Main.java

*Overloaded Method Resolution*

public class Main {

//Matching exactly overloaded methods parameters

public static void printExact(byte b) {

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

}

public static void printExact(Byte b) {

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

}

public static void printExact(int i) {

System.out.println("Exact int: " + i);

}

public static void printExact(Integer i) {

System.out.println("Exact Integer: " + i);

}

public static void printExact(long l) {

System.out.println("Exact long: " + l);

}

public static void printExact(Long l) {

System.out.println("Exact Long: " + l);

}

public static void printExact(char c) {

System.out.println("Exact char: " + c);

}

//Matching SuperClass of overloaded method parameters

public static void printSuper1(Number n) {

System.out.println("Super1 Number: " + n);

}

public static void printSuper1(Object o) {

System.out.println("Super1 Object: " + o);

}

public static void printSuper2(Object o) {

System.out.println("Super2 Object: " + o);

}

//Matching Larger Primitive Type of overloaded method parameters

public static void printLarge(double d) {

System.out.println("Large doule: " + d);

}

//Matching Autoboxed Type of overloaded method parameters

public static void printAuto(Integer i) {

System.out.println("AutoBox Integer: " + i);

}

public static void printAutoBox(long l) {

System.out.println("AutoBox long: " + l);

}

public static void printAutoBox(Long l) {

System.out.println("AutoBox Long: " + l);

}

public static void printAutoBox(Double d) {

System.out.println("AutoBox Double: " + d);

}

//Matching AutoUnboxed Type of overloaded method parameters

public static void printAutoUnBox1(int i) {

System.out.println("AutoUnBox1 int: " + i);

}

public static void printAutoUnBox1(long l) {

System.out.println("AutoUnBox1 long: " + l);

}

public static void printAutoUnBox1(float f) {

System.out.println("AutoUnBox1 float: " + f);

}

public static void printAutoUnBox2(int i) {

System.out.println("AutoUnBox2 int: " + i);

}

public static void printAutoUnBox2(Number n) {

System.out.println("AutoUnBox2 Number: " + n);

}

//Matching Varargs type of overloaded method parameters

public static void printVarArg(int i, int j) {

System.out.println("VarArg int: " + i + " , int: " + j);

}

public static void printVarArg(int... i) {

System.out.println("VarArg ints: " + i);

}

public static void printVarArg(short... s) {

System.out.println("VarArg shorts: " + s);

}

public static void printVarArg(double... d) {

System.out.println("VarArg doubles: " + d);

}

//Literal to overloaded method parameter check

public static void printLiteral1(byte b) {

System.out.println("Literal1 byte: " + b);

}

public static void printLiteral1(short s) {

System.out.println("Literal1 short: " + s);

}

public static void printLiteral1(int i) {

System.out.println("Literal1 int: " + i);

}

public static void printLiteral1(long l) {

System.out.println("Literal1 long: " + l);

}

public static void printLiteral1(float f) {

System.out.println("Literal1 float: " + f);

}

public static void printLiteral1(double d) {

System.out.println("Literal1 double: " + d);

}

public static void printLiteral1(char c) {

System.out.println("Literal1 char: " + c);

}

public static void printLiteral1(boolean bl) {

System.out.println("Literal1 boolean: " + bl);

}

public static void printLiteral2(byte b) {

System.out.println("Literal2 byte: " + b);

}

public static void printLiteral3(double d) {

System.out.println("Literal3 double: " + d);

}

public static void main(String[] args) {

byte b1 = 1;

Byte b2 = 2;

short s1 = 3;

Short s2 = 4;

int i1 = 5;

Integer i2 = 6;

long l1 = 7;

Long l2 = 8L;

char c1 = 'a';

Character c2 = 'b';

float f1 = 9.0f;

Float f2 = 10.0f;

double d1 = 11.0;

Double d2 = 12.0;

boolean bl1 = true;

Boolean bl2 = false;

//Matching exactly overloaded methods parameters

printExact(b1); //prints: Exact byte: 1

printExact(b2); //prints: Exact Byte: 2

printExact(i1); //prints: Exact int: 5

printExact(i2); //prints: Exact Integer: 6

printExact(l1); //prints: Exact long: 7

printExact(l2); //prints: Exact Long: 8

//Matching SuperClass of overloaded method parameters

printSuper1(b1); //prints: Super1 Number: 1

//(b1 is first autoboxed to Byte)

printSuper1(b2); //prints: Super1 Number: 2

printSuper1(bl1); //prints: Super1 Object: true

//bl1 is first autoboxed to Boolean

printSuper1(bl2); //prints: Super1 Object: false

printSuper2(b1); //prints: Super2 Object: 1

//(b1 is first autoboxed to Byte)

printSuper2(b2); //prints: Super2 Object: 2

printSuper2(bl1); //prints: Super1 Object: true

//Matching Larger Primitive Type of overloaded method parameters

printLarge(b1); //prints: Large doule: 1.0

printLarge(c1); //prints: Large doule: 97.0

printLarge(f1); //prints: Large double: 9.0

//printLarge(bl1); //Compilation fails. The method printLarge(double) in the type Main is not applicable for the arguments (boolean)

//printLarge(bl2); //Compilation fails. The method printLarge(double) in the type Main is not applicable for the arguments (Boolean)

//Matching Autoboxed Type of overloaded method parameters

printAutoBox(i1); //prints: AutoBox long: 5

//Larger Primitive has higher preference than autoboxing

printAutoBox(d1); //prints: AutoBox Double: 11.0

//Matching AutoUnboxed Type of overloaded method parameters

printAutoUnBox1(i2); //prints: AutoUnBox1 int: 6

printAutoUnBox1(l2); //prints: AutoUnBox1 long: 8

printAutoUnBox1(c2); //prints: AutoUnBox1 int: 98

//Character is unboxed to char and then matched with larger primitive int.

//printAutoUnBox1(d2); //Compilation fails. The method printAutoUnBox1(int) in the type Main is not applicable for the arguments (Double)

//Double can be unboxed to double. But then it can go to larger primitive type. Not to smaller type float.

printAutoUnBox2(i2); //prints: AutoUnBox2 Number: 6

//SuperClass has higher preference than autounboxing

//Matching Varargs type of overloaded method parameters

printVarArg(i1); //prints: VarArg ints: [I@1db9742

printVarArg(i1, i1); //prints: VarArg int: 5 , int: 5

printVarArg(i1, i2); //prints: VarArg int: 5 , int: 6

printVarArg(i1, i1, i1); //prints: VarArg ints: [I@106d69c

printVarArg(d1); //prints: VarArg doubles: [D@52e922

printVarArg(d1, d1); //prints: VarArg doubles: [D@25154f

printVarArg(d1, d2); //prints: VarArg doubles: [D@10dea4e

printVarArg(); //prints: VarArg shorts: [S@647e05

printVarArg(s1, d1); //prints: VarArg doubles: [D@1909752

printVarArg(s2, d2); //prints: VarArg doubles: [D@1f96302

//Literal to overloaded method parameter check

printLiteral1(1); //prints: Literal1 int: 1

printLiteral1(01); //prints: Literal1 int: 1

printLiteral1(0b1); //prints: Literal1 int: 1

printLiteral1(0x1); //prints: Literal1 int: 1

printLiteral1(0X1); //prints: Literal1 int: 1

printLiteral1(2\_147\_483\_647); //prints: Literal1 int: 2147483647

printLiteral1(0x7FFFFFFF); //prints: Literal1 int: 2147483647

printLiteral1(0x80000000); //prints: Literal1 int: -2147483648

//printLiteral1(2\_147\_483\_648); //Compilation fails. The literal 2\_147\_483\_648 of type int is out of range

printLiteral1(2L); //prints: Literal1 long: 2

printLiteral1(2\_147\_483\_648L); //prints: Literal1 long: 2147483648

printLiteral1(3.0); //prints: Literal1 double: 3.0

printLiteral1(3E0); //prints: Literal1 double: 3.0

printLiteral1(4.0F); //prints: Literal1 float: 4.0

printLiteral1(4E0F); //prints: Literal1 float: 4.0

printLiteral1('a'); //prints: Literal1 char: a

printLiteral1(true); //prints: Literal1 boolean: true

//printLiteral2(1); //Compilation fails. The method printLiteral2(byte) in the type Main is not applicable for the arguments (int)

printLiteral3(1); //prints: Literal3 double: 1.0

}

}