a);
9         System.out.println(b);
10    }
11 }
12
13 }
```

Output:



The screenshot shows the Eclipse IDE interface after running the Java application. The output window displays the results of the `System.out.println` statements. The first line shows the character representation of the byte value 5, which is the letter 'A' in the ASCII table. The second line shows the original byte value 5. The output window title is "terminated - DataTypes [Java Application]". The system tray at the bottom indicates the date as 20-06-2023 and the time as 19:48.

```
1 A
2 5
```

Conclusion: byte to char is possible explicitly.

byte to short:

Program:

```
Demojava ✘
1 package practice;
2
3 public class Demo{
4     public static void main(String[]args){
5         byte a=20;
6         short b;
7         b=a;
8         System.out.println(a);
9         System.out.println(b);
10    }
11 }
12
```

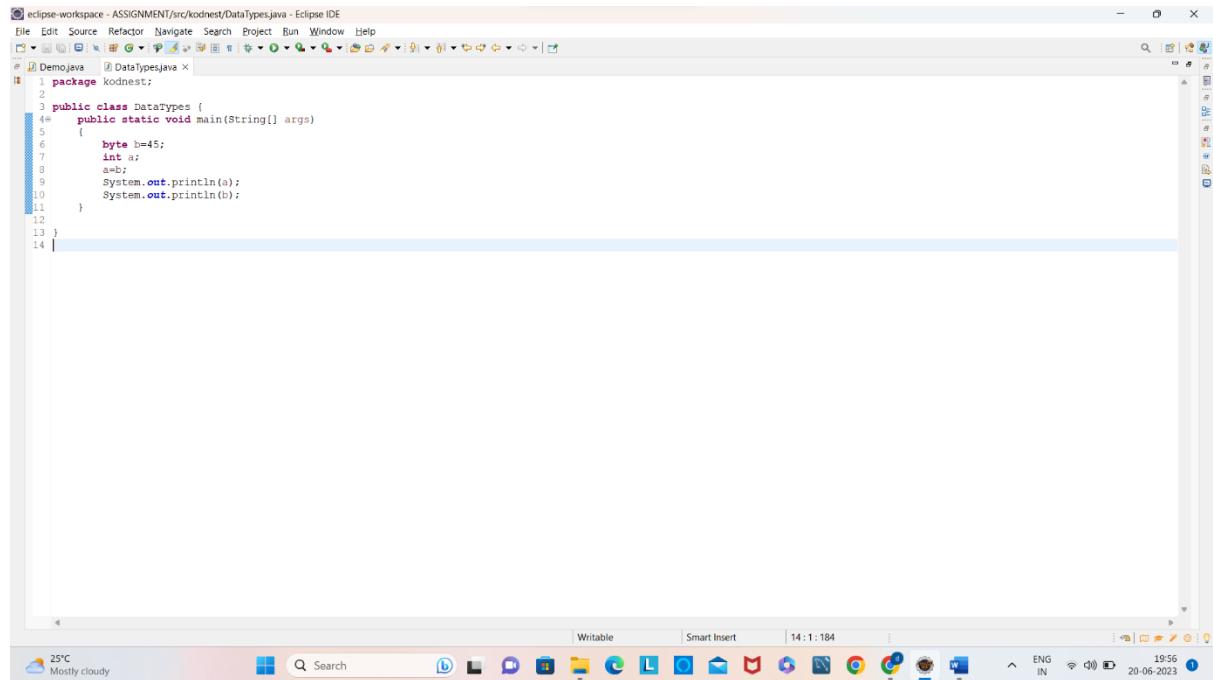
Output:

```
Markers Properties Servers Data Source Explorer Snippets Console ✘
<terminated> Demo [Java Application] C:\Program Files\Java\jre1.8.0_211\bin\javaw.exe (Nov 30, 2022, 10:43:58 PM)
20
20
```

Conclusion: byte to short is possible implicitly.

byte to int:

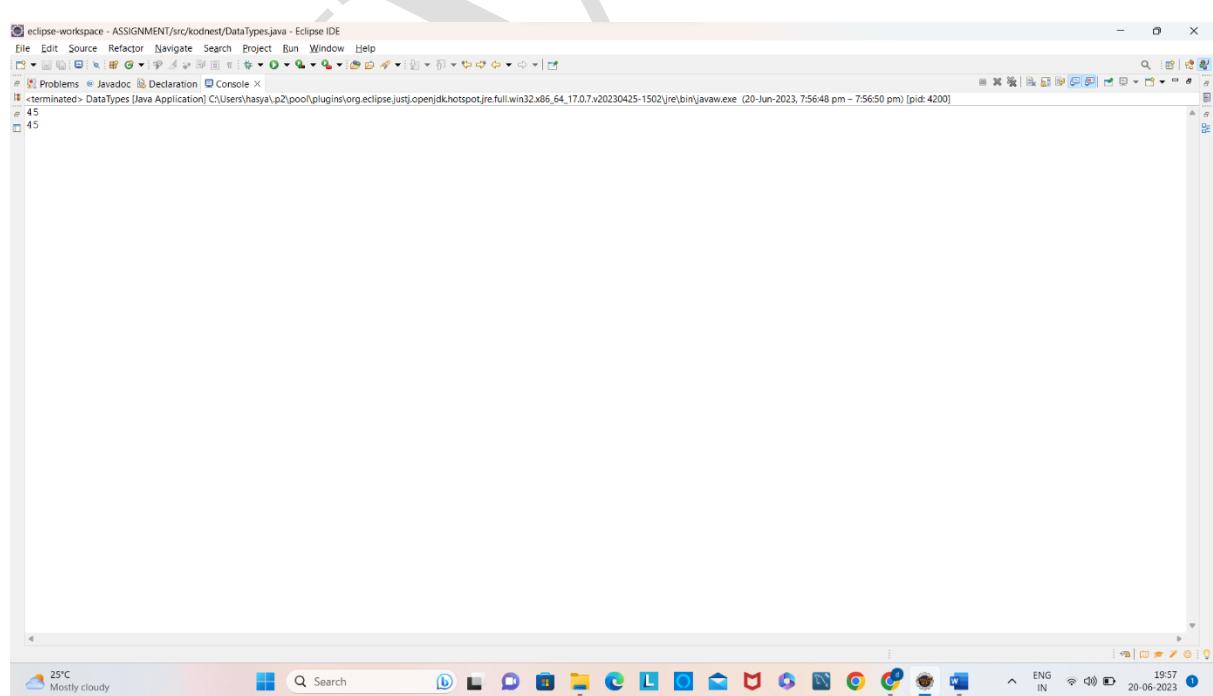
Program:



The screenshot shows the Eclipse IDE interface with a Java file named DataTypes.java open. The code defines a class DataTypes with a main method. Inside the main method, a byte variable b is assigned the value 45, and its value is printed to the console. An int variable a is then assigned the value of b, and its value is also printed to the console.

```
1 package kodnest;
2
3 public class DataTypes {
4     public static void main(String[] args) {
5         byte b=45;
6         int a;
7         a=b;
8         System.out.println(a);
9         System.out.println(b);
10    }
11 }
12
13 }
```

Output:



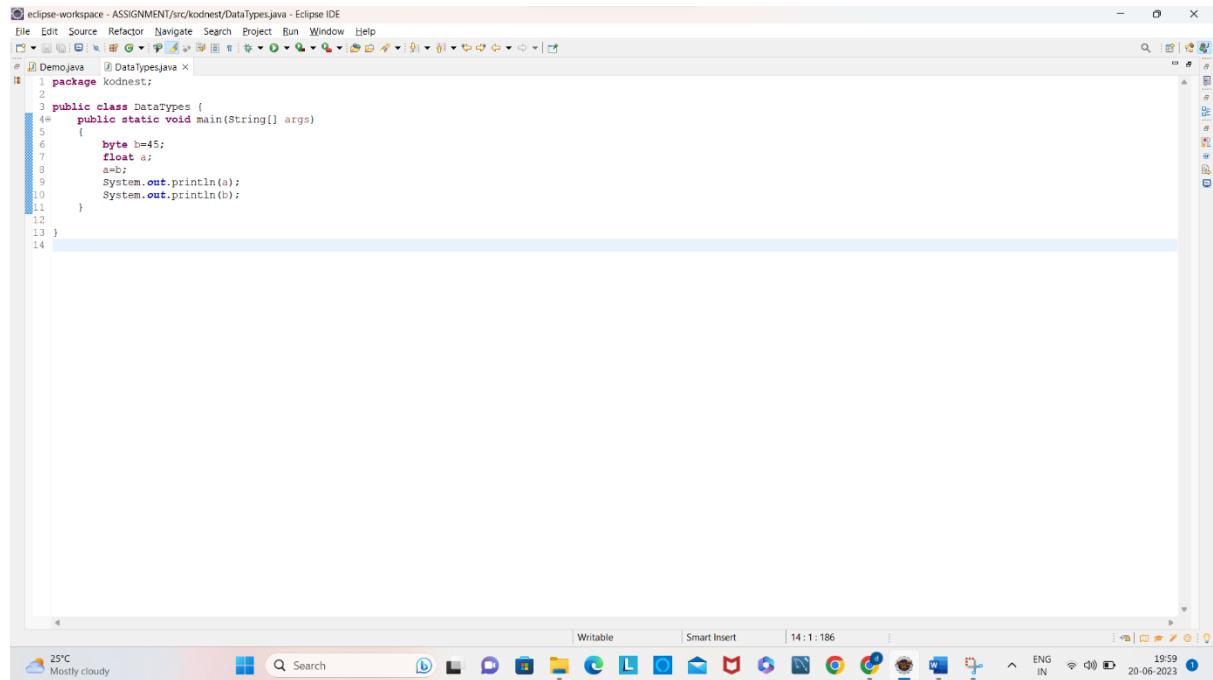
The screenshot shows the Eclipse IDE interface with the Java application running. The output window displays the results of the program execution. The byte variable b is printed as 45, and the int variable a is also printed as 45, demonstrating that a byte can be implicitly converted to an int.

```
45
45
```

Conclusion: byte to int is possible implicitly.

byte to float:

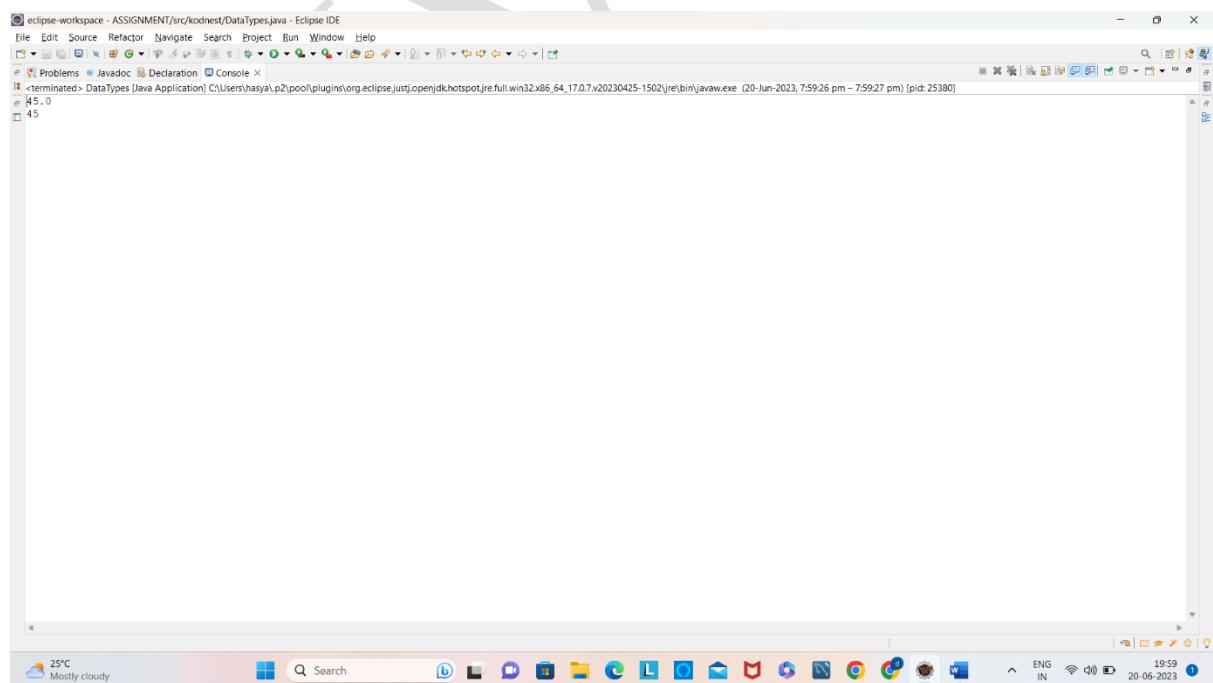
Program:



The screenshot shows the Eclipse IDE interface with a Java file named `DataTypes.java` open. The code defines a class `DataTypes` with a `main` method. Inside the `main` method, a `byte` variable `b=45;` is assigned the value 45. This value is then cast to a `float` variable `a=b;`. Finally, both variables are printed to the console using `System.out.println`.

```
1 package kodnest;
2
3 public class DataTypes {
4     public static void main(String[] args) {
5         byte b=45;
6         float a;
7         a=b;
8         System.out.println(a);
9         System.out.println(b);
10    }
11 }
12
13 }
```

Output:



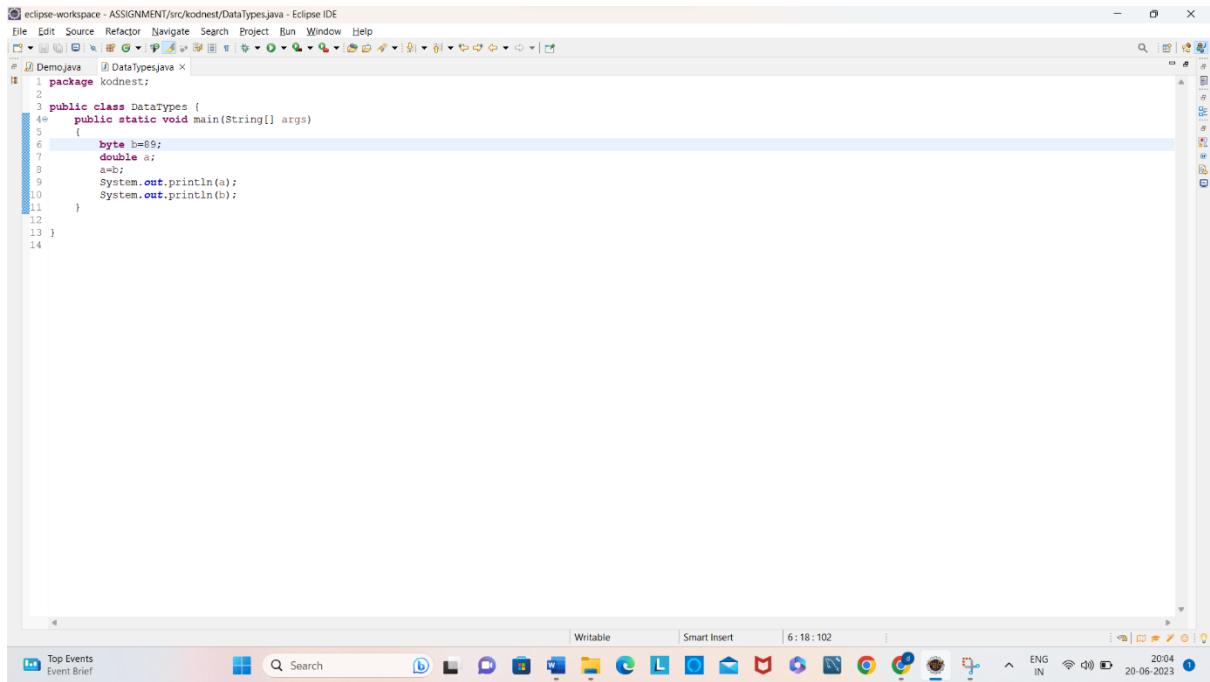
The screenshot shows the Eclipse IDE interface after running the Java application. The `Console` tab is active, displaying the output of the `System.out.println` statements. The output shows the value 45 followed by a new line.

```
45
```

Conclusion: byte to float is possible implicitly.

byte to double:

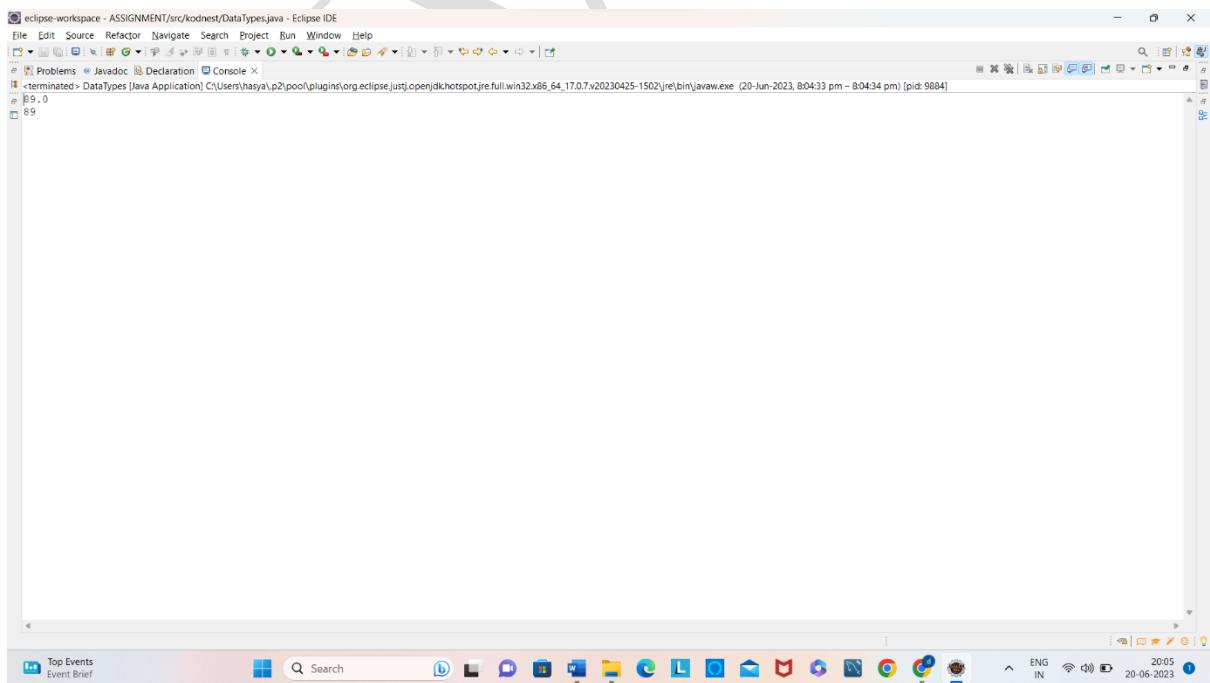
Program:



The screenshot shows the Eclipse IDE interface with a Java file named `DataTypes.java` open. The code defines a class `DataTypes` with a `main` method. Inside the `main` method, a `byte` variable `b=9` is assigned to a `double` variable `a`. Both variables are then printed to the console using `System.out.println`.

```
1 package kodnest;
2
3 public class DataTypes {
4     public static void main(String[] args) {
5         byte b=9;
6         double a;
7         a=b;
8         System.out.println(a);
9         System.out.println(b);
10    }
11
12 }
13 }
```

Output:



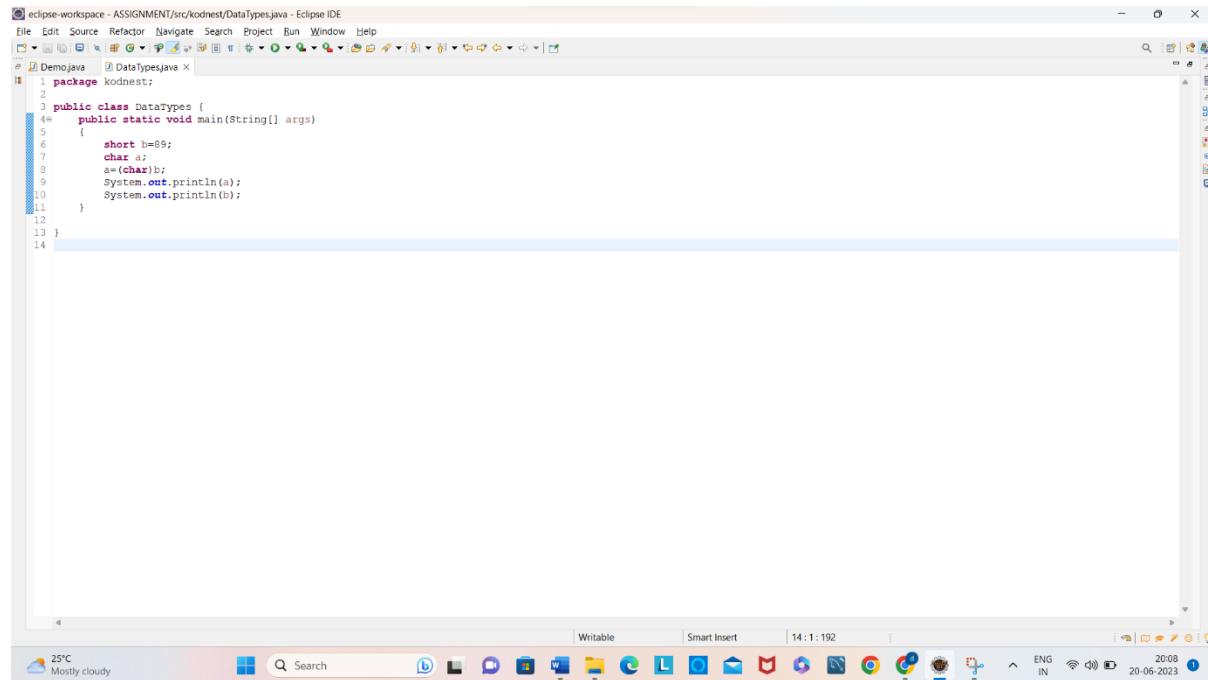
The screenshot shows the Eclipse IDE interface with the `Console` tab selected. The output window displays the results of the `System.out.println` statements from the previous code. The first line shows the value of `a` as `9.0`, and the second line shows the value of `b` as `9`.

```
9.0
9
```

Conclusion: byte to double is possible implicitly.

short to char:

Program:

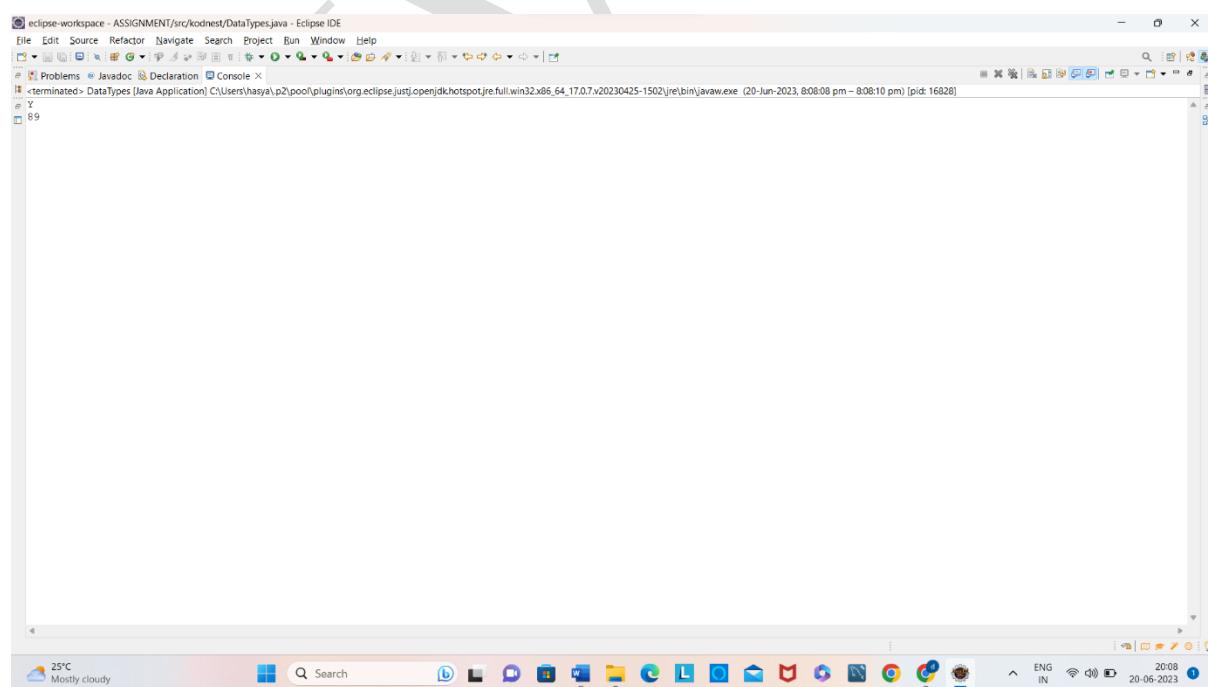


The screenshot shows the Eclipse IDE interface with the following code in the editor:

```
1 package kodnest;
2
3 public class DataTypes {
4     public static void main(String[] args) {
5         short b=89;
6         char a;
7         a=(char)b;
8         System.out.println(a);
9         System.out.println(b);
10    }
11 }
12
13 }
```

The code demonstrates that a short variable can be converted to a char variable by casting it. The output will be the character represented by the value of the short variable.

Output:



The screenshot shows the Eclipse IDE interface with the following output in the Console view:

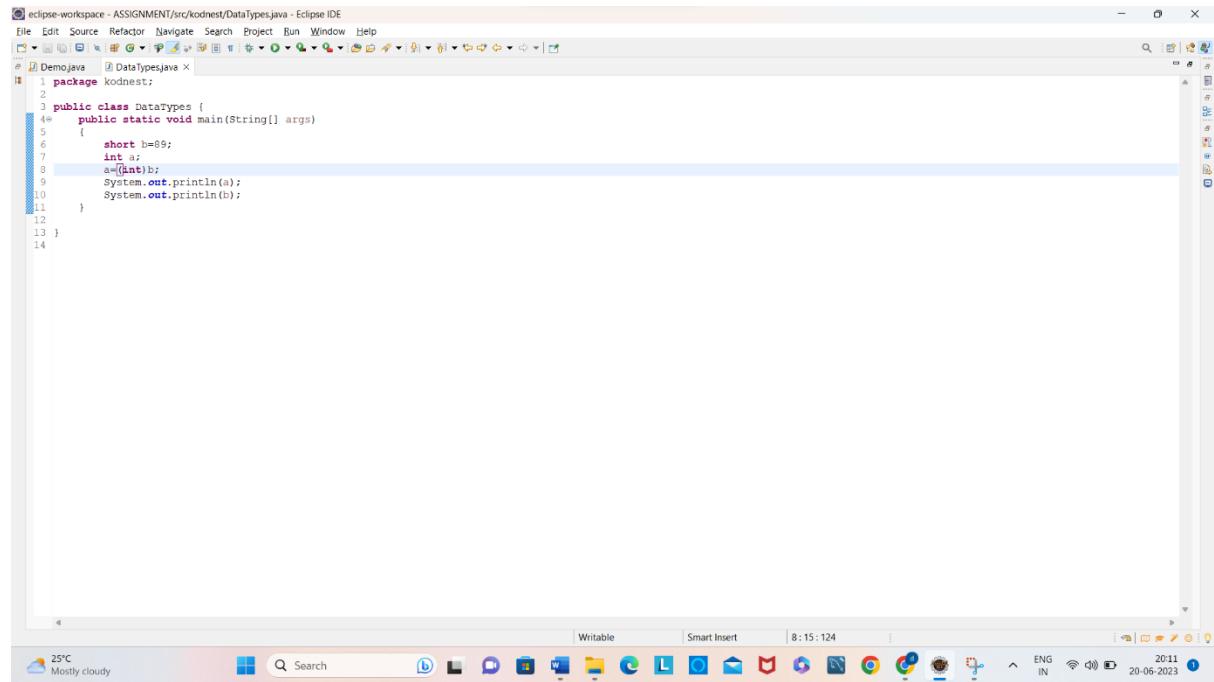
```
terminated> DataTypes [Java Application] C:\Users\hasya\p2pool\plugins\org.eclipse.jdt.openjdk.hotspot.jre.full.win32.x86_64_17.0.7.v20230425-1502\jre\bin\javaw.exe (20-Jun-2023, 8:08:08 pm - 8:08:10 pm) [pid: 16628]
89
```

The output shows the character '89' printed to the console, which corresponds to the character representation of the value 89 in ASCII.

Conclusion: short to char is possible explicitly.

short to int:

Program:

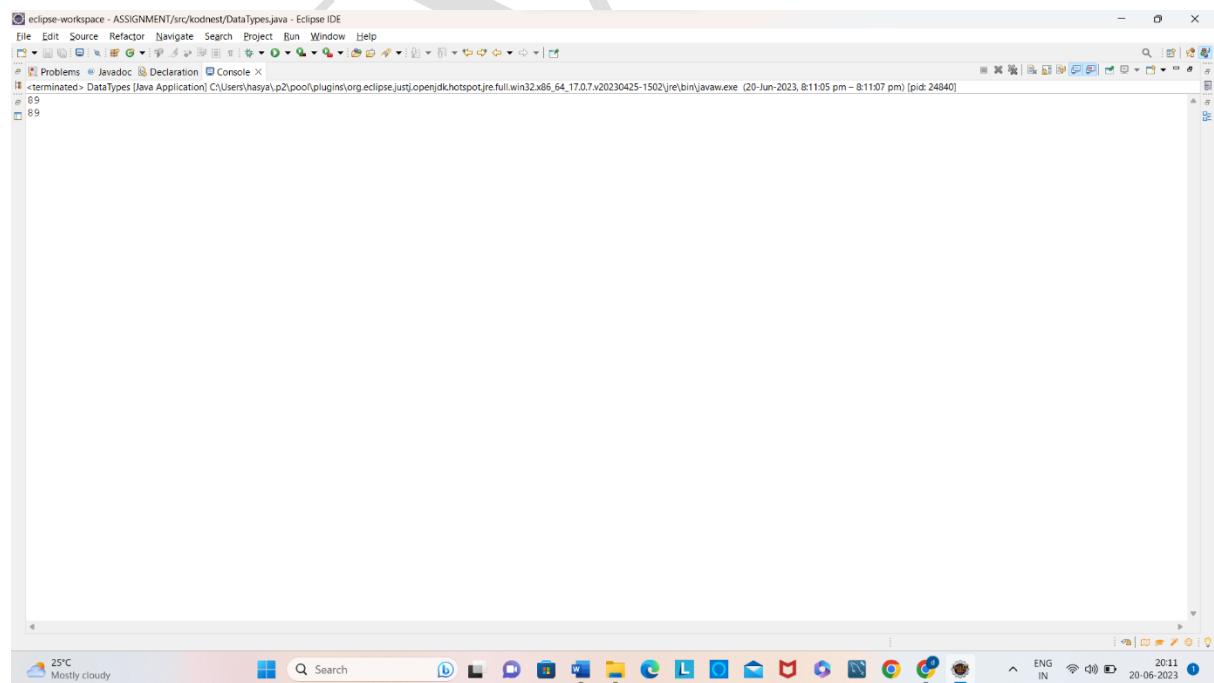


The screenshot shows the Eclipse IDE interface with the following code in the editor:

```
1 package kodnest;
2
3 public class DataTypes {
4     public static void main(String[] args) {
5         short b=89;
6         int a;
7         a=(int)b;
8         System.out.println(a);
9         System.out.println(b);
10    }
11
12 }
13 }
```

The code defines a class named `DataTypes` with a `main` method. It declares a `short` variable `b` with the value 89, converts it to an `int` and stores it in `a`, then prints both `a` and `b` to the console.

Output:



The screenshot shows the Eclipse IDE interface with the following output in the `Console` tab:

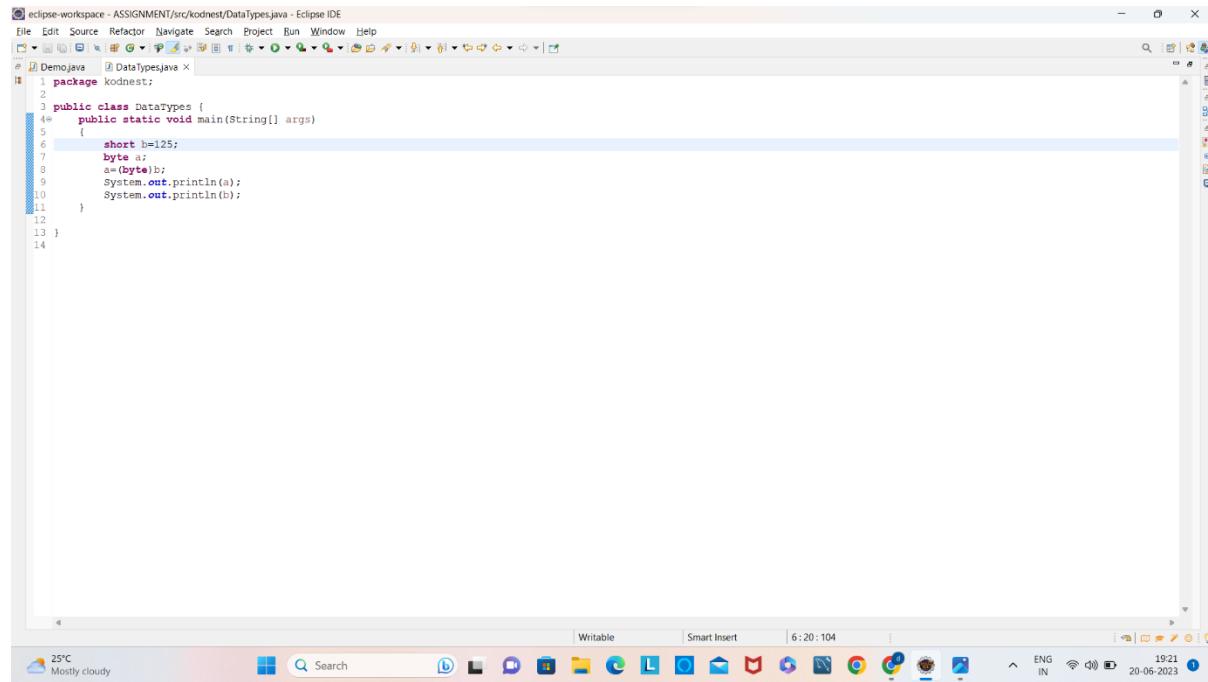
```
terminated> DataTypes [Java Application] C:\Users\hasya\p2pool\plugins\org.eclipse.jdt.openjdk.hotspot.jre.full.win32.x86_64_17.0.7.v20230425-1502\jre\bin\javaw.exe (20-Jun-2023, 8:11:05 pm - 8:11:07 pm) [pid: 24840]
89
89
```

The output shows the program has terminated successfully, printing the value 89 twice to the console.

Conclusion: short to int is possible implicitly.

short to byte:

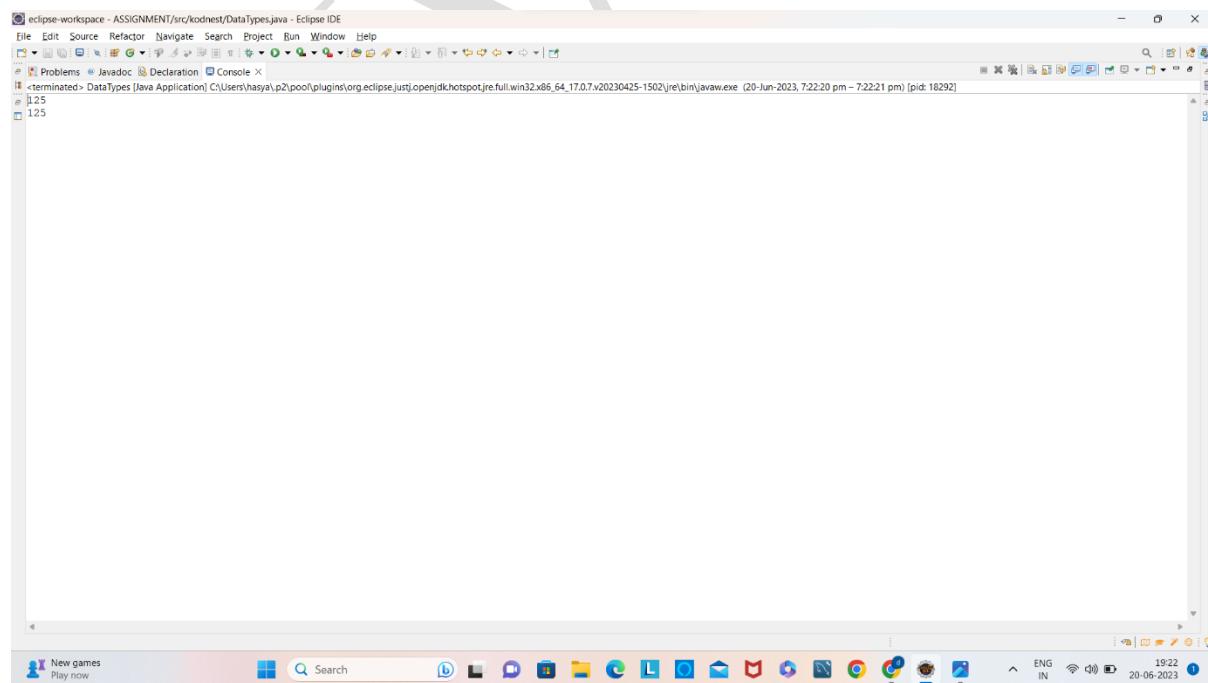
Program:



The screenshot shows the Eclipse IDE interface with a Java file named `DataTypes.java` open. The code defines a class `DataTypes` with a `main` method. Inside the `main` method, a `short` variable `b=125;` is declared and assigned the value 125. This value is then cast to a `byte` type and stored in variable `a=(byte)b;`. Finally, both variables are printed to the console using `System.out.println`.

```
1 package kodnest;
2
3 public class DataTypes {
4     public static void main(String[] args) {
5         short b=125;
6         byte a;
7         a=(byte)b;
8         System.out.println(a);
9         System.out.println(b);
10    }
11 }
12
13 }
```

Output:



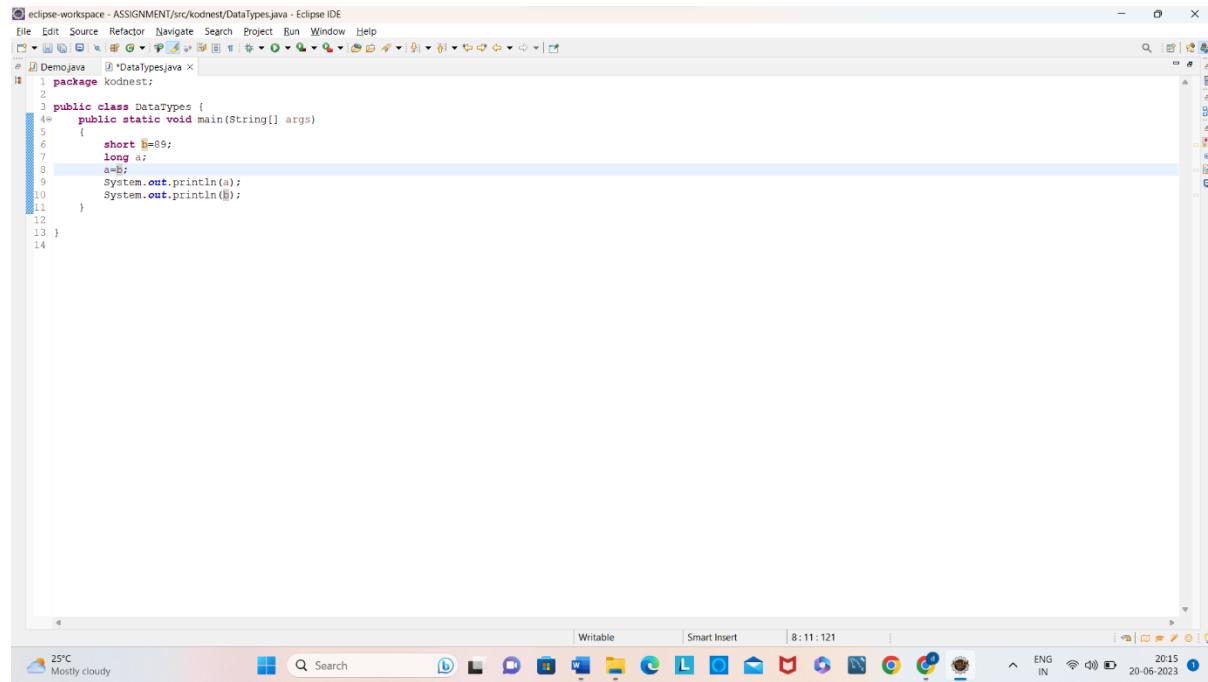
The screenshot shows the Eclipse IDE interface after running the Java application. The output window displays the results of the `System.out.println` statements. The first line shows the value of variable `a`, which is explicitly cast from `short` to `byte`, resulting in the value 125. The second line shows the value of variable `b`, which is the original `short` value, also resulting in 125.

```
125
125
```

Conclusion: short to byte is possible explicitly.

short to long:

Program:

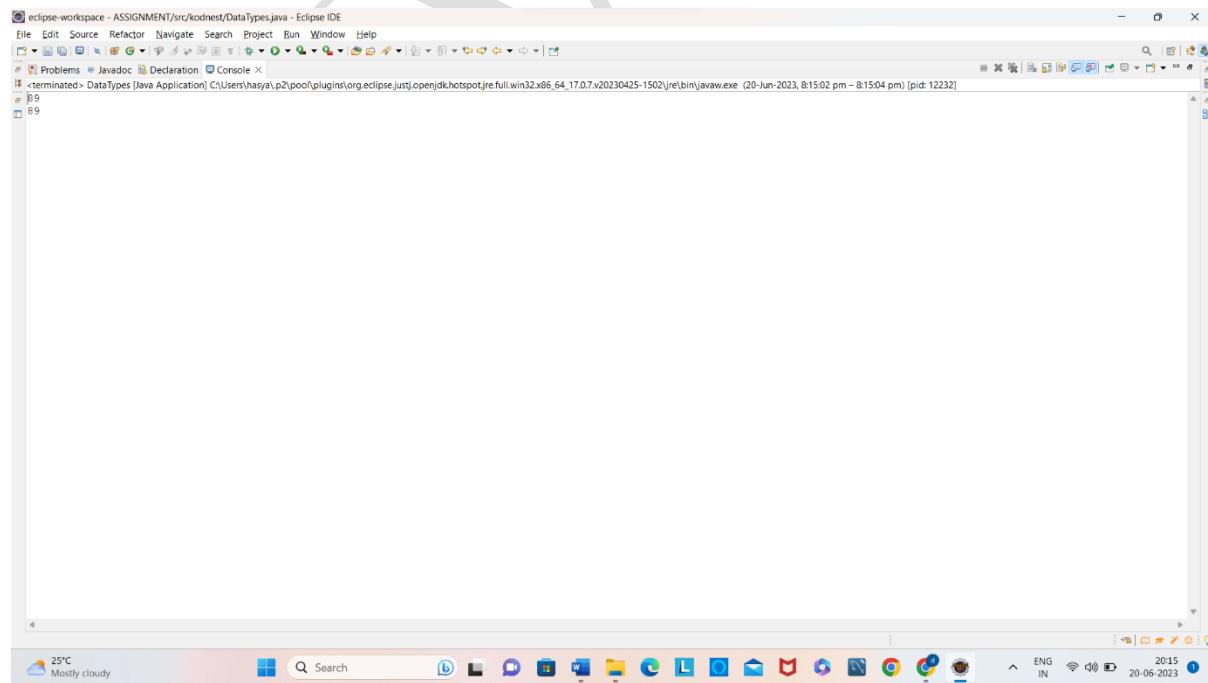


The screenshot shows the Eclipse IDE interface with a Java file named "DataTypes.java" open. The code is as follows:

```
1 package kodnest;
2
3 public class DataTypes {
4     public static void main(String[] args) {
5         short s=89;
6         long l;
7         l=s;
8         System.out.println(s);
9         System.out.println(l);
10    }
11
12 }
13 }
```

The code demonstrates a narrowing cast from a short variable to a long variable. The output of the program is shown in the Eclipse IDE's Run tab, which displays the values 89 and 89.

Output:



The screenshot shows the Eclipse IDE interface with a Java file named "DataTypes.java" open. The code is identical to the one in the previous screenshot:

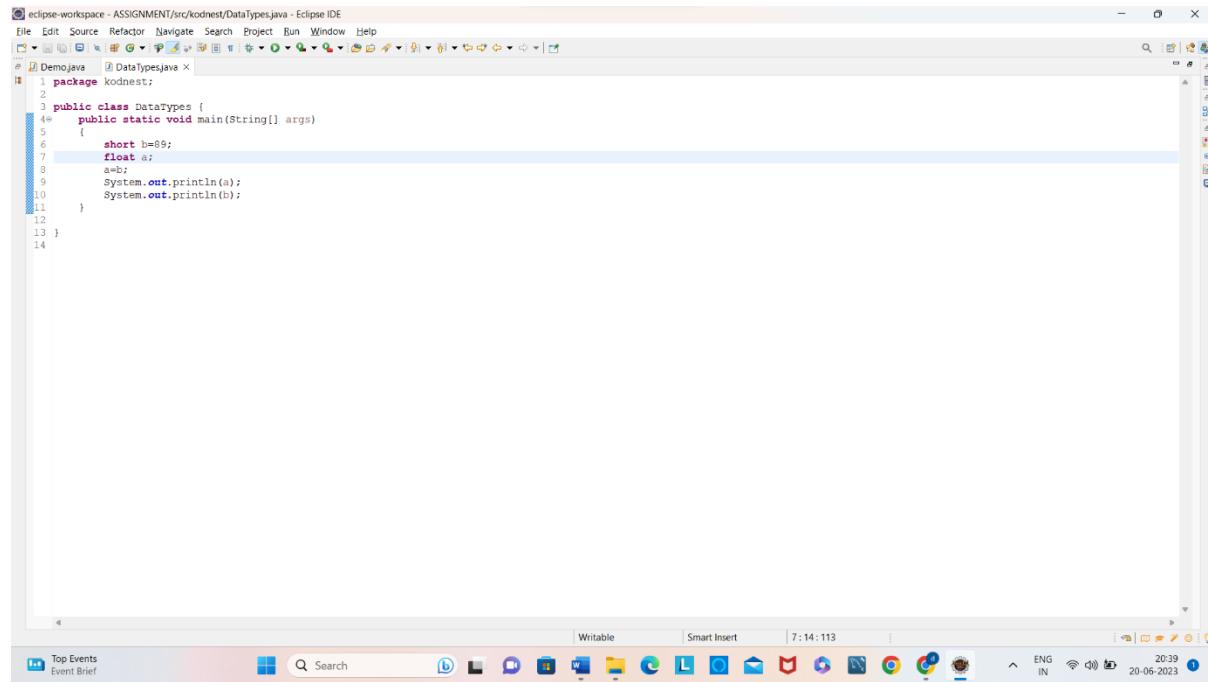
```
1 package kodnest;
2
3 public class DataTypes {
4     public static void main(String[] args) {
5         short s=89;
6         long l;
7         l=s;
8         System.out.println(s);
9         System.out.println(l);
10    }
11
12 }
13 }
```

The output of the program is shown in the Eclipse IDE's Run tab, which displays the values 89 and 89.

Conclusion: short to long is possible implicitly.

short to float:

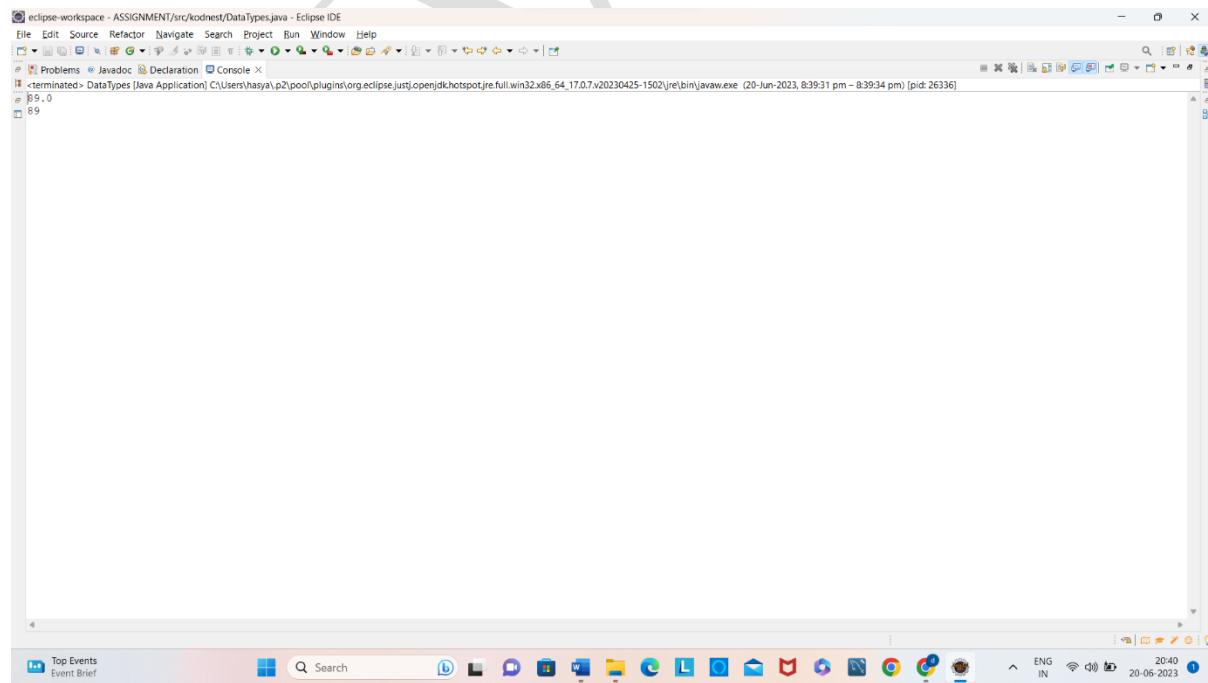
Program:



The screenshot shows the Eclipse IDE interface with a Java file named `DataTypes.java` open. The code defines a class `DataTypes` with a `main` method. In the `main` method, a `short` variable `a` is assigned the value `89`, and a `float` variable `b` is assigned the value of `a`. Both variables are then printed to the console using `System.out.println`.

```
1 package kodnest;
2
3 public class DataTypes {
4     public static void main(String[] args) {
5         short a=89;
6         float b;
7         a=b;
8         System.out.println(a);
9         System.out.println(b);
10    }
11
12 }
13 }
```

Output:



The screenshot shows the Eclipse IDE interface after running the Java application. The `Console` tab is active, displaying the output of the `System.out.println` statements. The output shows the value `89` for both `a` and `b`, indicating that the implicit conversion from `short` to `float` did not change the value.

```
89
89
```

Conclusion: short to float is possible implicitly.

short to double:

Program:



```
eclipse-workspace - ASSIGNMENT/src/kodnest/DataTypes.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
DemoJava DataTypes.java
1 package kodnest;
2
3 public class DataTypes {
4     public static void main(String[] args) {
5         short b=87;
6         double a;
7         a=b;
8         System.out.println(a);
9         System.out.println(b);
10    }
11 }
12
13 }
```

Output:

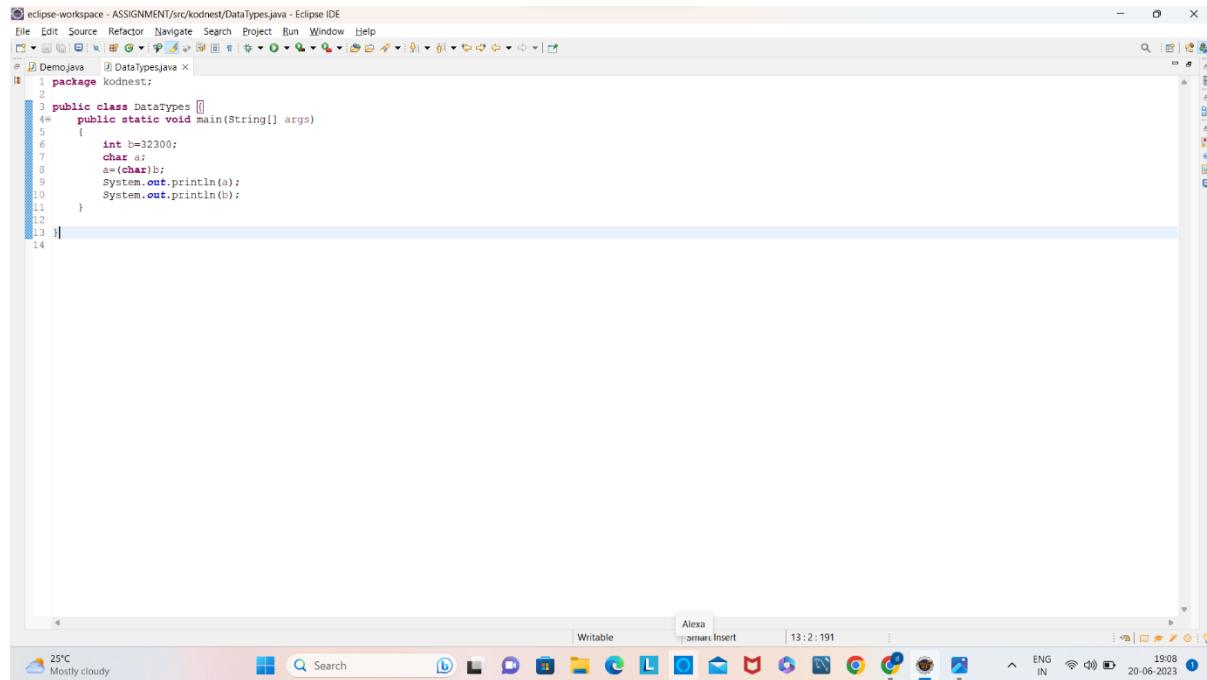


```
eclipse-workspace - ASSIGNMENT/src/kodnest/DataTypes.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Problems Javadoc Declaration Console
terminated - DataTypes [Java Application] C:\Users\hasya\p2pool\plugins\org.eclipse.jdt.openjdk.hotspot.jre.full.win32.x86_64_17.0.7.v20230425-1502\jre\bin\javaw.exe (20-Jun-2023, 8:43:14 pm - 8:43:15 pm) [pid: 9112]
87.0
87
```

Conclusion: short to double is possible implicitly.

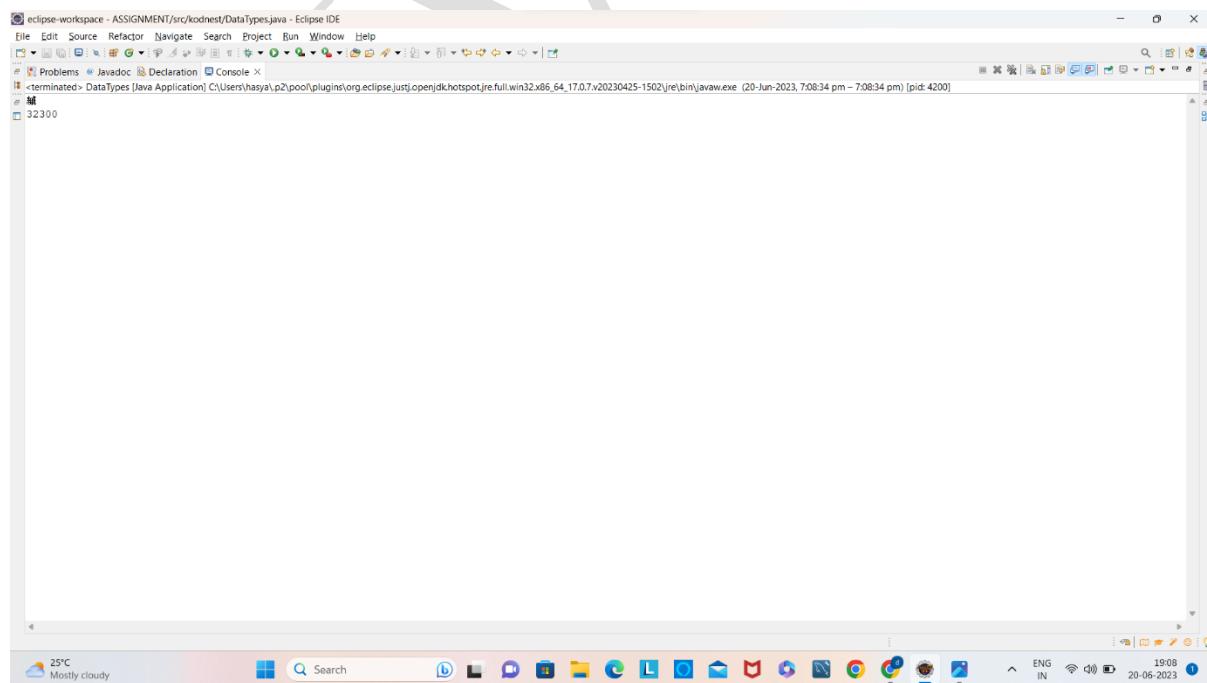
int to char:

Program:



```
1 eclipse-workspace - ASSIGNMENT/src/kodnest/DataTypes.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
2 DemoJava DataTypes.java X
3 package kodnest;
4
5 public class DataTypes {
6     public static void main(String[] args) {
7         int b=32300;
8         char a;
9         a=(char)b;
10        System.out.println(a);
11        System.out.println(b);
12    }
13 }
14
```

Output:

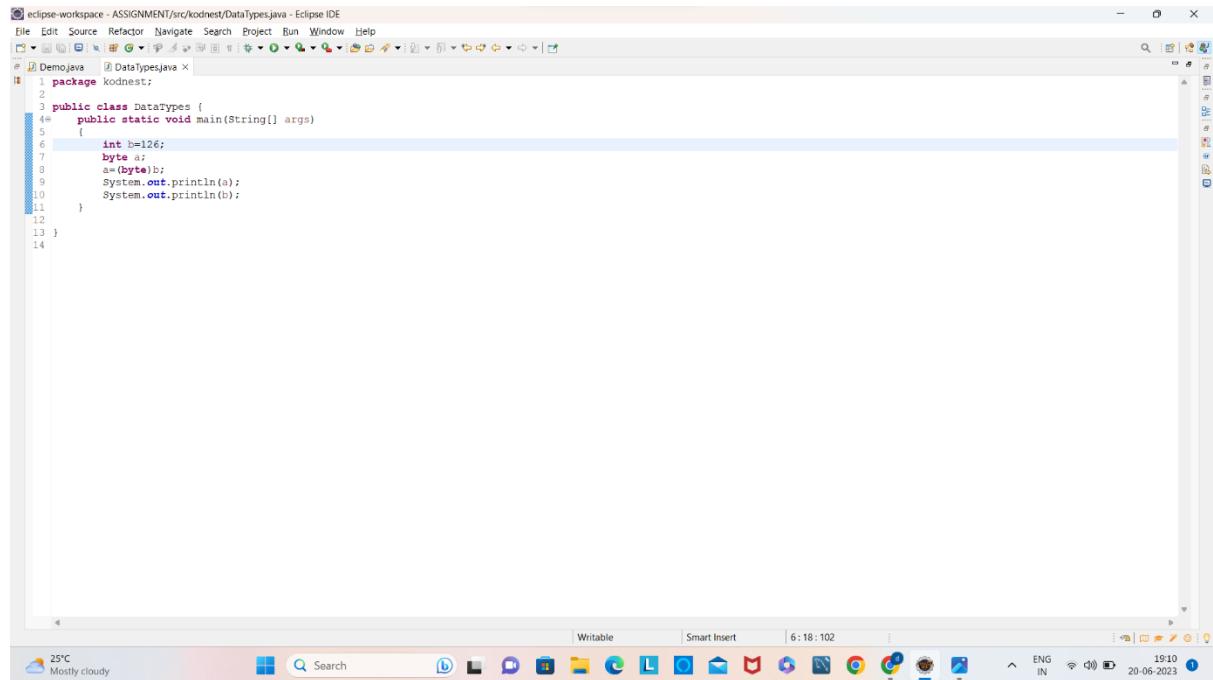


```
1 eclipse-workspace - ASSIGNMENT/src/kodnest/DataTypes.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
2 Problems Javadoc Declaration Console X
3 [terminated - DataTypes [Java Application] C:\Users\hasya\p2pool\plugins\org.eclipse.jdt.openjdk.hotspot.jre.full.win32.x86_64_17.0.7.v20230425-1502\jre\bin\javaw.exe (20-Jun-2023, 7:08:34 pm - 7:08:34 pm) [pid: 4200]
4
5 32300
```

Conclusion: int to char is possible implicitly.

int to byte:

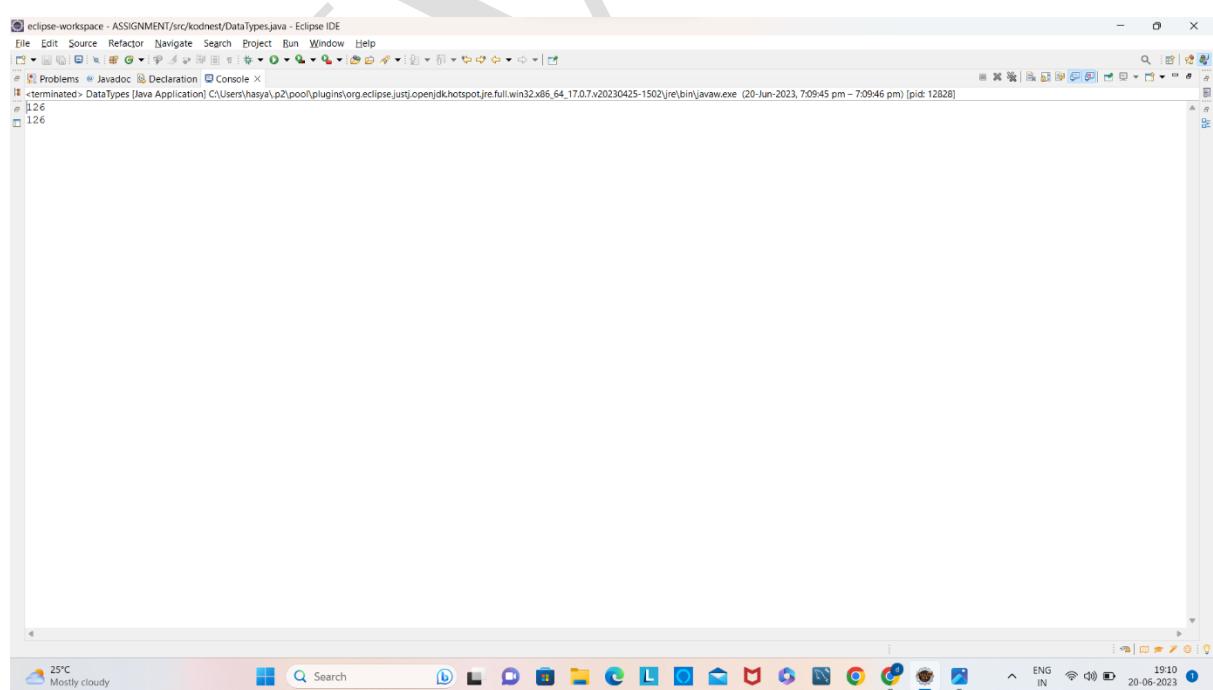
Program:



The screenshot shows the Eclipse IDE interface with a Java file named `DataTypes.java` open. The code defines a class `DataTypes` with a `main` method. Inside the `main` method, an `int` variable `b=126` is assigned to a `byte` variable `a`. Both variables are then printed to the console using `System.out.println`.

```
1 package kodnest;
2
3 public class DataTypes {
4     public static void main(String[] args) {
5         int b=126;
6         byte a ;
7         a=(byte)b;
8         System.out.println(a);
9         System.out.println(b);
10    }
11 }
12
13 }
```

Output:



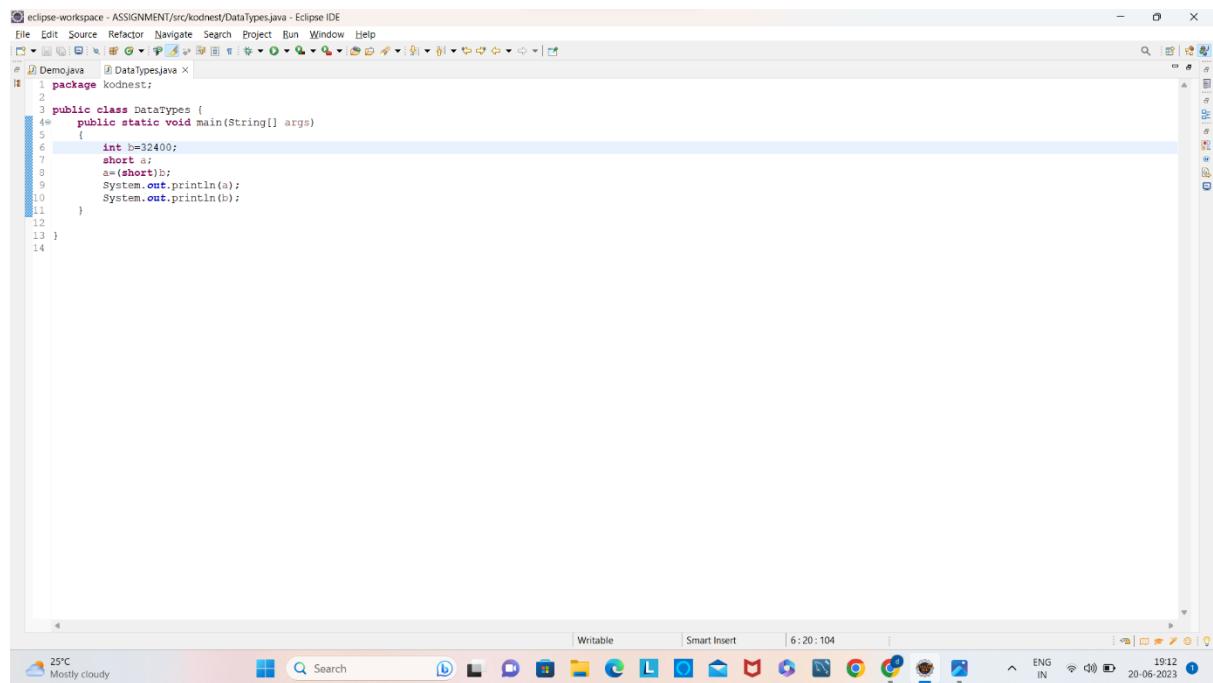
The screenshot shows the Eclipse IDE interface with the `Console` tab selected. The output window displays the results of the `System.out.println` statements from the previous code. The first line shows the value of `a` as `126`, and the second line shows the value of `b` as `126`.

```
126
126
```

Conclusion: int to byte is possible implicitly.

int to short:

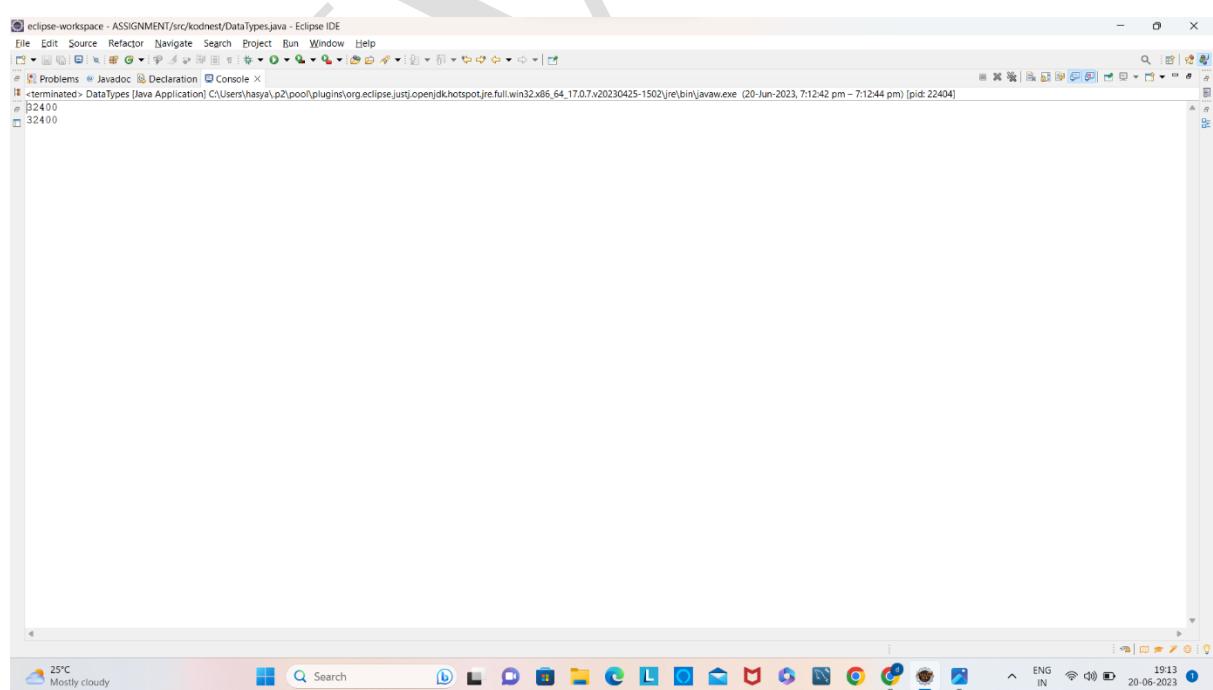
Program:



The screenshot shows the Eclipse IDE interface with a Java file named `DataTypes.java` open. The code defines a class `DataTypes` with a `main` method. In the `main` method, an `int` variable `a` is assigned the value `32400`, and a `short` variable `b` is assigned the same value. Both variables are then printed to the console using `System.out.println`. The code is as follows:

```
1 package kodnest;
2
3 public class DataTypes {
4     public static void main(String[] args) {
5         int a=32400;
6         short b;
7         a=(short)b;
8         System.out.println(a);
9         System.out.println(b);
10    }
11
12 }
13 }
```

Output:



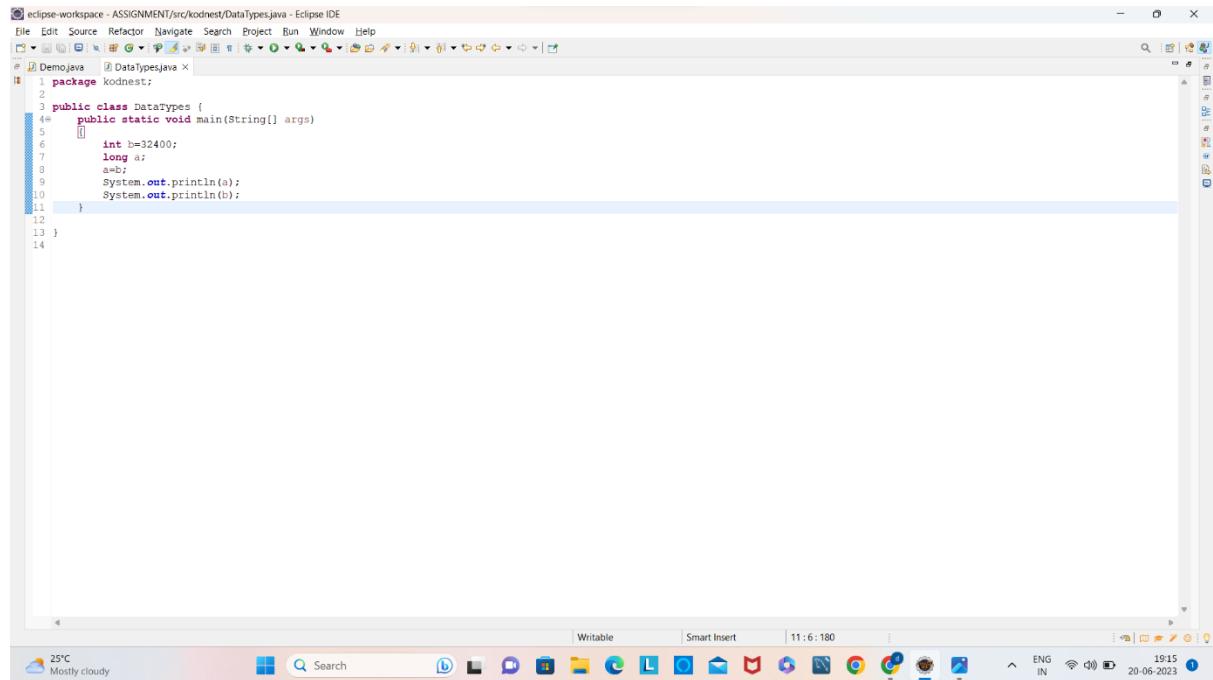
The screenshot shows the Eclipse IDE interface after running the Java application. The `Console` tab is active, displaying the output of the `System.out.println` statements. The output shows the value `32400` for variable `a` and `32400` for variable `b`, indicating that the `int` value was implicitly converted to a `short` when assigned to the `short` variable `b`.

```
2400
32400
```

Conclusion: int to short is possible implicitly.

int to long:

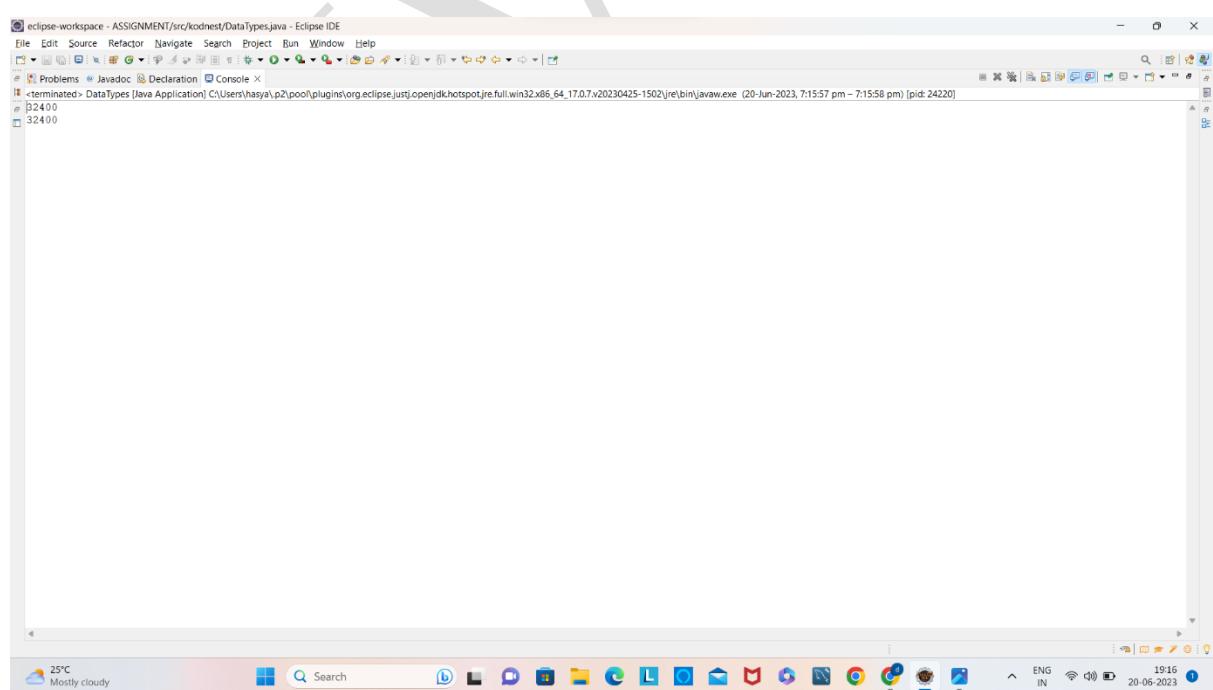
Program:



The screenshot shows the Eclipse IDE interface with a Java file named `DataTypes.java` open. The code defines a class `DataTypes` with a `main` method. Inside the `main` method, there is a line of code where an `int` variable `a` is assigned the value `32400`, and then it is cast to a `long` variable `b`. Both variables are then printed to the console using `System.out.println`.

```
1 package kodnest;
2
3 public class DataTypes {
4     public static void main(String[] args) {
5         int a=32400;
6         long b;
7         a=b;
8         System.out.println(a);
9         System.out.println(b);
10    }
11 }
12
13 }
```

Output:



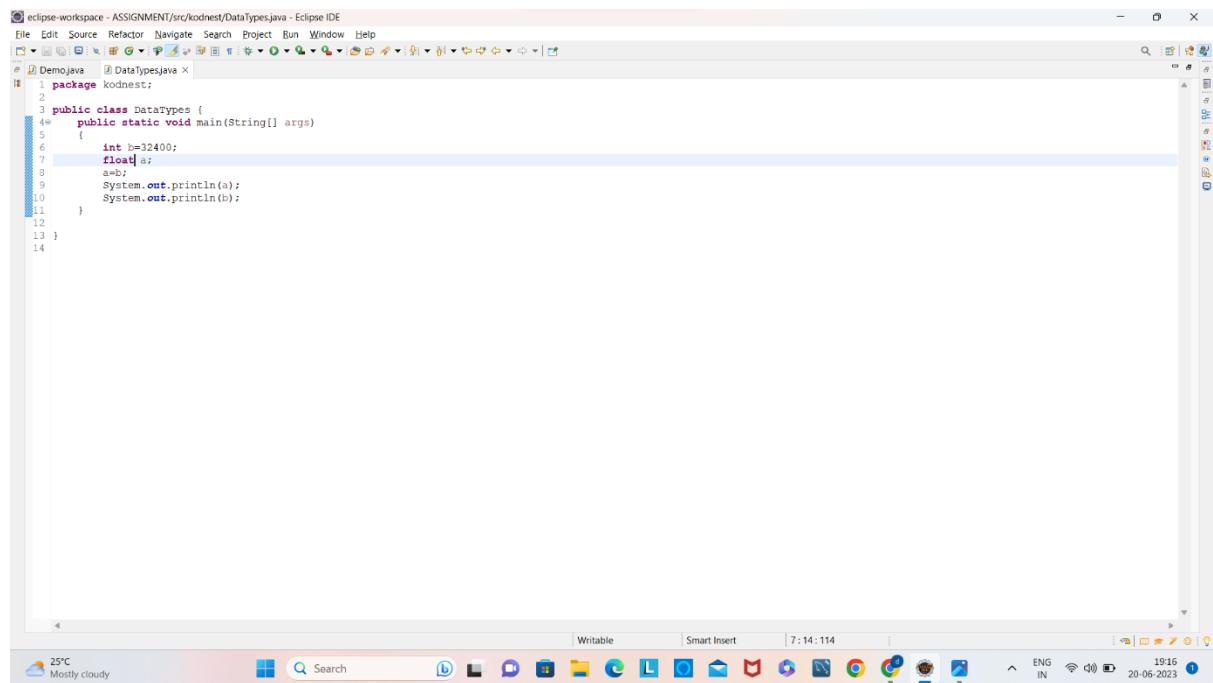
The screenshot shows the Eclipse IDE interface with the `Console` tab selected. The output window displays the results of the `System.out.println` statements. The first line shows the value `32400`, and the second line shows the value `32400` again, demonstrating that the `int` value was implicitly converted to a `long` when assigned to the `long` variable `b`.

```
32400
32400
```

Conclusion: int to long is possible implicitly.

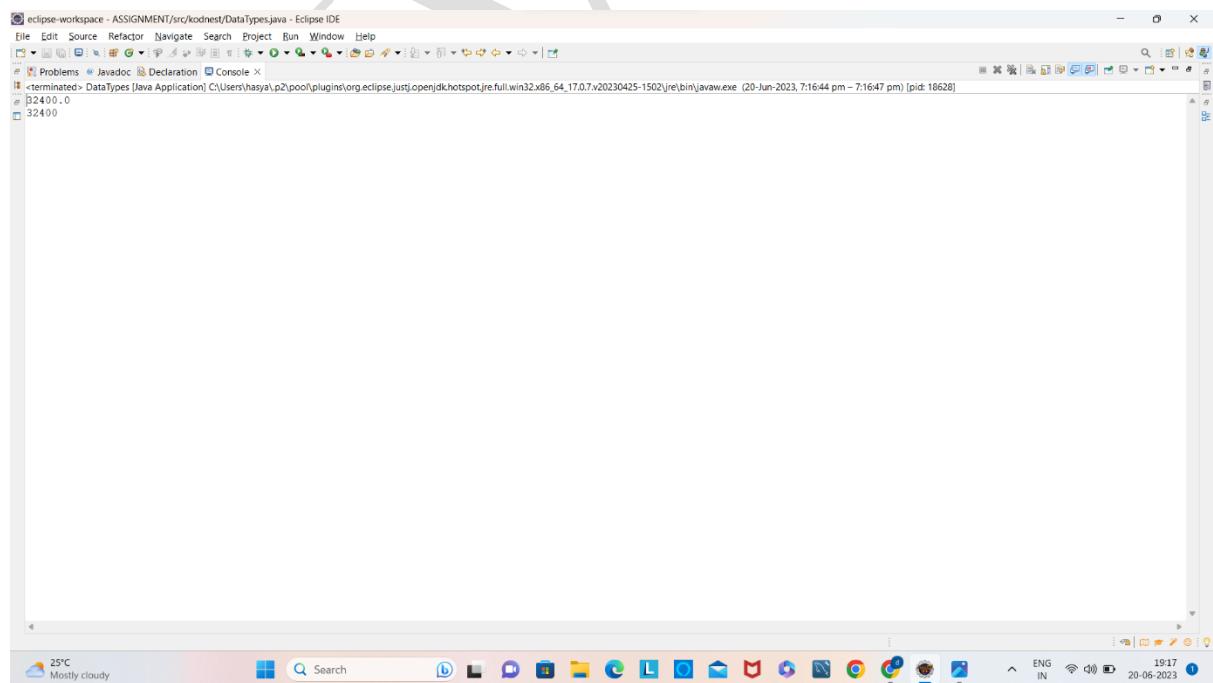
int to float:

Program:



```
eclipse-workspace - ASSIGNMENT/src/kodnest/DataTypes.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Demo.java DataTypes.java
1 package kodnest;
2
3 public class DataTypes {
4     public static void main(String[] args) {
5         int b=32400;
6         float a;
7         a=b;
8         System.out.println(a);
9         System.out.println(b);
10    }
11
12
13 }
```

Output:



```
eclipse-workspace - ASSIGNMENT/src/kodnest/DataTypes.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Problems Javadoc Declaration Console
terminated> DataTypes [Java Application] C:\Users\hasya\p2pool\plugins\org.eclipse.jdt.openjdk.hotspot.jre.full.win32.x86_64_17.0.7.v20230425-1502\jre\bin\javaw.exe (20-Jun-2023, 7:16:44 pm - 7:16:47 pm) [pid: 18628]
32400.0
```

Conclusion: int to float is possible implicitly.

int to double:

Program:



```
eclipse-workspace - ASSIGNMENT/src/kodnest/DataTypes.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Demo.java DataTypes.java
1 package kodnest;
2
3 public class DataTypes {
4     public static void main(String[] args) {
5         int b=3248;
6         double a;
7         a=b;
8         System.out.println(a);
9         System.out.println(b);
10    }
11
12
13 }
```

Output:

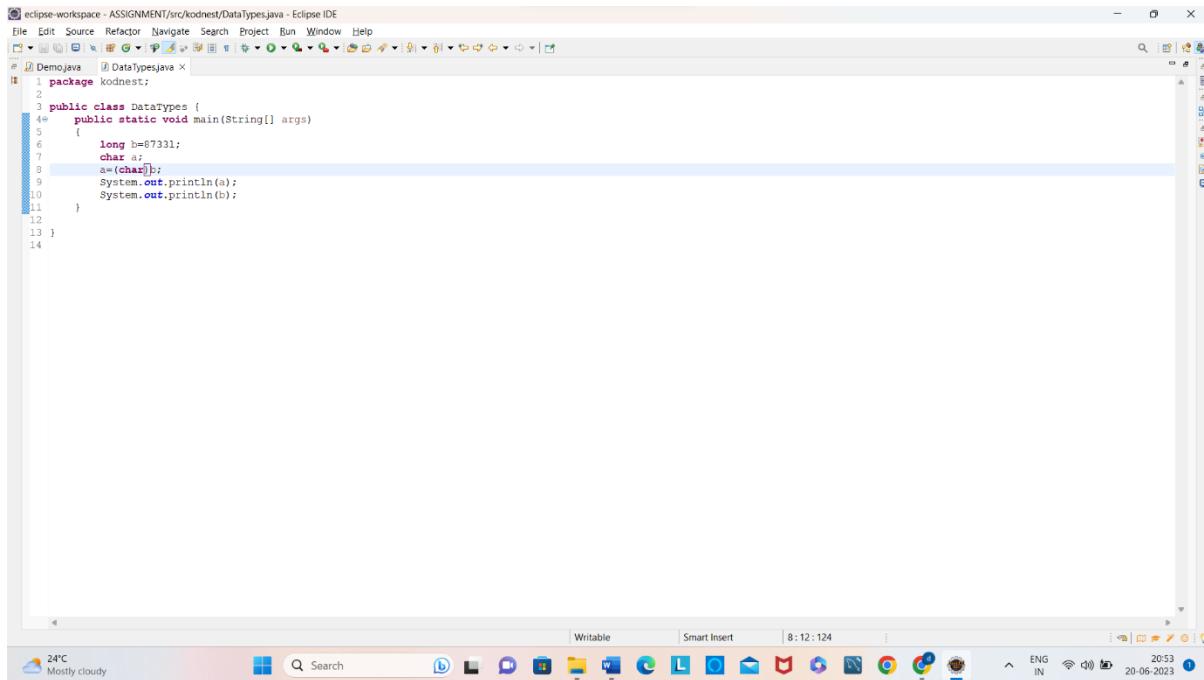


```
eclipse-workspace - ASSIGNMENT/src/kodnest/DataTypes.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Problems Javadoc Declaration Console
terminated> DataTypes [Java Application] C:\Users\hasya\p2pool\plugins\org.eclipse.jdt.openjdk.hotspot.jre.full.win32.x86_64_17.0.7.v20230425-1502\jre\bin\javaw.exe (20-Jun-2023, 7:19:25 pm - 7:19:27 pm) [pid: 996]
32489.0
32489
```

Conclusion: int to double is possible implicitly.

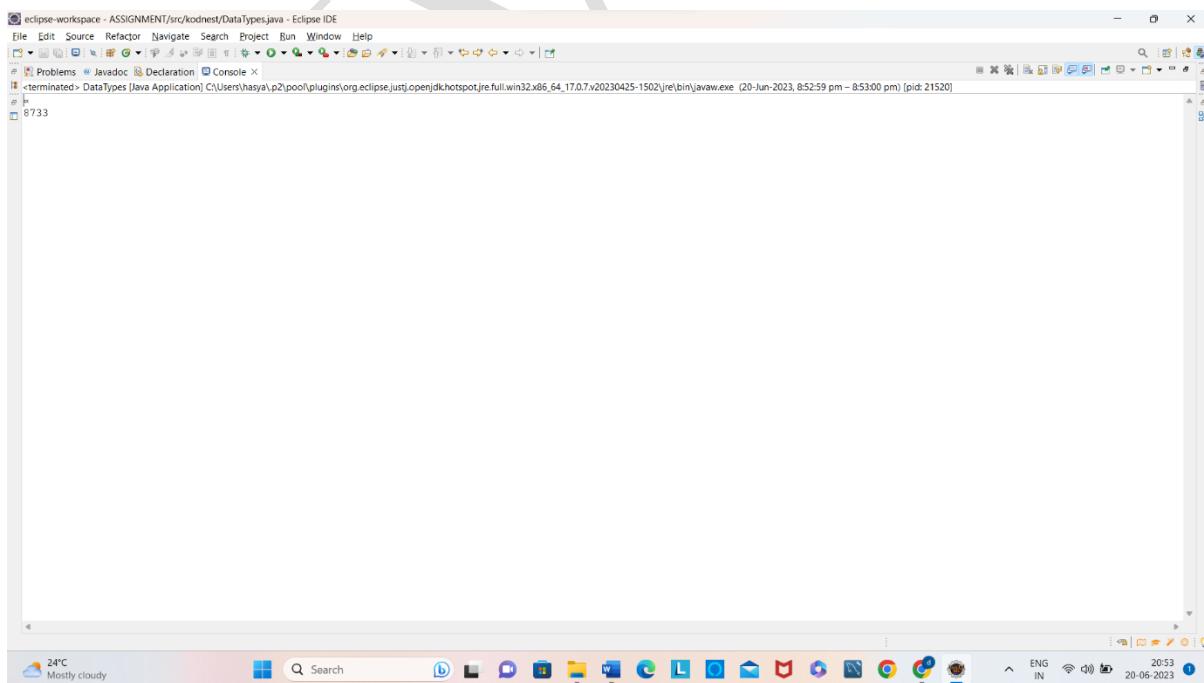
long to char:

Program:



```
eclipse-workspace - ASSIGNMENT/src/kodnest/DataTypes.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Demo.java DataTypes.java
1 package kodnest;
2
3 public class DataTypes {
4     public static void main(String[] args) {
5         long b=67331;
6         char a;
7         a=(char)b;
8         System.out.println(a);
9         System.out.println(b);
10    }
11
12
13 }
```

Output:

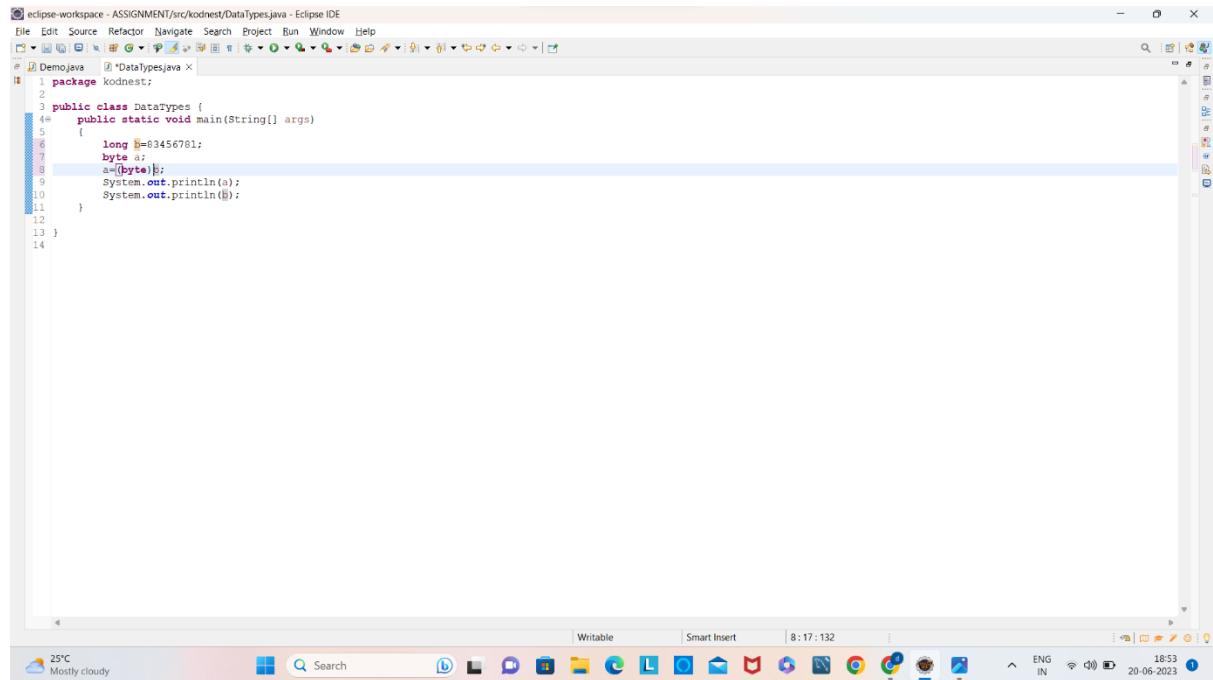


```
eclipse-workspace - ASSIGNMENT/src/kodnest/DataTypes.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Problems Javadoc Declaration Console
terminated> DataTypes [Java Application] C:\Users\hasya\p2pool\plugins\org.eclipse.jdt.openjdk.hotspot.jre.full.win32.x86_64_17.0.7.v20230425-1502\jre\bin\javaw.exe (20-Jun-2023, 8:52:59 pm - 8:53:00 pm) [pid: 21520]
8733
```

Conclusion: long to char is possible explicitly.

long to byte:

Program:

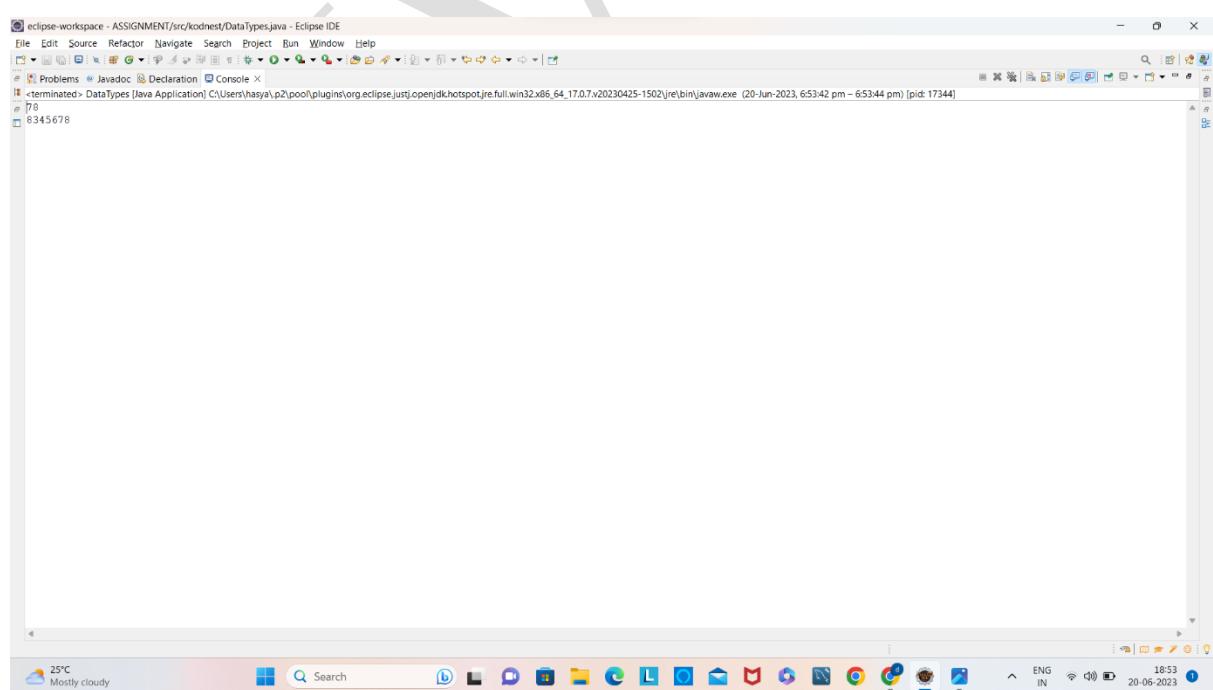


The screenshot shows the Eclipse IDE interface with the following code in the editor:

```
1 package kodnest;
2
3 public class DataTypes {
4     public static void main(String[] args) {
5         long l=63456781;
6         byte a;
7         a=(byte)l;
8         System.out.println(a);
9         System.out.println(l);
10    }
11
12 }
13 }
```

The code demonstrates casting a long value to a byte and printing both values to the console.

Output:



The screenshot shows the Eclipse IDE interface after running the Java application. The output window displays the following:

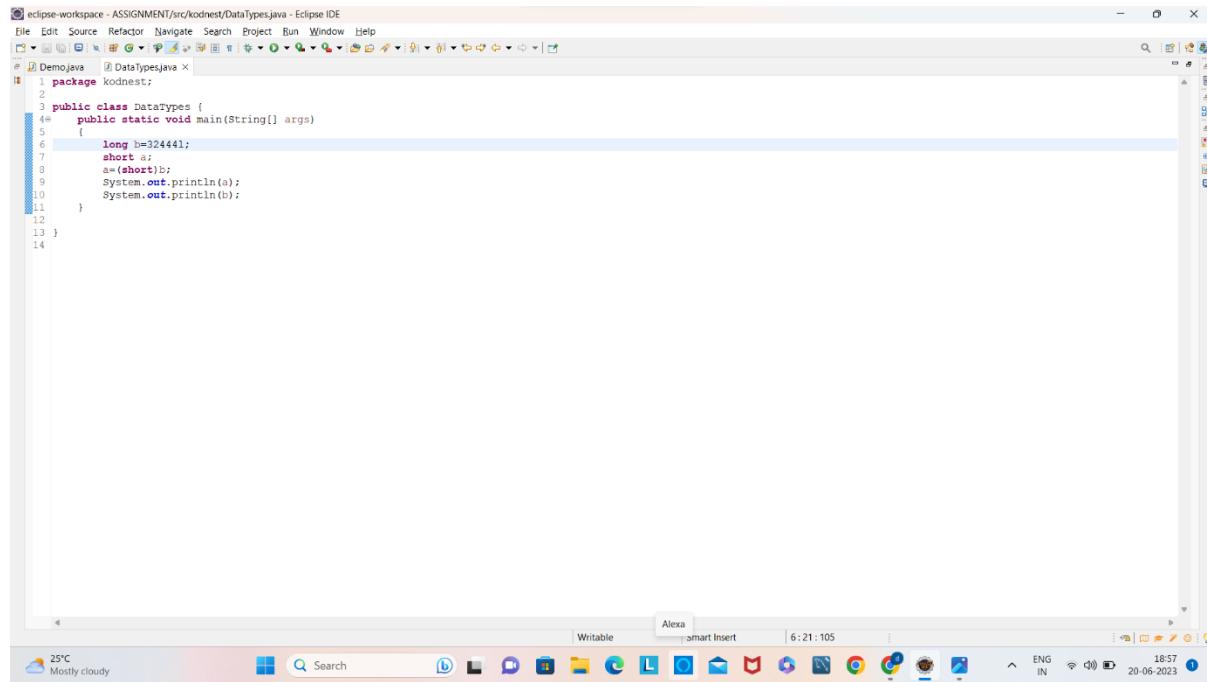
```
8345678
```

The output shows the byte value 8345678, which is the result of casting the long value 63456781 to a byte.

Conclusion: long to byte is possible explicitly.

long to short:

Program:

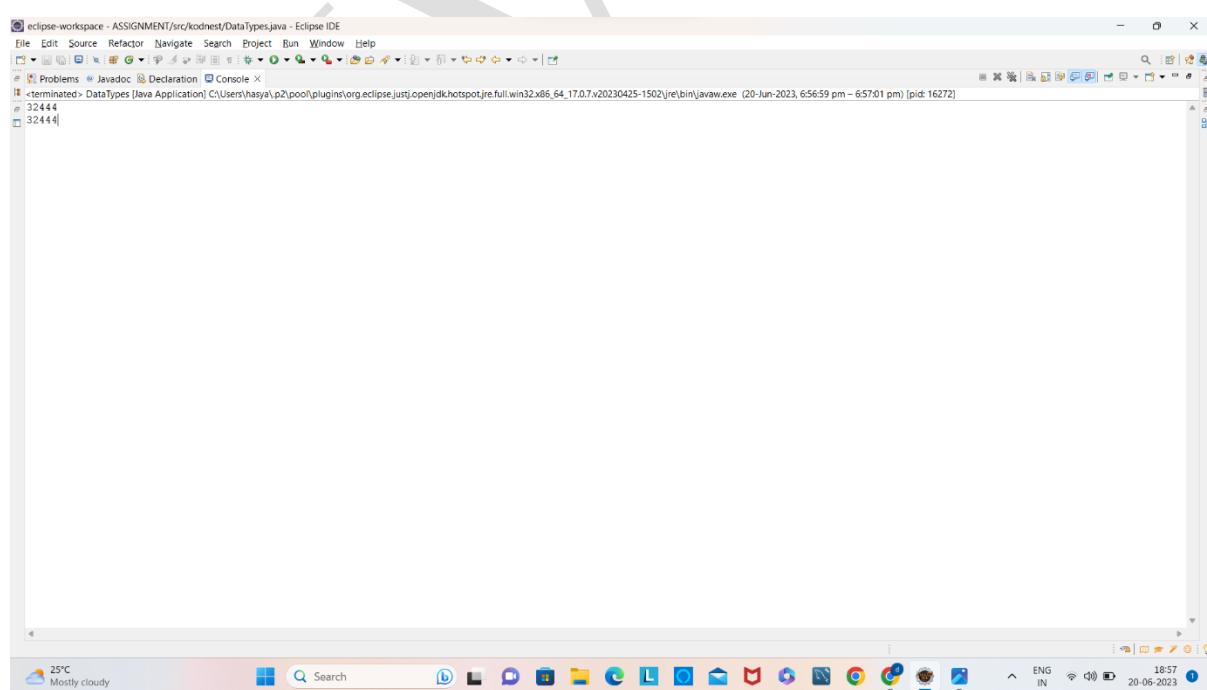


The screenshot shows the Eclipse IDE interface. The top menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. Below the menu is a toolbar with various icons. The left sidebar shows a project structure with 'Demo.java' and 'DataTypes.java'. The main editor window contains the following Java code:

```
1 package kodnest;
2
3 public class DataTypes {
4     public static void main(String[] args) {
5         long b=32444L;
6         short a;
7         a=(short)b;
8         System.out.println(a);
9         System.out.println(b);
10    }
11
12 }
13
14
```

The code converts a long value to a short value and prints both to the console.

Output:



The screenshot shows the Eclipse IDE interface after running the Java application. The top menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. Below the menu is a toolbar with various icons. The left sidebar shows a project structure with 'Problems', 'JavaView', 'Declaration', and 'Console'. The 'Console' tab is selected, showing the output of the application's execution. The output text is:

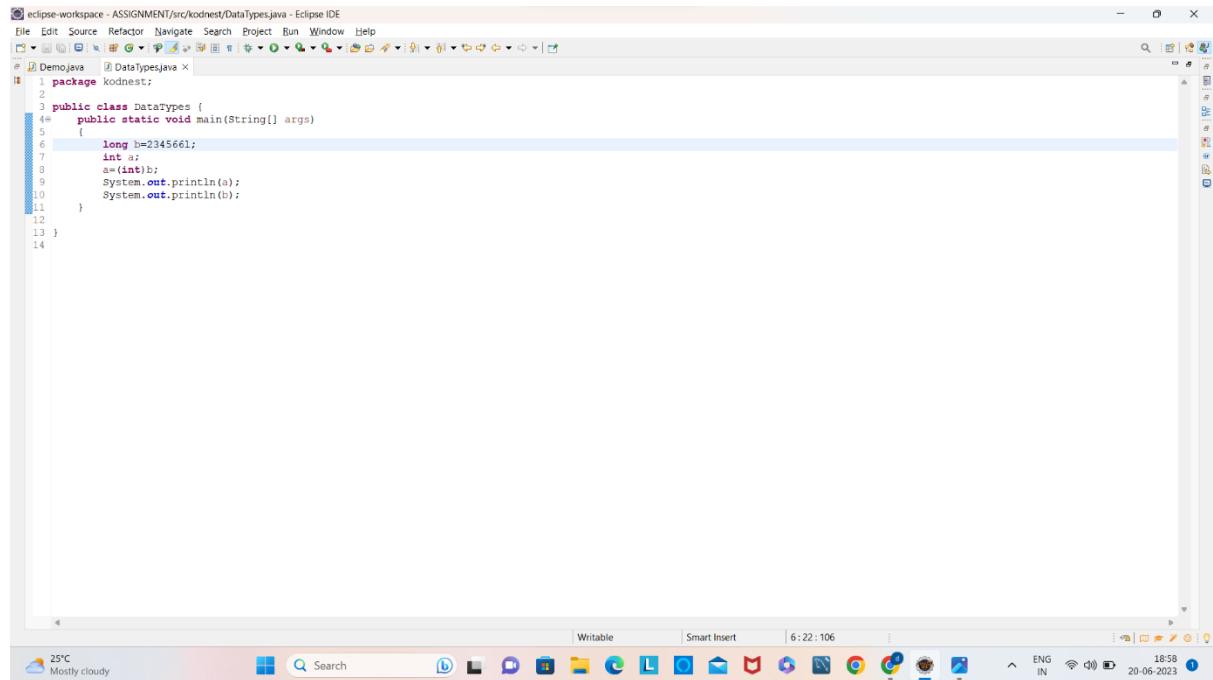
```
32444
32444
```

The application successfully converted the long value 32444L to a short value 32444 and printed both to the console.

Conclusion: long to short is possible explicitly.

long to int:

Program:

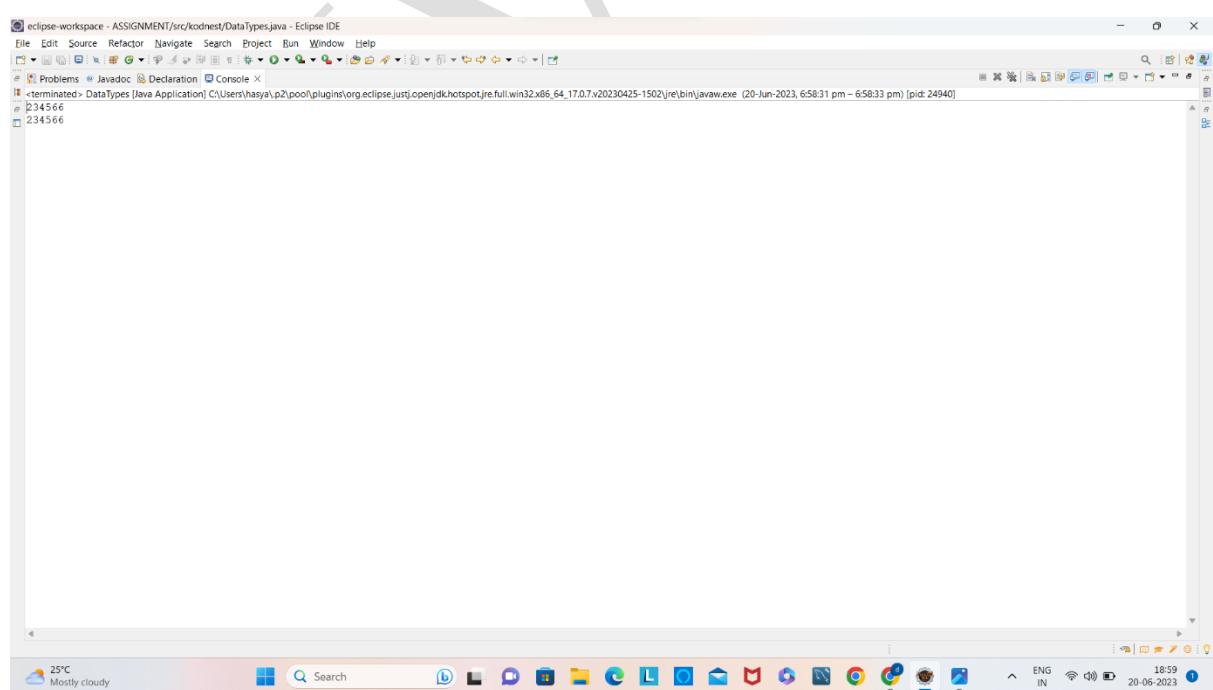


The screenshot shows the Eclipse IDE interface with the following code in the editor:

```
1 package kodnest;
2
3 public class DataTypes {
4     public static void main(String[] args) {
5         long b=2345661;
6         int a;
7         a=(int)b;
8         System.out.println(a);
9         System.out.println(b);
10    }
11
12 }
13 }
```

The code demonstrates casting a long value to an int and printing both values to the console.

Output:



The screenshot shows the Eclipse IDE interface with the following output in the Console view:

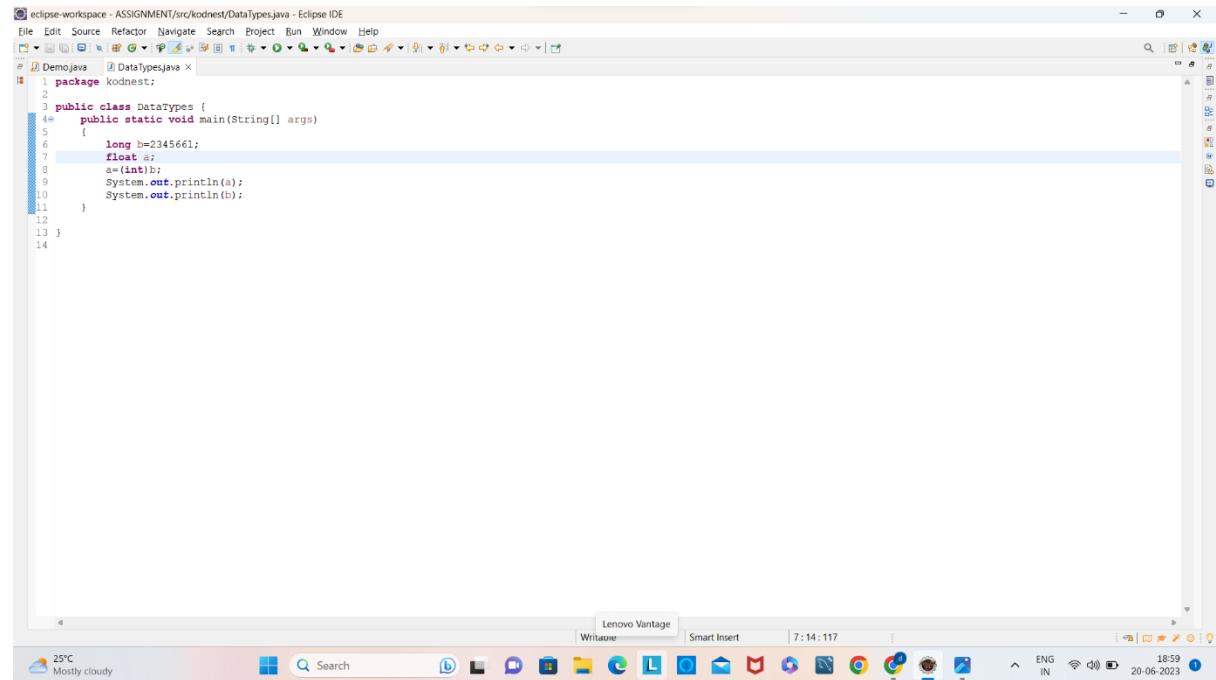
```
234566
234566
```

The output shows the original long value and the converted int value, both printed to the console.

Conclusion: long to int is possible explicitly.

long to float:

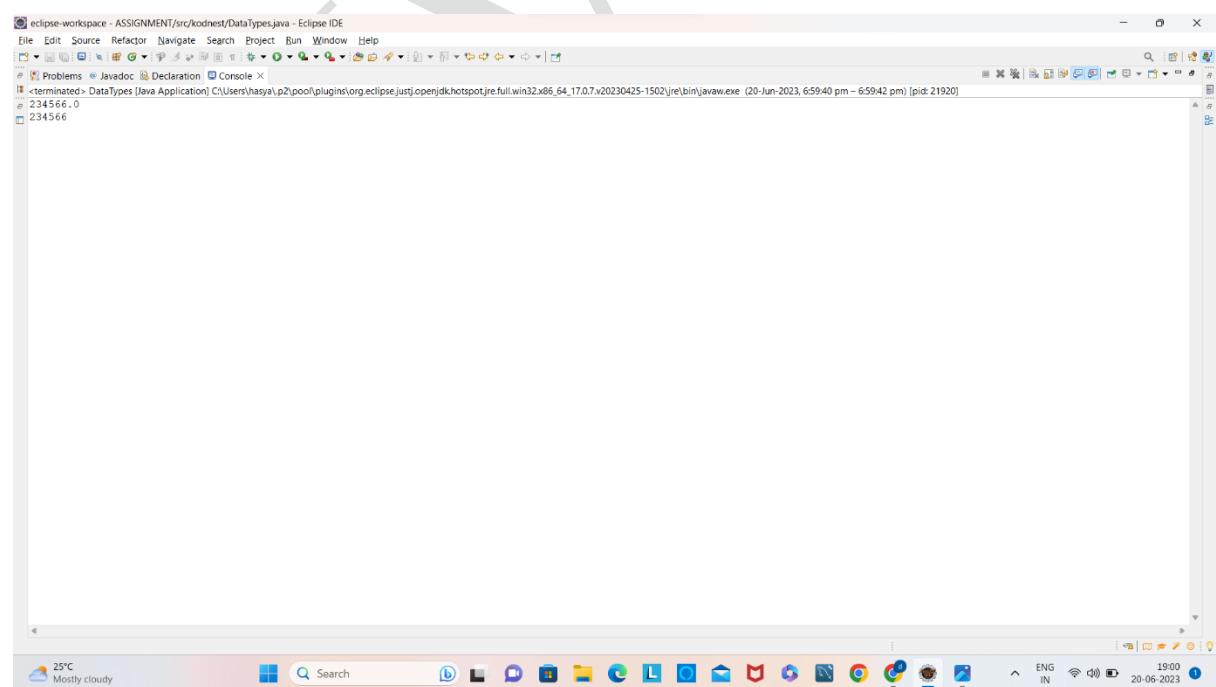
Program:



The screenshot shows the Eclipse IDE interface with a Java file named `DataTypes.java` open. The code defines a class `DataTypes` with a `main` method. Inside the `main` method, a long variable `b` is assigned the value `2345661`. This value is then cast to a float variable `a` using the `(float)` cast operator. Finally, both variables are printed to the console using `System.out.println`.

```
1 package kodnest;
2
3 public class DataTypes {
4     public static void main(String[] args) {
5         long b=2345661;
6         float a=(float)b;
7         System.out.println(a);
8         System.out.println(b);
9     }
10 }
11
12
13 }
```

Output:



The screenshot shows the Eclipse IDE interface after running the Java application. The `Console` tab is active, displaying the output of the `System.out.println` statements. The output shows the value `234566.0`, which is the result of the explicit type conversion of the long value to a float.

```
234566.0
234566
```

Conclusion: long to float is possible explicitly.

long to double:

Program:



```
eclipse-workspace - ASSIGNMENT/src/kodnest/DataTypes.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Demo.java DataTypes.java
1 package kodnest;
2
3 public class DataTypes {
4     public static void main(String[] args) {
5         long b=12345l;
6         double a;
7         a=(double)b;
8         System.out.println(a);
9         System.out.println(b);
10    }
11
12
13 }
14
```

Output:

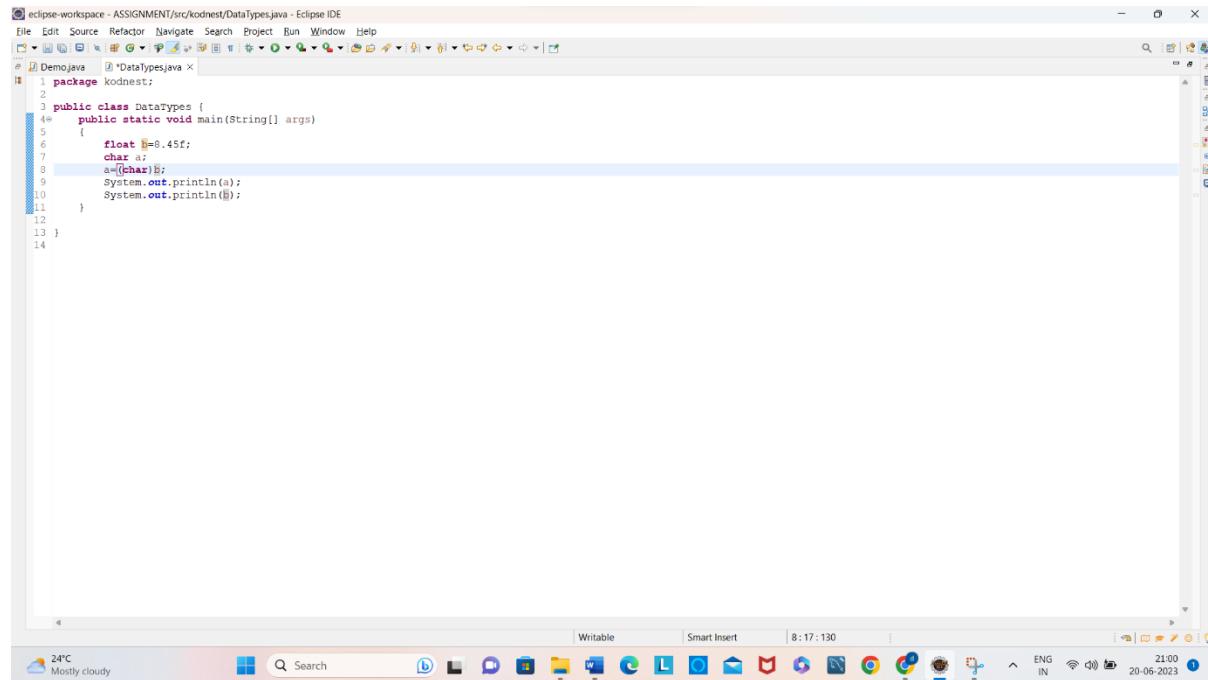


```
eclipse-workspace - ASSIGNMENT/src/kodnest/DataTypes.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Problems Javadoc Declaration Console
terminated> DataTypes [Java Application] C:\Users\hasya\p2pool\plugins\org.eclipse.jdt.openjdk.hotspot.jre.full.win32.x86_64_17.0.7.v20230425-1502\jre\bin\javaw.exe (20-Jun-2023, 7:05:30 pm - 7:05:32 pm) [pid: 11660]
12345.0
12345
```

Conclusion: long to double is possible explicitly.

float to char:

Program:

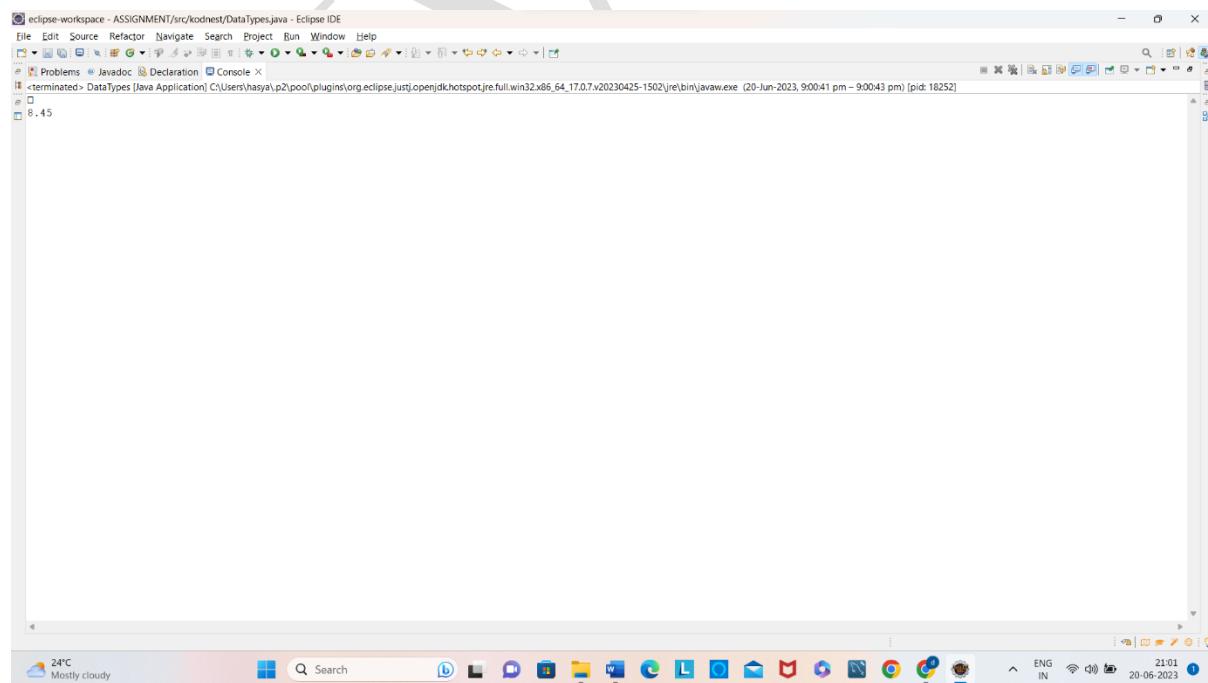


The screenshot shows the Eclipse IDE interface with the following code in the editor:

```
1 package kodnest;
2
3 public class DataTypes {
4     public static void main(String[] args) {
5         float f=8.45f;
6         char a;
7         a=(char)f;
8         System.out.println(a);
9         System.out.println(f);
10    }
11
12 }
13 }
```

The code defines a class named `DataTypes` with a `main` method. It declares a `float` variable `f` with the value `8.45f`, converts it to a `char` variable `a` using casting, and then prints both the character `a` and the float value `f` to the console.

Output:



The screenshot shows the Eclipse IDE interface after running the program. The output window displays the results:

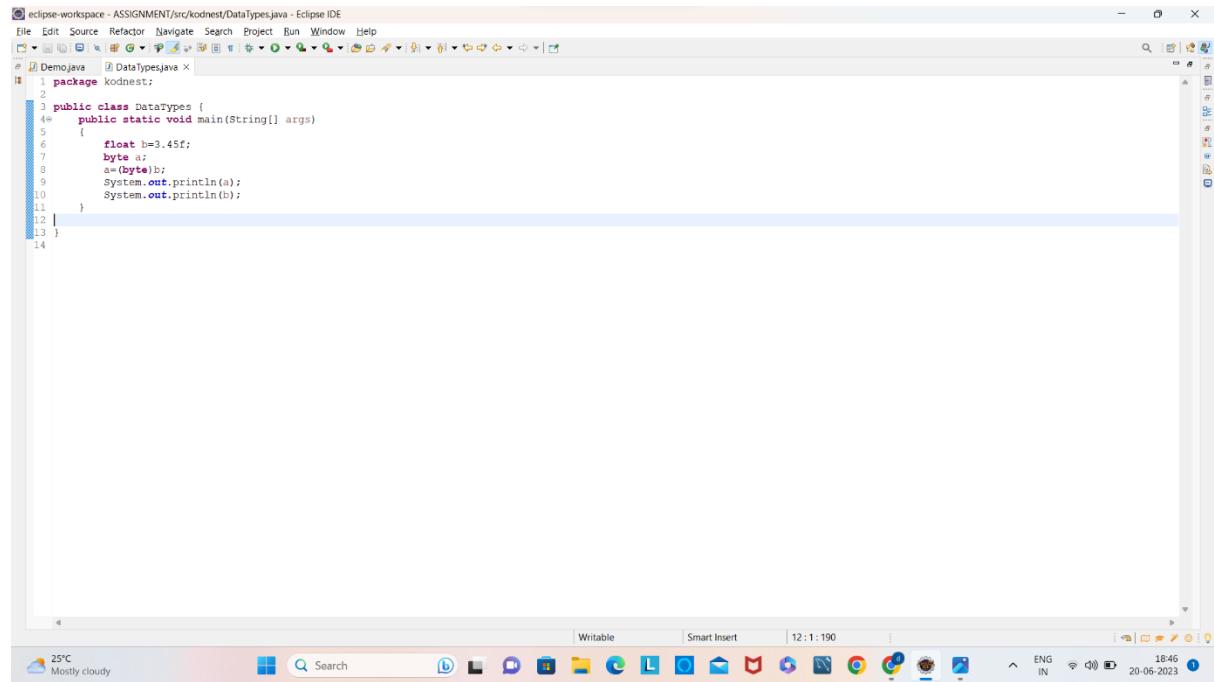
```
8.45
```

The output shows the float value `8.45` printed to the console.

Conclusion: float to char is possible explicitly.

float to byte:

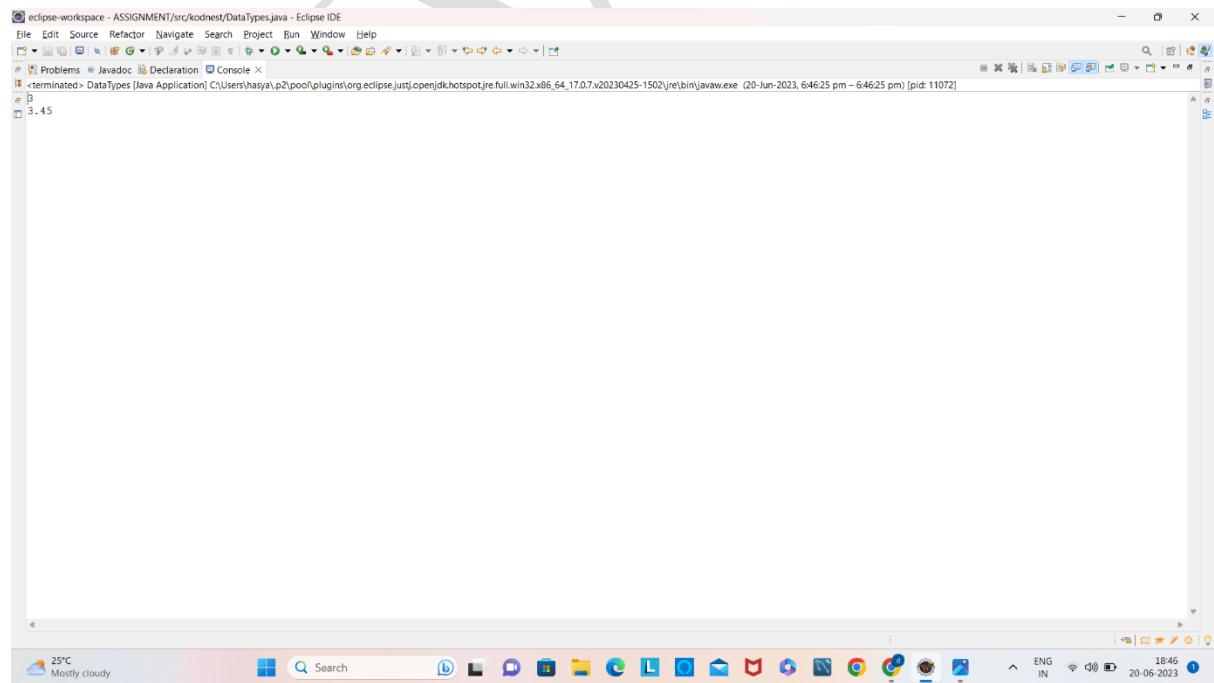
Program:



The screenshot shows the Eclipse IDE interface with a Java file named `DataTypes.java` open. The code defines a class `DataTypes` with a `main` method. Inside the `main` method, a `float` variable `b` is assigned the value `3.45f`. A `byte` variable `a` is then assigned the value of `b` using the `(byte)` cast operator. Finally, both variables are printed to the console using `System.out.println`.

```
1 package kodnest;
2
3 public class DataTypes {
4     public static void main(String[] args) {
5         float b=3.45f;
6         byte a;
7         a=(byte)b;
8         System.out.println(a);
9         System.out.println(b);
10    }
11 }
12
13 }
```

Output:



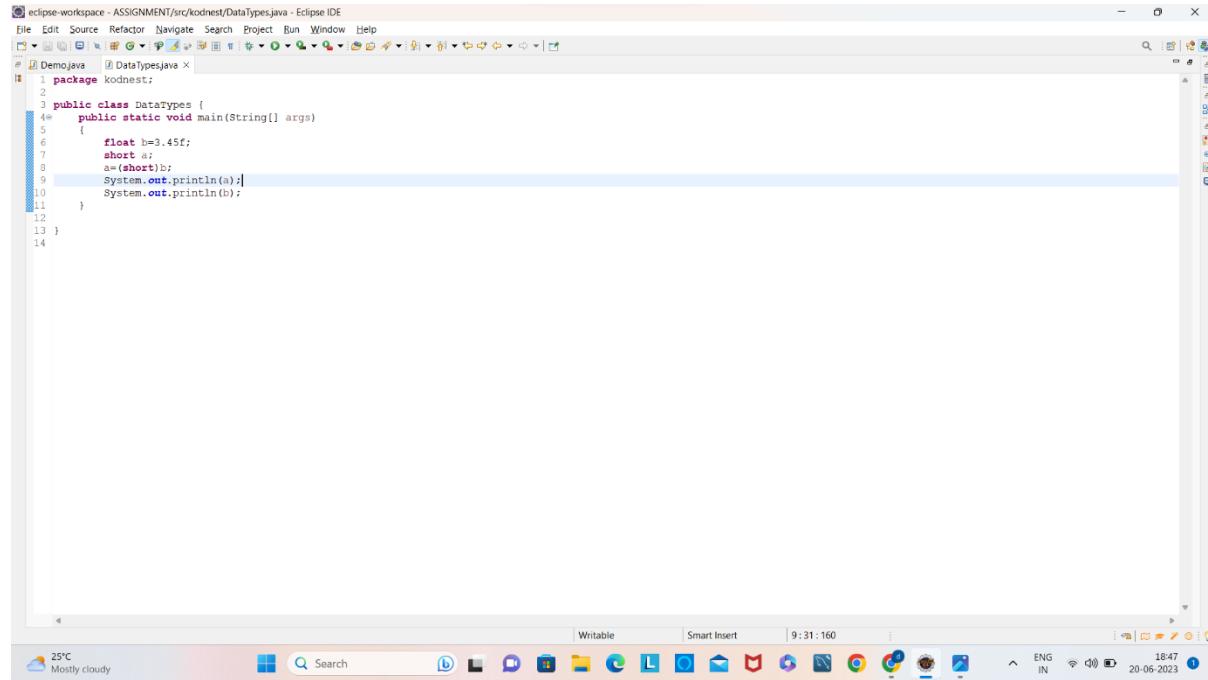
The screenshot shows the Eclipse IDE interface after running the Java application. The output window displays the results of the `System.out.println` statements. The first line shows the value of variable `a` as `3`, and the second line shows the value of variable `b` as `3.45`.

```
3
3.45
```

Conclusion: float to byte is possible explicitly.

float to short:

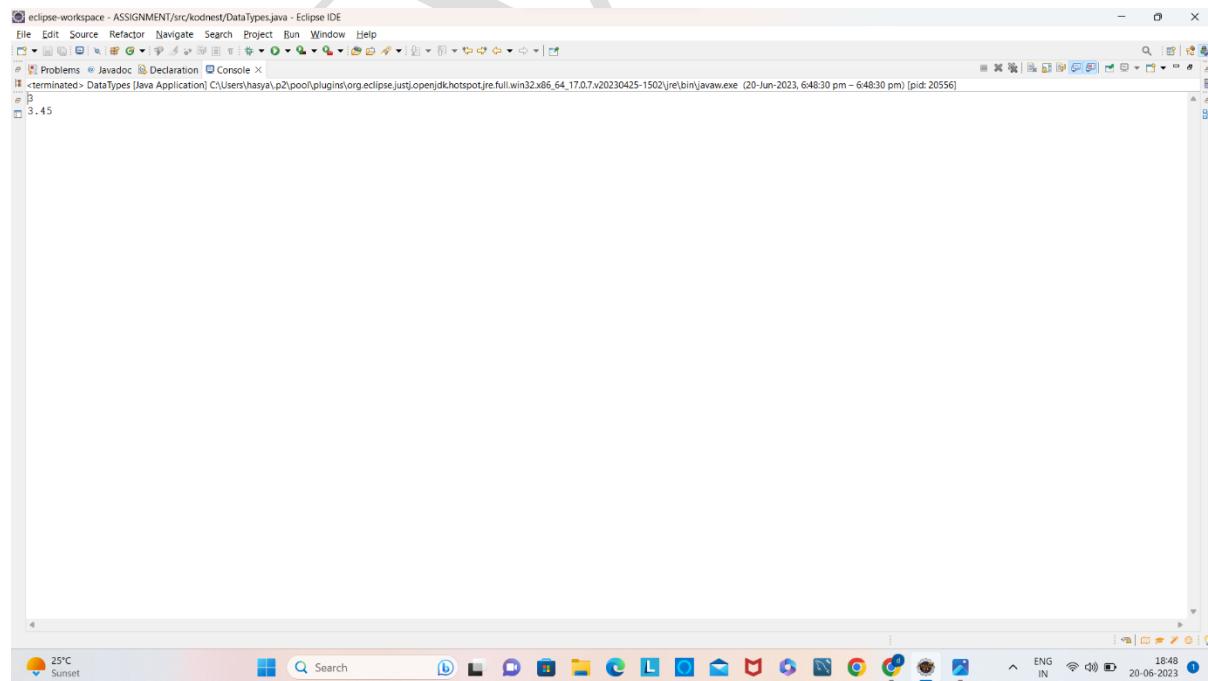
Program:



The screenshot shows the Eclipse IDE interface with a Java file named `DataTypes.java` open. The code defines a class `DataTypes` with a `main` method. Inside the `main` method, a `float` variable `b` is assigned the value `3.45f`. This value is then cast to a `short` type and stored in a variable `a`. Finally, both `a` and `b` are printed to the console using `System.out.println`.

```
1 package kodnest;
2
3 public class DataTypes {
4     public static void main(String[] args) {
5         float b=3.45f;
6         short a;
7         a=(short)b;
8         System.out.println(a);
9         System.out.println(b);
10    }
11 }
12
13 }
```

Output:



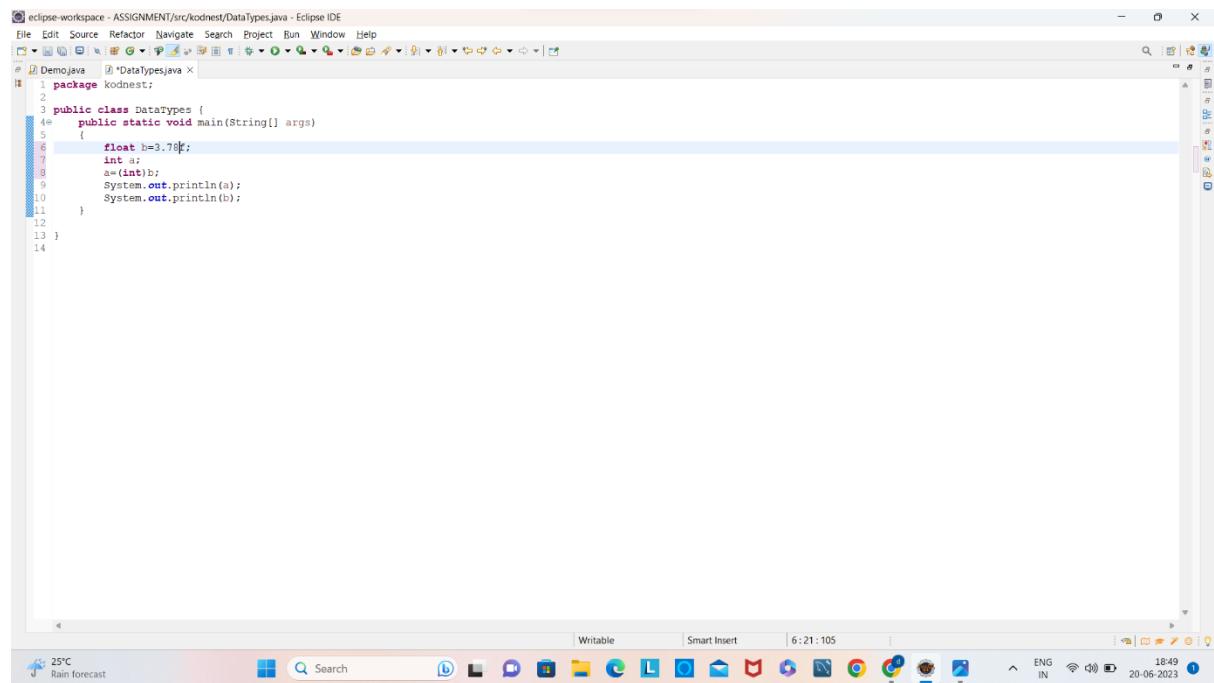
The screenshot shows the Eclipse IDE interface after running the Java application. The `Console` tab is active, displaying the output of the `System.out.println` statements. The output shows the value `3.45`, which is the result of the explicit cast from `float` to `short`.

```
3.45
```

Conclusion: float to short is possible explicitly.

float to int:

Program:

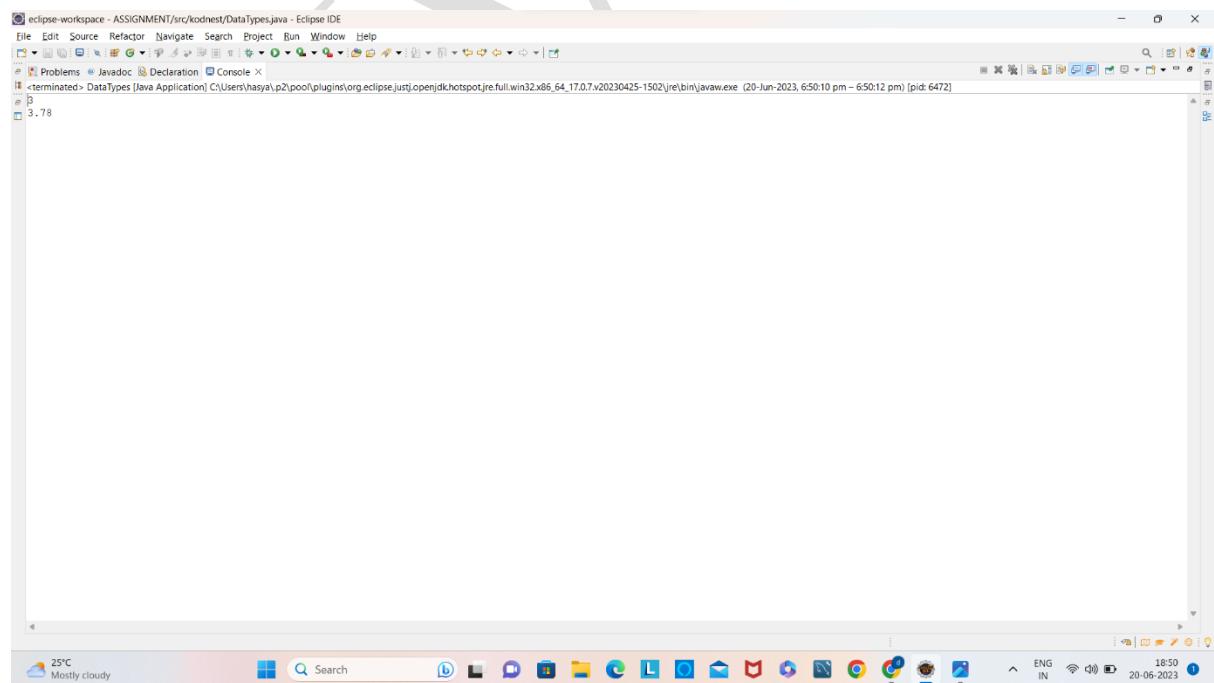


The screenshot shows the Eclipse IDE interface with the following Java code in the editor:

```
1 package kodnest;
2
3 public class DataTypes {
4     public static void main(String[] args) {
5         float b=3.78f;
6         int a;
7         a=(int)b;
8         System.out.println(a);
9         System.out.println(b);
10    }
11
12 }
13 }
```

The code defines a class named `DataTypes` with a `main` method. It declares a `float` variable `b` with a value of `3.78f`. It then converts `b` to an `int` and stores it in `a`. Finally, it prints both `a` and `b` using `System.out.println`.

Output:



The screenshot shows the Eclipse IDE interface after running the program. The output is displayed in the Console tab:

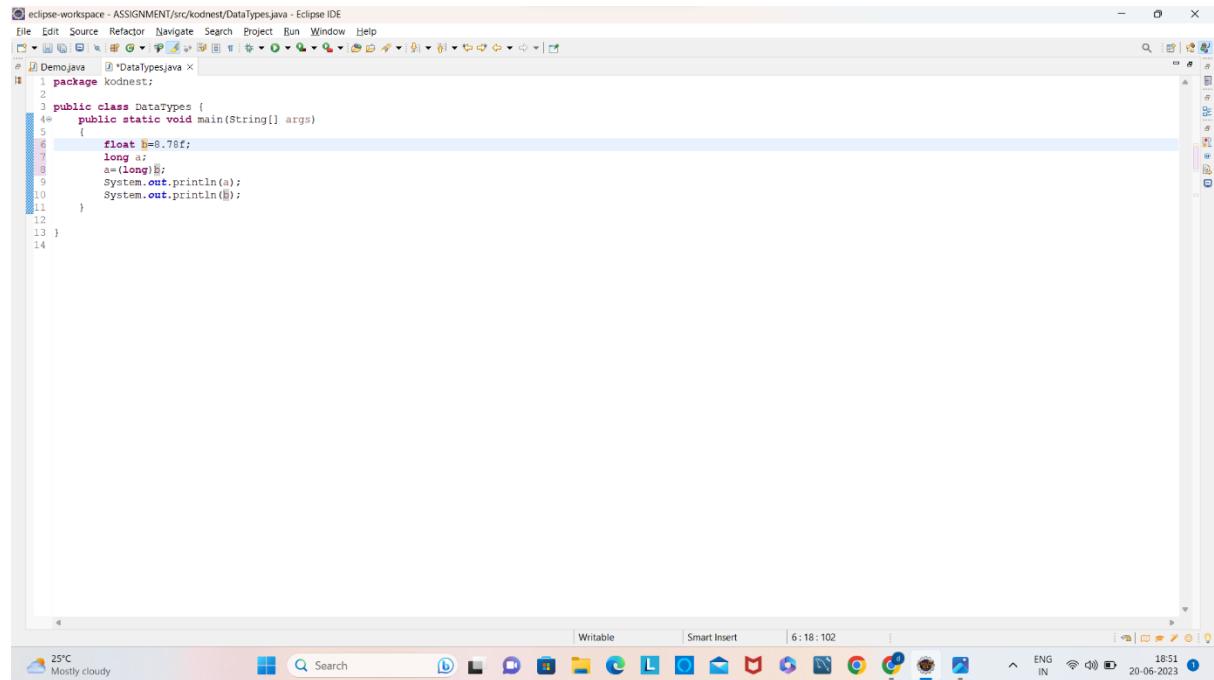
```
terminated> DataTypes [Java Application] C:\Users\hasya\p2pool\plugins\org.eclipse.jdt.openjdk.hotspot.jre.full.win32.x86_64_17.0.7.v20230425-1502\jre\bin\javaw.exe (20-Jun-2023, 6:50:10 pm - 6:50:12 pm) [pid: 6472]
3
3.78
```

The output shows the value `3` followed by `3.78`, indicating that the float value was converted to an integer explicitly.

Conclusion: float to int is possible explicitly.

float to long:

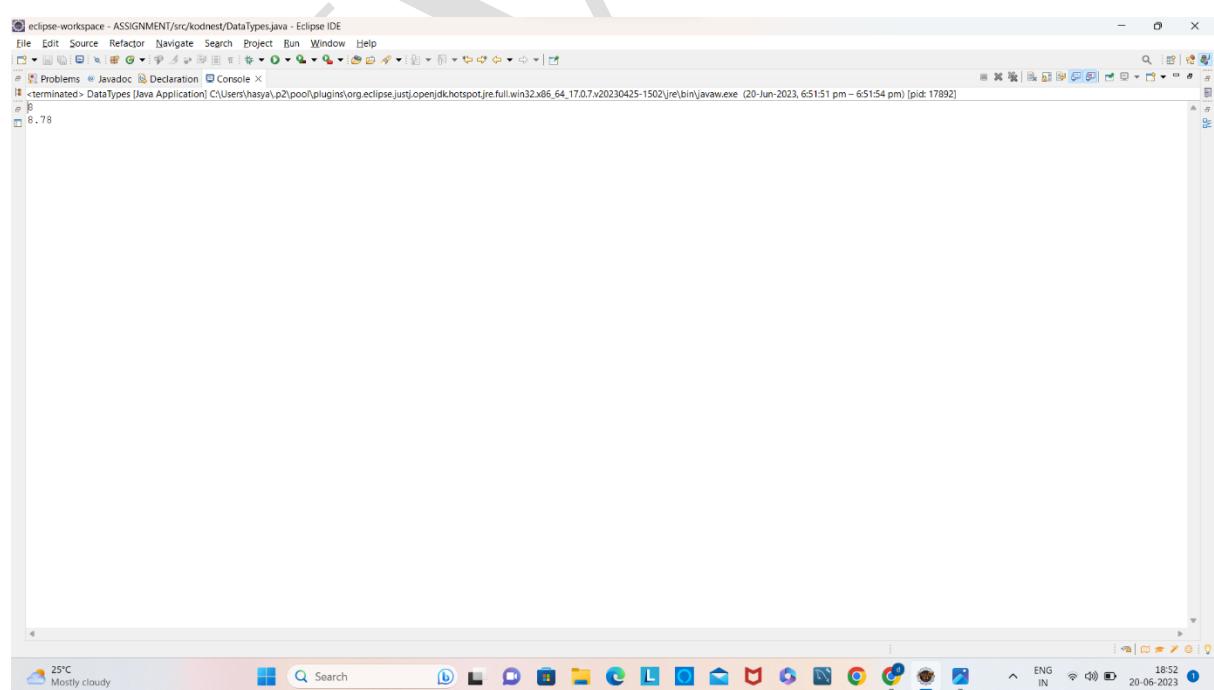
Program:



The screenshot shows the Eclipse IDE interface with a Java file named 'DataTypes.java' open. The code defines a class 'DataTypes' with a main method. Inside the main method, a float variable 'a' is assigned the value 8.78f. This variable is then explicitly cast to a long type using the (long) operator. Finally, the value is printed to the console using System.out.println. The Eclipse interface includes toolbars, a menu bar, and various editor panes.

```
1 package kodnest;
2
3 public class DataTypes {
4     public static void main(String[] args) {
5         float a=8.78f;
6         long b;
7         a=(long)b;
8         System.out.println(a);
9         System.out.println(b);
10    }
11
12 }
13 }
```

Output:



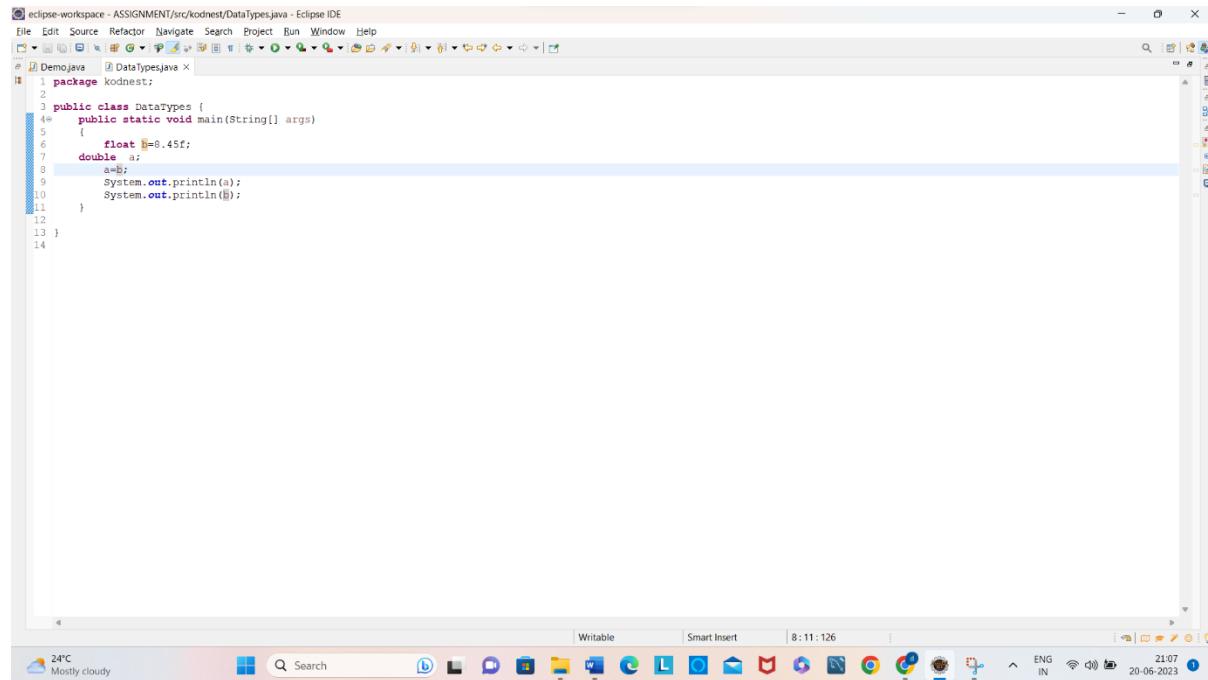
The screenshot shows the Eclipse IDE interface after running the Java application. The 'Console' tab is active, displaying the output of the program. The output shows the value 8.78 being printed to the console. The Eclipse interface includes toolbars, a menu bar, and various editor panes.

```
8.78
```

Conclusion: float to long is possible explicitly.

float to double:

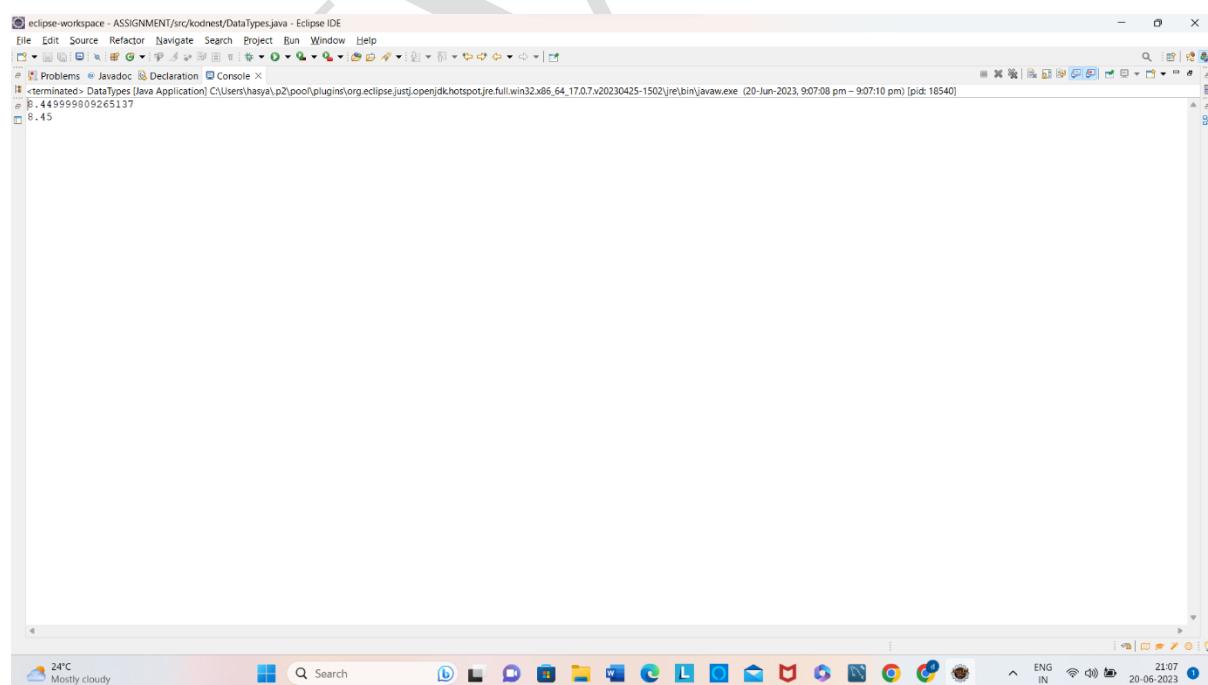
Program:



The screenshot shows the Eclipse IDE interface with a Java file named `DataTypes.java` open. The code defines a class `DataTypes` with a `main` method. Inside the `main` method, a `float` variable `a` is assigned the value `8.45f`. A `double` variable `b` is then assigned the value of `a` using the assignment operator `=`. Finally, both `a` and `b` are printed to the console using `System.out.println`.

```
1 package kodnest;
2
3 public class DataTypes {
4     public static void main(String[] args) {
5         float a=8.45f;
6         double b=a;
7         System.out.println(a);
8         System.out.println(b);
9     }
10 }
11
12
13 }
```

Output:



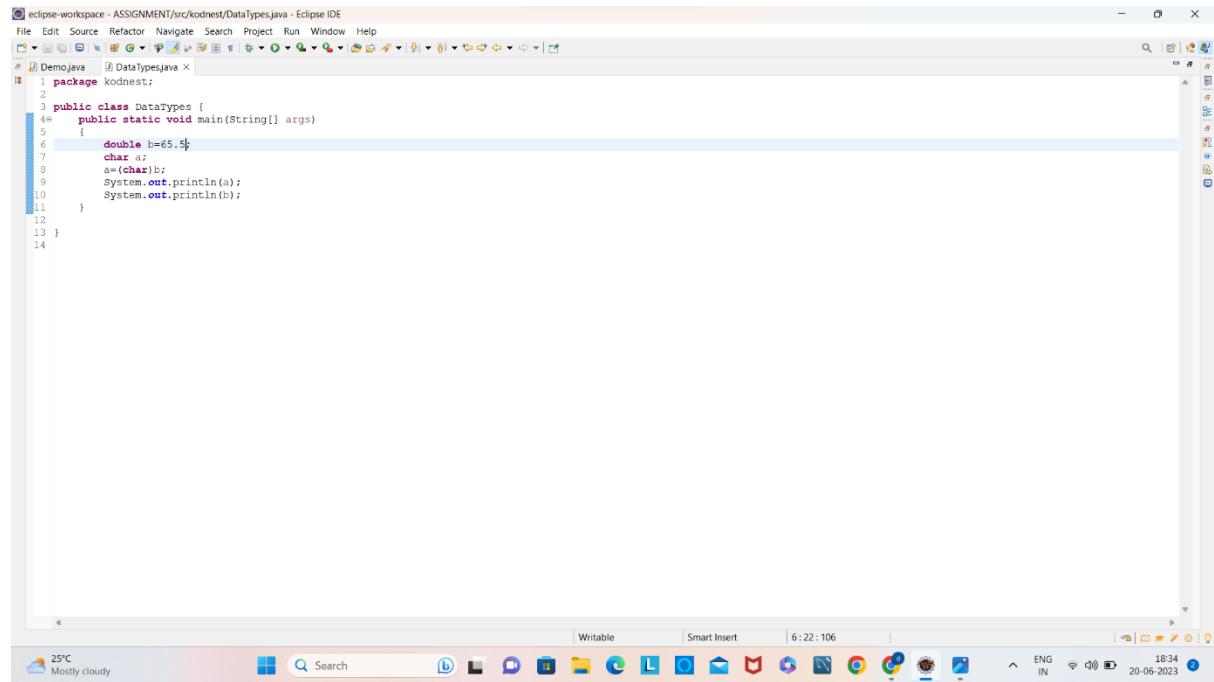
The screenshot shows the Eclipse IDE interface with the `Console` tab selected. The output window displays the results of the `System.out.println` statements from the previous code. The first line shows the variable `a` with its float value `8.45`. The second line shows the variable `b` with its double value `8.45`, demonstrating that the float value was implicitly converted to a double when assigned to the double variable.

```
8.45
8.45
```

Conclusion: float to double is possible implicitly.

double to char:

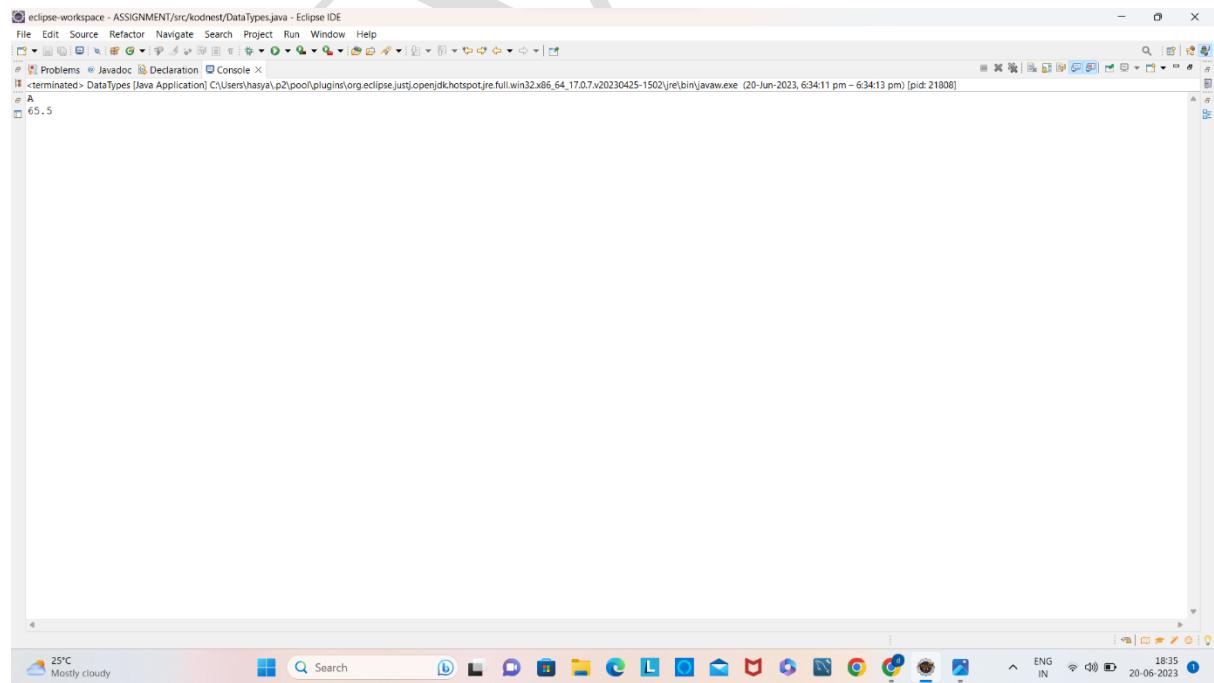
Program:



The screenshot shows the Eclipse IDE interface with a Java file named `DataTypes.java` open. The code defines a class `DataTypes` with a `main` method. Inside the `main` method, a double variable `b` is assigned the value `65.5`. A character variable `a` is then assigned the value of `b` using the assignment operator `=`. Finally, both `a` and `b` are printed to the console using `System.out.println`.

```
1 package kodnest;
2
3 public class DataTypes {
4     public static void main(String[] args) {
5         double b=65.5;
6         char a;
7         a=(char)b;
8         System.out.println(a);
9         System.out.println(b);
10    }
11 }
12
13 }
```

Output:



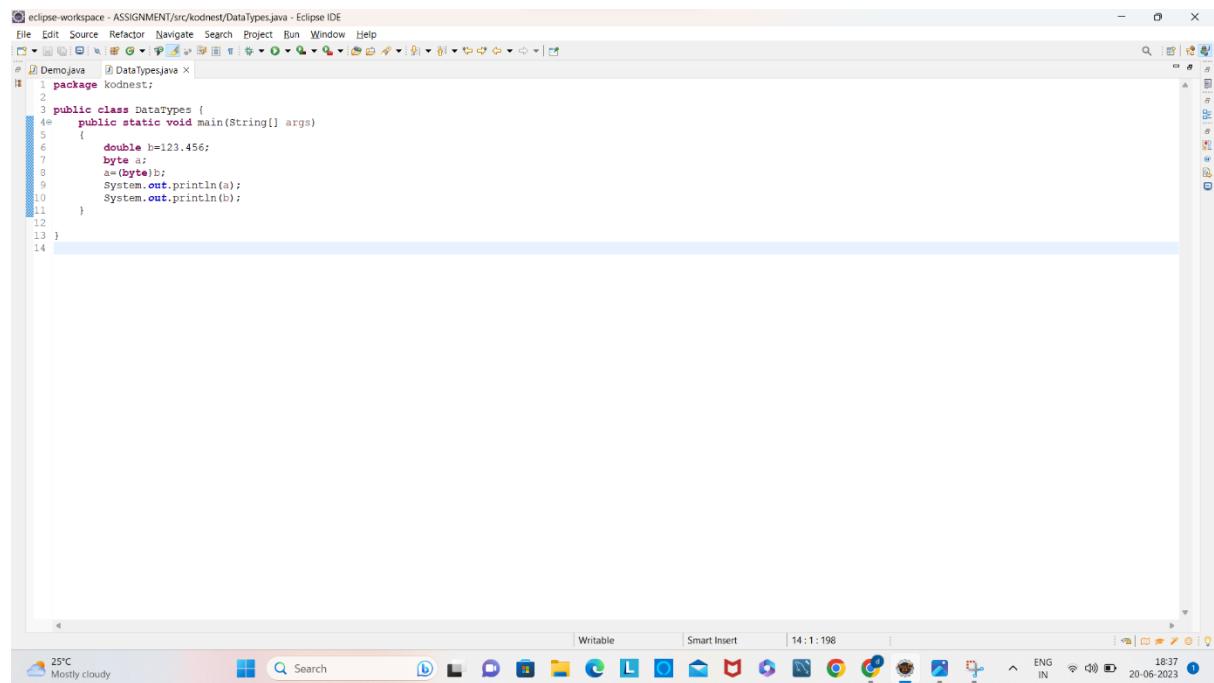
The screenshot shows the Eclipse IDE interface after running the Java application. The output window displays the results of the `System.out.println` statements. The first line shows the character `A`, and the second line shows the double value `65.5`.

```
65.5
```

Conclusion: double to char is possible explicitly.

double to byte:

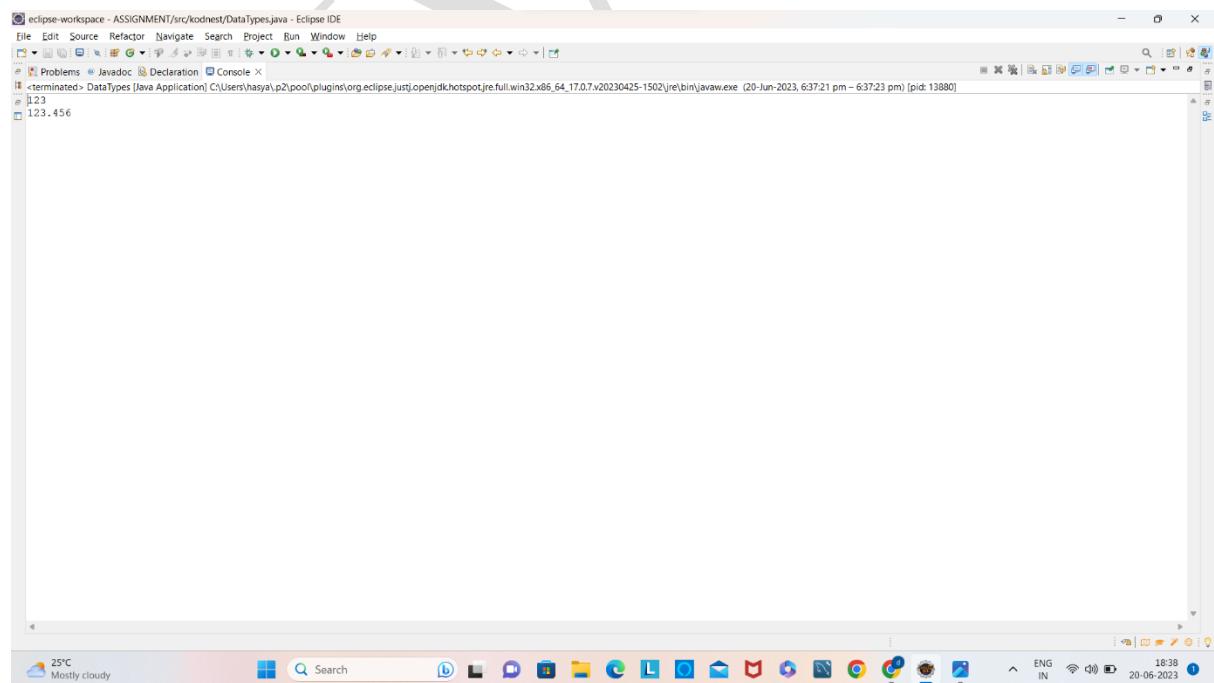
Program:



The screenshot shows the Eclipse IDE interface with a Java file named `DataTypes.java` open. The code defines a class `DataTypes` with a `main` method. Inside the `main` method, a double variable `b` is assigned the value `123.456`. This value is then cast to a byte variable `a` using the expression `a=(byte)b;`. Finally, both variables are printed to the console using `System.out.println`.

```
1 package kodnest;
2
3 public class DataTypes {
4     public static void main(String[] args) {
5         double b=123.456;
6         byte a;
7         a=(byte)b;
8         System.out.println(a);
9         System.out.println(b);
10    }
11 }
12
13 }
```

Output:



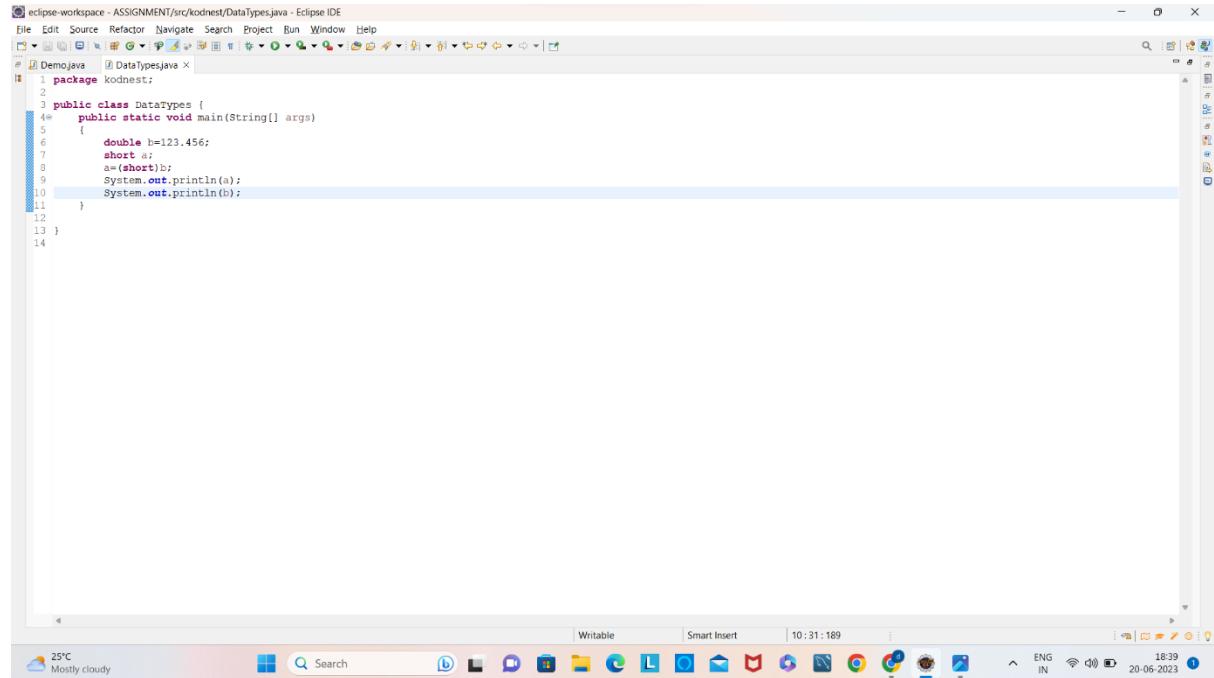
The screenshot shows the Eclipse IDE interface after running the Java application. The output window displays the results of the `System.out.println` statements. The first line shows the value `123`, which is the integer part of the original double value `123.456`. The second line shows the full value `123.456`, indicating that the decimal part was lost during the explicit conversion.

```
123
123.456
```

Conclusion: double to byte is possible explicitly.

double to short:

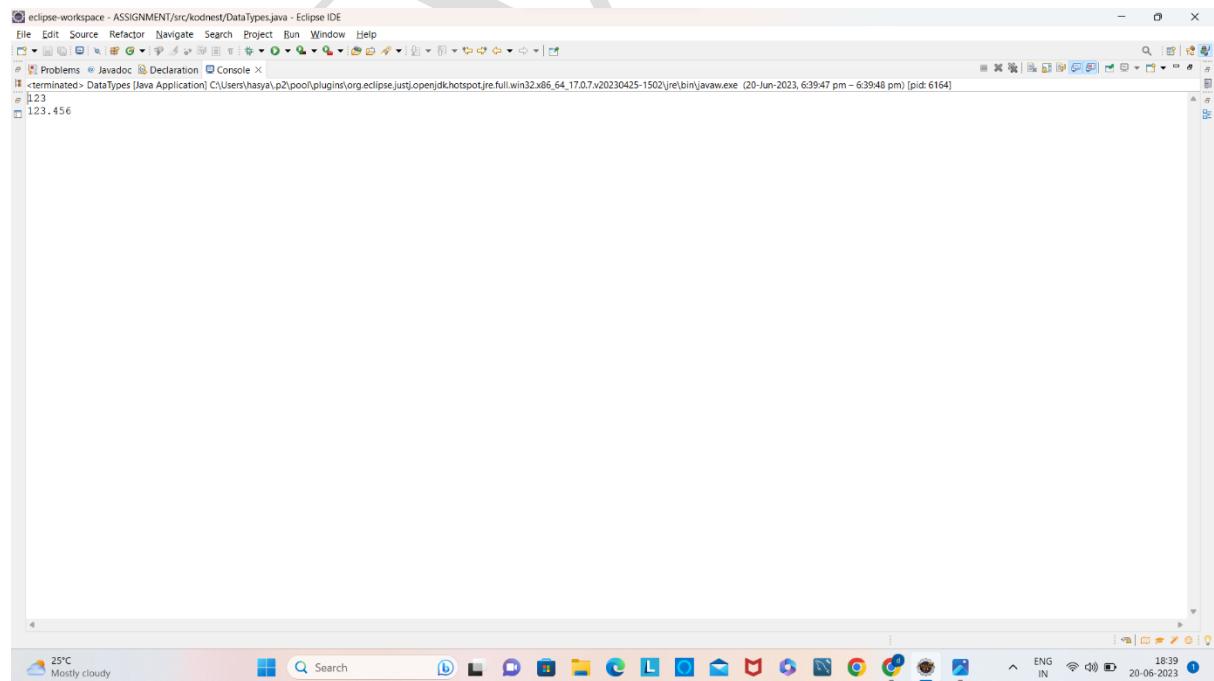
Program:



The screenshot shows the Eclipse IDE interface with a Java file named `DataTypes.java` open. The code demonstrates a narrowing cast from a `double` to a `short`. The code is as follows:

```
1 package kodnest;
2
3 public class DataTypes {
4     public static void main(String[] args) {
5         double b=123.456;
6         short a;
7         a=(short)b;
8         System.out.println(a);
9         System.out.println(b);
10    }
11
12
13 }
```

Output:



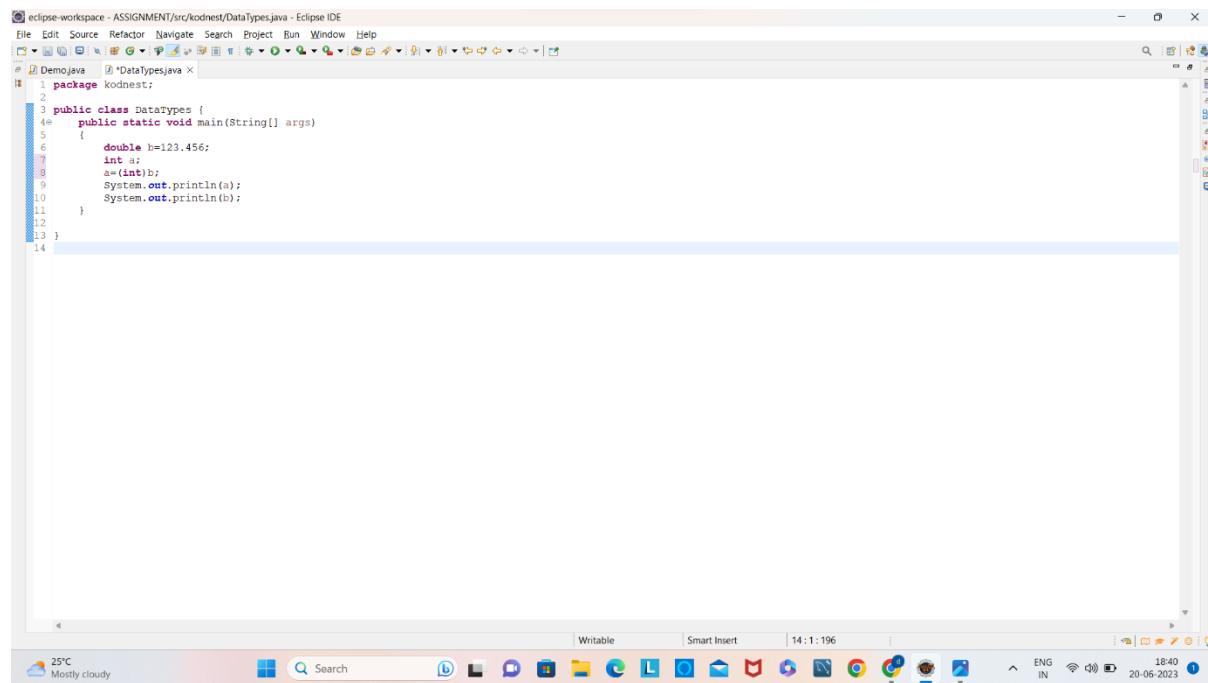
The screenshot shows the Eclipse IDE interface with the Java application running. The console output shows the results of the print statements:

```
123
123.456
```

Conclusion: double to short is possible explicitly.

double to int:

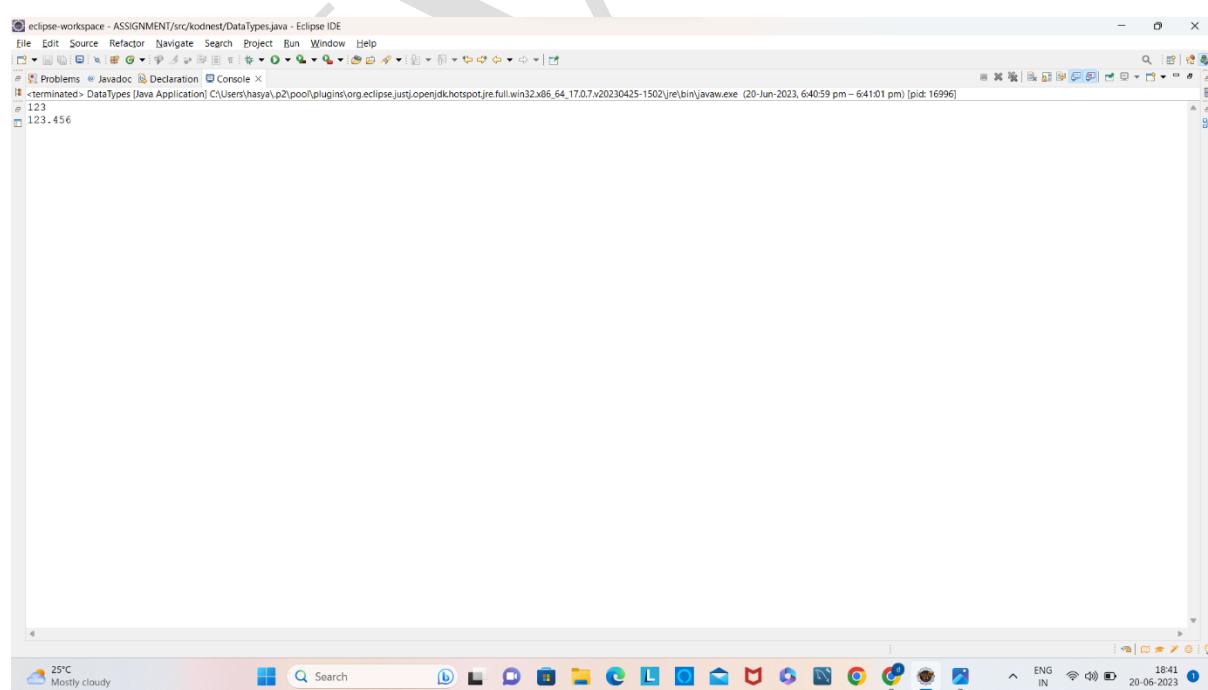
Program:



The screenshot shows the Eclipse IDE interface with a Java file named 'DataTypes.java' open. The code defines a class 'DataTypes' with a main method. In the main method, a double variable 'b' is assigned the value 123.456. It is then cast to an int variable 'a' using the (int)b expression. Both variables are then printed to the console using System.out.println.

```
1 package kodnest;
2
3 public class DataTypes {
4     public static void main(String[] args) {
5         double b=123.456;
6         int a;
7         a=(int)b;
8         System.out.println(a);
9         System.out.println(b);
10    }
11 }
12
13 }
```

Output:



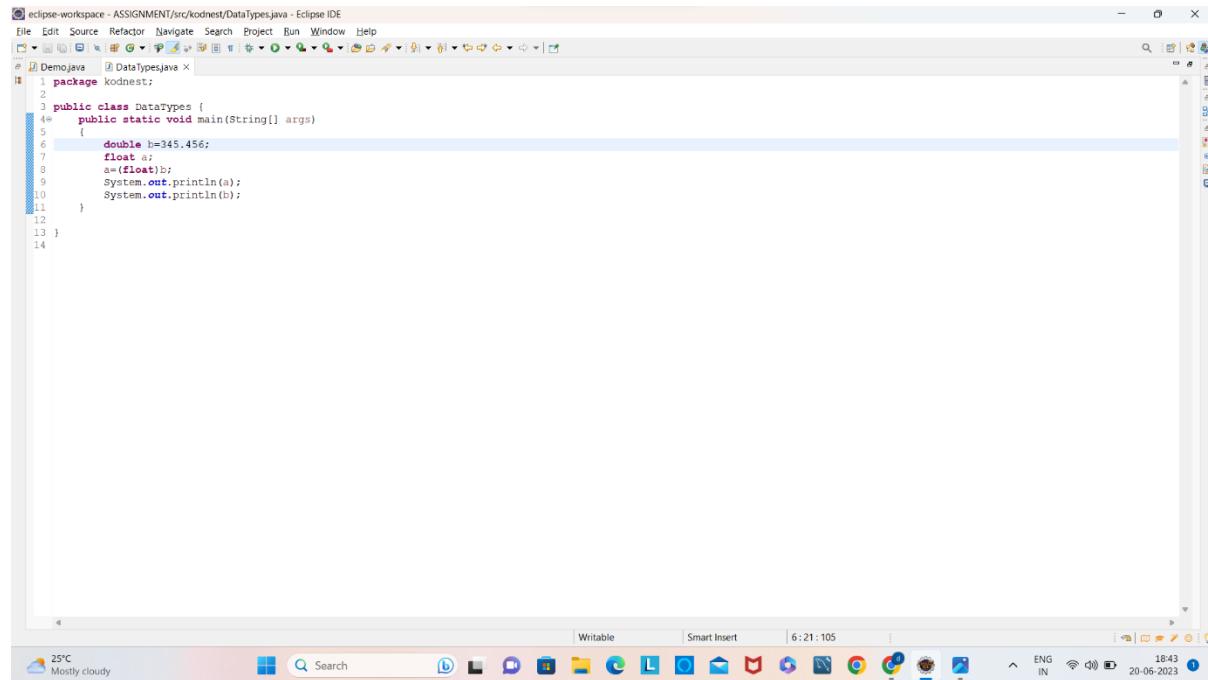
The screenshot shows the Eclipse IDE interface after running the Java application. The 'Console' tab is selected, displaying the output of the program. The output shows the value 123, which is the result of the explicit type conversion of the double value 123.456 to an int.

```
123
```

Conclusion: double to int is possible explicitly.

double to float:

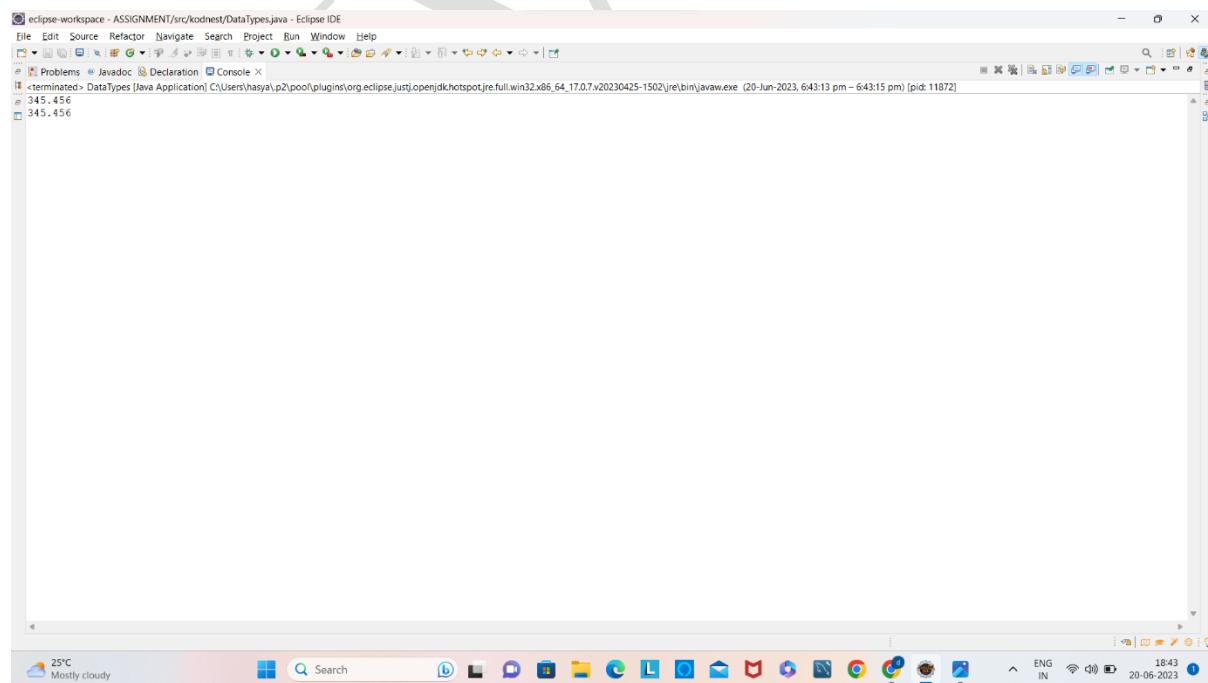
Program:



The screenshot shows the Eclipse IDE interface with a Java file named `DataTypes.java` open. The code defines a class `DataTypes` with a `main` method. In the `main` method, a `double` variable `b` is assigned the value `345.456`. This value is then cast to a `float` variable `a` using the `(float)b` expression. Finally, both variables are printed to the console using `System.out.println`.

```
1 package kodnest;
2
3 public class DataTypes {
4     public static void main(String[] args) {
5         double b=345.456;
6         float a=(float)b;
7         System.out.println(a);
8         System.out.println(b);
9     }
10 }
11
12
13 }
```

Output:



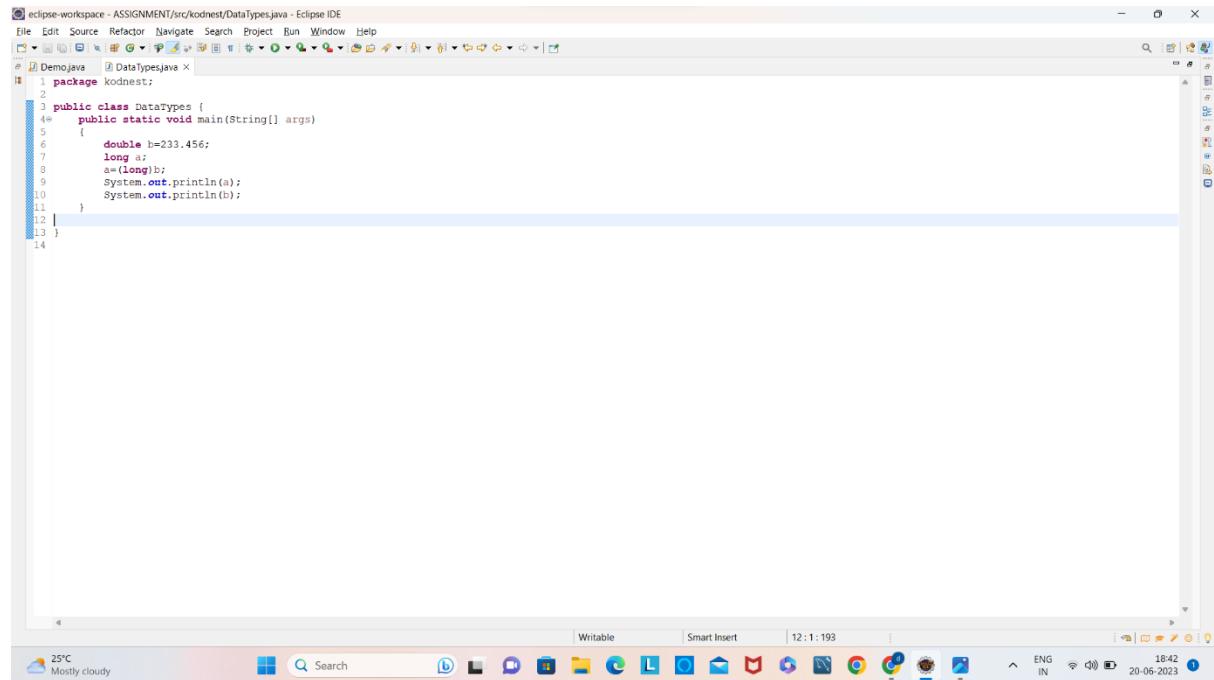
The screenshot shows the Eclipse IDE interface after running the Java application. The output window displays the results of the `System.out.println` statements. The first line shows the value `345.456`, and the second line shows the value `345.456`, demonstrating that the `double` value was successfully converted to a `float`.

```
345.456
345.456
```

Conclusion: double to float is possible explicitly.

double to long:

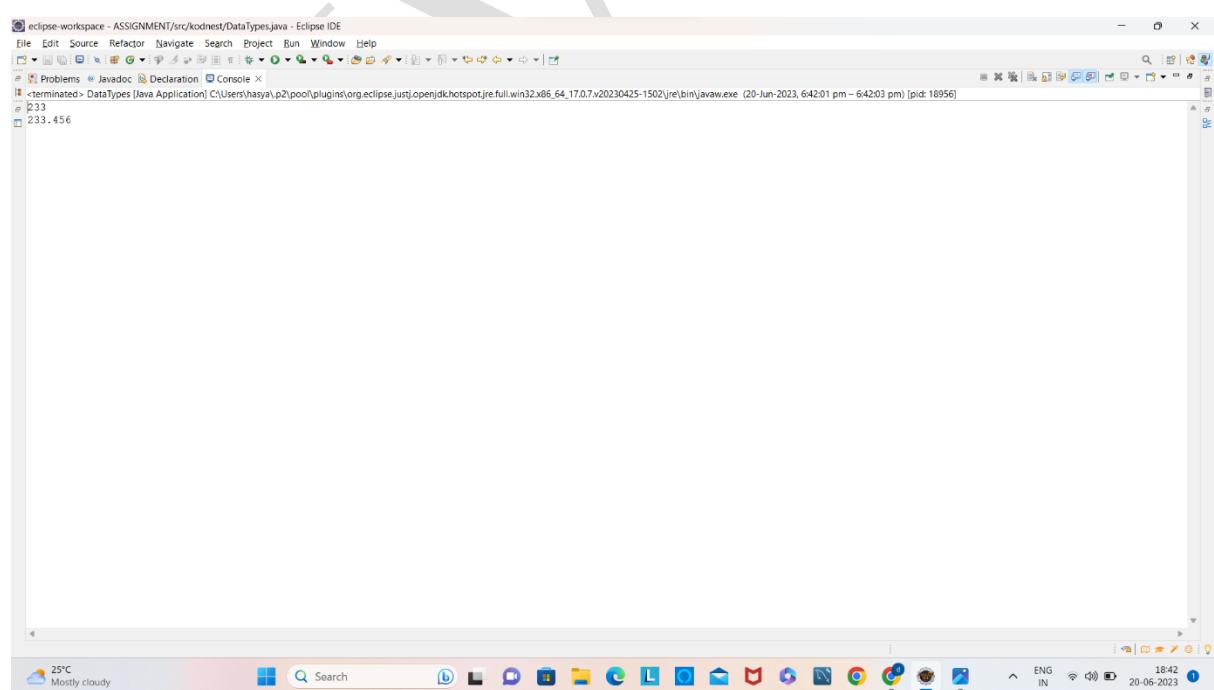
Program:



The screenshot shows the Eclipse IDE interface with a Java file named `DataTypes.java` open. The code defines a class `DataTypes` with a `main` method. Inside the `main` method, a `double` variable `b` is assigned the value `233.456`. This variable is then explicitly cast to a `long` type and stored in variable `a`. Finally, both variables are printed to the console using `System.out.println`.

```
1 package kodnest;
2
3 public class DataTypes {
4     public static void main(String[] args) {
5         double b=233.456;
6         long a;
7         a=(long)b;
8         System.out.println(a);
9         System.out.println(b);
10    }
11 }
12
13 }
```

Output:



The screenshot shows the Eclipse IDE interface after running the Java application. The `Console` tab is active, displaying the output of the `System.out.println` statements. The output shows the value `233` followed by `233.456`, indicating that the double value was successfully converted to a long integer.

```
233
233.456
```

Conclusion: double to long is possible explicitly.

byte to long:

Program:



```
eclipse-workspace - ASSIGNMENT/src/kodnest/DataTypes.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Demo.java DataTypes.java X
1 package kodnest;
2
3 public class DataTypes {
4     public static void main(String[] args) {
5         byte b=7;
6         long a;
7         a=b;
8         System.out.println(a);
9         System.out.println(b);
10    }
11
12
13 }
14
```

Output:



```
eclipse-workspace - ASSIGNMENT/src/kodnest/DataTypes.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Problems JavaView Declaration Console X
terminated - DataTypes [Java Application] C:\Users\hasya\p2pool\plugins\org.eclipse.justj.openjdk.hotspot.jre.full.win32.x86_64_17.0.7.v20230425-1502\jre\bin\javaw.exe (20-Jun-2023, 9:18:16 pm - 9:18:18 pm) [pid: 24976]
87
87
```

Conclusion: byte to long is possible implicitly.

Byte → Short → Int → Long –> Float → Double

Fig : flow of implicit type casting

Double → Float → Long → Int → Short → Byte

Fig : flow of explicit type casting

TABLE FOR TYPE CASTING OF DIFFERENT DATA TYPES