**Java Assignment 2 Sandeep Sir**

**1.**

**a. Modifiers and Super Types:**

* **Modifiers:** public final
* **Super Types:** Object

**b. Converting boolean to String:**

boolean status = true;

String strStatus = Boolean.toString(status);

System.out.println(strStatus);

Output: true

**c. Converting String to boolean:**

String strStatus = "true";

boolean status = Boolean.parseBoolean(strStatus);

System.out.println(status);

Output: true

**d. Converting "1" or "0" to boolean (incorrect):**

String strStatus = "1";

boolean status = Boolean.parseBoolean(strStatus);

System.out.println(status); // Output: false (incorrect)

* The parseBoolean method treats "1" or "0" as false. To correctly convert "1" or "0" to boolean, you should use Integer.parseInt and then compare the result with 0.

**e. Converting boolean to Boolean wrapper:**

Java

boolean status = true;

Boolean wrapperStatus = Boolean.valueOf(status);

System.out.println(wrapperStatus); // Output: true

**f. Converting String to Boolean wrapper:**

Java

String strStatus = "true";

Boolean wrapperStatus = Boolean.valueOf(strStatus);

System.out.println(wrapperStatus); // Output: true

**g. Converting boolean to other primitive types or vice versa:**

* **boolean to int/char:**

int intValue = (status) ? 1 : 0;

char charValue = (status) ? 'T' : 'F';

* **int/char to boolean:**

boolean boolValue = (intValue != 0);

boolean boolValue = (charValue == 'T');

* **boolean to String:**

String strValue = Boolean.toString(status);

* **String to boolean:**

boolean boolValue = Boolean.parseBoolean(strValue);

2.

**a. Modifiers and Super Types:**

* **Modifiers:** public final
* **Super Types:** Object

**b. Testing the number of bytes used to represent a byte value:**

Java

System.out.println("Number of bytes used to represent a byte value: " + Byte.BYTES);

**c. Finding the minimum and maximum values of byte:**

Java

System.out.println("Minimum value of byte: " + Byte.MIN\_VALUE);

System.out.println("Maximum value of byte: " + Byte.MAX\_VALUE);

**d. Converting byte to String:**

byte number = 42;

String strNumber = Byte.toString(number);

System.out.println(strNumber); // Output: 42

**e. Converting String to byte:**

String strNumber = "127";

byte number = Byte.parseByte(strNumber);

System.out.println(number); // Output: 127

**f. Converting "Ab12Cd3" to byte (incorrect):**

Java

String strNumber = "Ab12Cd3";

try {

byte number = Byte.parseByte(strNumber);

System.out.println(number); // This will throw a NumberFormatException

} catch (NumberFormatException e) {

System.out.println("Invalid input: " + e.getMessage());

}

* The parseByte method will throw a NumberFormatException if the input string cannot be parsed into a byte value.

**g. Converting byte to Byte wrapper:**

Java

byte number = 42;

Byte wrapperNumber = Byte.valueOf(number);

System.out.println(wrapperNumber); // Output: 42

**h. Converting String to Byte wrapper:**

Java

String strNumber = "127";

Byte wrapperNumber = Byte.valueOf(strNumber);

System.out.println(wrapperNumber); // Output: 127

**i. Converting byte to other primitive types or vice versa:**

* **byte to int/long/short:**
  + int intValue = number;
  + long longValue = number;
  + short shortValue = number;
* **int/long/short to byte:**
  + byte byteValue = (byte) intValue;
  + byte byteValue = (byte) longValue;
  + byte byteValue = (byte) shortValue;
* **byte to char:**
  + char charValue = (char) number;
* **char to byte:**
  + byte byteValue = (byte) charValue;
* **byte to float/double:**

float floatValue = number;

double doubleValue = number;

* **float/double to byte:**

byte byteValue = (byte) floatValue;

byte byteValue = (byte) doubleValue;

4.

**a. Modifiers and Super Types:**

* **Modifiers:** public, final, static
* **Super Types:** Object

**b. Bytes Used to Represent an int:**

public class IntegerBytesExample {

public static void main(String[] args) {

int bytesUsed = Integer.BYTES;

System.out.println("Bytes used to represent an int: " + bytesUsed);

}

}

**c. Minimum and Maximum Values of int:**

public class IntegerMinMaxExample {

public static void main(String[] args) {

int minValue = Integer.MIN\_VALUE;

int maxValue = Integer.MAX\_VALUE;

System.out.println("Minimum   int value: " + minValue);

System.out.println("Maximum int value: " + maxValue);

}

}

**d. Converting int to String:**

public class IntToStringExample {

public static void main(String[] args) {

int number = 123;

String strNumber = Integer.toString(number);

System.out.println("Integer as String: " + strNumber);

}

}

**e. Converting String to int:**

public class StringToIntExample {

public static void main(String[] args) {

String strNumber = "456";

int number = Integer.parseInt(strNumber);

System.out.println("String as Integer: " + number);

}

}

**f. Handling NumberFormatException:**

public class InvalidStringToIntExample {

public static void main(String[] args) {

String strNumber = "Ab12Cd3";

try {

int number = Integer.parseInt(strNumber);

System.out.println("String as Integer: " + number);

} catch (NumberFormatException e) {

System.out.println("Invalid input: " + e.getMessage());

}

}

}

**g. Converting int to Integer:**

public class IntToIntegerExample {

public static void main(String[] args) {

int number = 789;

Integer integerObject = Integer.valueOf(number);

System.out.println("Integer object: " + integerObject);

}

}

**h. Converting String to Integer:**

public class StringToIntegerExample {

public static void main(String[] args) {

String strNumber = "100";

Integer integerObject = Integer.valueOf(strNumber);

System.out.println("Integer object: " + integerObject);

}

}

**i. Adding Integers:**

public class IntegerAdditionExample {

public static void main(String[] args) {

int num1 = 10;

int num2 = 20;

int sum = Integer.sum(num1, num2);

System.out.println("Sum: " + sum);

}

}

**j. Finding Minimum and Maximum Integers:**

public class IntegerMinMaxExample {

public static void main(String[] args) {

int num1 = 10;

int num2 = 20;

int min = Integer.min(num1, num2);

int max = Integer.max(num1, num2);

System.out.println("Minimum: " + min);

System.out.println("Maximum: " + max);

}

}

**k. Converting int to Binary, Octal, and Hexadecimal:**

Java

public class IntegerConversionExample {

public static void main(String[] args) {

int number = 7;

String binaryString = Integer.toBinaryString(number);

String octalString = Integer.toOctalString(number);

String hexadecimalString = Integer.toHexString(number);

System.out.println("Binary: " + binaryString);

System.out.println("Octal: " + octalString);

System.out.println("Hexadecimal: " + hexadecimalString);

}

}

**l. Converting int to Other Primitive Types:**

* **int to byte, short, char:** Truncation may occur if the value is too large.
* **int to long:** No data loss.
* **int to float, double:** Potential loss of precision due to floating-point representation.
* **Other primitive types to int:** Conversion may involve widening or narrowing, potentially leading to data loss or overflow.

3.

**Java short Data Type**

**a. Modifiers and Super Types**

* **Modifiers:** public, final, static
* **Super Type:** Object

**b. Bytes Used to Represent a short**

public class ShortBytesExample {

public static void main(String[] args) {

int bytesUsed = Short.BYTES;

System.out.println("Bytes used to represent a short: " + bytesUsed);

}

}

**c. Minimum and Maximum Values of short**

public class ShortMinMaxExample {

public static void main(String[] args) {

short minValue = Short.MIN\_VALUE;

short maxValue = Short.MAX\_VALUE;

System.out.println("Minimum short value: " + minValue);

System.out.println("Maximum short value: " + maxValue);

}

}

**d. Converting short to String**

public class ShortToStringExample {

public static void main(String[] args) {

short number = 123;

String strNumber = Short.toString(number);

System.out.println("Short as String: " + strNumber);

}

}

**e. Converting String to short**

public class StringToShortExample {

public static void main(String[] args) {

String strNumber = "456";

short number = Short.parseShort(strNumber);

System.out.println("String as Short: " + number);

}

}

**f. Handling NumberFormatException**

public class InvalidStringToShortExample {

public static void main(String[] args) {

String strNumber = "Ab12Cd3";

try {

short number = Short.parseShort(strNumber);

System.out.println("String as Short: " + number);

} catch (NumberFormatException e) {

System.out.println("Invalid input: " + e.getMessage());

}

}

}

**g. Converting short to Short Wrapper Class**

public class ShortToShortWrapperExample {

public static void main(String[] args) {

short number = 789;

Short shortObject = Short.valueOf(number);

System.out.println("Short object: " + shortObject);

}

}

**h. Converting String to Short Wrapper Class**

public class StringToShortWrapperExample {

public static void main(String[] args) {

String strNumber = "100";

Short shortObject = Short.valueOf(strNumber);

System.out.println("Short object: " + shortObject);

}

}

**i. Converting short to Other Primitive Types**

* **short to byte:** No data loss if the short value is within the range of byte.
* **short to char:** Conversion is possible, but the value will be treated as a Unicode code point.
* **short to int:** No data loss.
* **short to long:** No data loss.
* **short to float or double:** Potential loss of precision due to floating-point representation.
* **Other primitive types to short:** Conversion may involve narrowing, potentially leading to data loss or overflow.

5.

**Java long Data Type**

**a. Modifiers and Super Types**

* **Modifiers:** public, final, static
* **Super Type:** Object

**b. Bytes Used to Represent a long**

public class LongBytesExample {

public static void main(String[] args) {

int bytesUsed = Long.BYTES;

System.out.println("Bytes used to represent a long: " + bytesUsed);

}

}

**c. Minimum and Maximum Values of long**

public class LongMinMaxExample {

public static void main(String[] args) {

long minValue = Long.MIN\_VALUE;

long maxValue = Long.MAX\_VALUE;

System.out.println("Minimum long value: " + minValue);

System.out.println("Maximum long value: " + maxValue);

}

}

**d. Converting long to String**

public class LongToStringExample {

public static void main(String[] args) {

long number = 1234567890;

String strNumber = Long.toString(number);

System.out.println("Long as String: " + strNumber);

}

}

**e. Converting String to long**

public class StringToLongExample {

public static void main(String[] args) {

String strNumber = "4567890123";

long number = Long.parseLong(strNumber);

System.out.println("String as Long: " + number);

}

}

**f. Handling NumberFormatException**

public class InvalidStringToLongExample {

public static void main(String[] args) {

String strNumber = "Ab12Cd3";

try {

long number = Long.parseLong(strNumber);

System.out.println("String as Long: " + number);

} catch (NumberFormatException e) {

System.out.println("Invalid input: " + e.getMessage());

}

}

}

**g. Converting long to Long Wrapper Class**

public class LongToLongWrapperExample {

public static void main(String[] args) {

long number = 7890123456;

Long longObject = Long.valueOf(number);

System.out.println("Long object: " + longObject);

}

}

**h. Converting String to Long Wrapper Class**

public class StringToLongWrapperExample {

public static void main(String[] args) {

String strNumber = "1000000000";

Long longObject = Long.valueOf(strNumber);

System.out.println("Long object: " + longObject);

}

}

**i. Adding Longs**

public class LongAdditionExample {

public static void main(String[] args) {

long num1 = 1123;

long num2 = 9845;

long sum = Long.sum(num1, num2);

System.out.println("Sum: " + sum);

}

}

**j. Finding Minimum and Maximum Longs**

public class LongMinMaxExample {

public static void main(String[] args) {

long num1 = 1122;

long num2 = 5566;

long min = Long.min(num1, num2);

long max = Long.max(num1, num2);

System.out.println("Minimum: " + min);

System.out.println("Maximum: " + max);

}

}

**k. Converting long to Binary, Octal, and Hexadecimal**

public class LongConversionExample {

public static void main(String[] args) {

long number = 7;

String binaryString = Long.toBinaryString(number);

String octalString = Long.toOctalString(number);

String hexadecimalString = Long.toHexString(number);

System.out.println("Binary: " + binaryString);

System.out.println("Octal: " + octalString);

System.out.println("Hexadecimal: " + hexadecimalString);

}

}

**l. Converting long to Other Primitive Types**

* **long to byte, short, char, int:** Truncation may occur if the long value is too large.
* **long to float, double:** Potential loss of precision due to floating-point representation.
* **Other primitive types to long:** Conversion may involve widening, potentially leading to data loss or overflow.

6.

**Java float Data Type**

**a. Modifiers and Super Types**

* **Modifiers:** public, final, static
* **Super Type:** Object

**b. Bytes Used to Represent a float**

public class FloatBytesExample {

public static void main(String[] args) {

int bytesUsed = Float.BYTES;

System.out.println("Bytes used to represent a float: " + bytesUsed);

}

}

**c. Minimum and Maximum Values of float**

public class FloatMinMaxExample {

public static void main(String[] args) {

float minValue = Float.MIN\_VALUE;

float maxValue = Float.MAX\_VALUE;

System.out.println("Minimum float value: " + minValue);

System.out.println("Maximum float value: " + maxValue);

}

}

**d. Converting float to String**

public class FloatToStringExample {

public static void main(String[] args) {

float number = 123.456f;

String strNumber = Float.toString(number);

System.out.println("Float as String: " + strNumber);

}

}

**e. Converting String to float**

public class StringToFloatExample {

public static void main(String[] args) {

String strNumber = "456.789";

float number = Float.parseFloat(strNumber);

System.out.println("String as Float: " + number);

}

}

**f. Handling NumberFormatException**

public class InvalidStringToFloatExample {

public static void main(String[] args) {

String strNumber = "Ab12Cd3";

try {

float number = Float.parseFloat(strNumber);

System.out.println("String as Float: " + number);

} catch (NumberFormatException e) {

System.out.println("Invalid input: " + e.getMessage());

}

}

}

**g. Converting float to Float Wrapper Class**

public class FloatToFloatWrapperExample {

public static void main(String[] args) {

float number = 789.012f;

Float floatObject = Float.valueOf(number);

System.out.println("Float object: " + floatObject);

}

}

**h. Converting String to Float Wrapper Class**

public class StringToFloatWrapperExample {

public static void main(String[] args) {

String strNumber = "100.00";

Float floatObject = Float.valueOf(strNumber);

System.out.println("Float object: " + floatObject);

}

}

**i. Adding Floats**

public class FloatAdditionExample {

public static void main(String[] args) {

float num1 = 112.3f;

float num2 = 984.5f;

float sum = Float.sum(num1, num2);

System.out.println("Sum: " + sum);

}

}

**j. Finding Minimum and Maximum Floats**

public class FloatMinMaxExample {

public static void main(String[] args) {

float num1 = 112.2f;

float num2 = 556.6f;

float min = Float.min(num1, num2);

float max = Float.max(num1, num2);

System.out.println("Minimum: " + min);

System.out.println("Maximum: " + max);

}

}

**k. Square Root of a Float**

public class FloatSqrtExample {

public static void main(String[] args) {

float number = -25.0f;

double sqrt = Math.sqrt(number); // Using Math.sqrt for square root

System.out.println("Square root: " + sqrt);

}

}

**l. Dividing Zero by Zero**

public class FloatDivisionByZeroExample {

public static void main(String[] args) {

float num1 = 0.0f;

float num2 = 0.0f;

float result = num1 / num2; // This will result in a NaN (Not a Number)

System.out.println("Result: " + result);

}

}

**m. Converting float to Other Primitive Types**

* **float to byte, short, char, int, long:** Truncation may occur, and precision may be lost.
* **float to double:** Widening conversion, no data loss (but precision may be different).
* **Other primitive types to float:** Conversion may involve widening or narrowing, potentially leading to data loss or overflow.

7.

**Java double Data Type**

**a. Modifiers and Super Types**

* **Modifiers:** public, final, static
* **Super Type:** Object

**b. Bytes Used to Represent a double**

public class DoubleBytesExample {

public static void main(String[] args) {

int bytesUsed = Double.BYTES;

System.out.println("Bytes used to represent a double: " + bytesUsed);

}

}

**c. Minimum and Maximum Values of double**

public class DoubleMinMaxExample {

public static void main(String[] args) {

double minValue = Double.MIN\_VALUE;

double maxValue = Double.MAX\_VALUE;

System.out.println("Minimum double value: " + minValue);

System.out.println("Maximum double value: " + maxValue);

}

}

**d. Converting double to String**

public class DoubleToStringExample {

public static void main(String[] args) {

double number = 123.456;

String strNumber = Double.toString(number);

System.out.println("Double as String: " + strNumber);

}

}

**e. Converting String to double**

public class StringToDoubleExample {

public static void main(String[] args) {

String strNumber = "456.789";

double number = Double.parseDouble(strNumber);

System.out.println("String as Double: " + number);

}

}

**f. Handling NumberFormatException**

public class InvalidStringToDoubleExample {

public static void main(String[] args) {

String strNumber = "Ab12Cd3";

try {

double number = Double.parseDouble(strNumber);

System.out.println("String as Double: " + number);

} catch (NumberFormatException e) {

System.out.println("Invalid input: " + e.getMessage());

}

}

}

**g. Converting double to Double Wrapper Class**

public class DoubleToDoubleWrapperExample {

public static void main(String[] args) {

double number = 789.012;

Double doubleObject = Double.valueOf(number);

System.out.println("Double object: " + doubleObject);

}

}

**h. Converting String to Double Wrapper Class**

public class StringToDoubleWrapperExample {

public static void main(String[] args) {

String strNumber = "100.00";

Double doubleObject = Double.valueOf(strNumber);

System.out.println("Double object: " + doubleObject);

}

}

**i. Adding Doubles**

public class DoubleAdditionExample {

public static void main(String[] args) {

double num1 = 112.3;

double num2 = 984.5;

double sum = Double.sum(num1, num2);

System.out.println("Sum: " + sum);

}

}

**j. Finding Minimum and Maximum Doubles**

public class DoubleMinMaxExample {

public static void main(String[] args) {

double num1 = 112.2;

double num2 = 556.6;

double min = Double.min(num1, num2);

double max = Double.max(num1, num2);

System.out.println("Minimum: " + min);

System.out.println("Maximum: " + max);

}

}

**k. Square Root of a Double**

public class DoubleSqrtExample {

public static void main(String[] args) {

double number = -25.0;

double sqrt = Math.sqrt(number); // Using Math.sqrt for square root

System.out.println("Square root: " + sqrt);

}

}

**l. Dividing Zero by Zero**

public class DoubleDivisionByZeroExample {

public static void main(String[] args) {

double num1 = 0.0;

double num2 = 0.0;

double result = num1 / num2; // This will result in a NaN (Not a Number)

System.out.println("Result: " + result);

}

}

**m. Converting double to Other Primitive Types**

* **double to byte, short, char, int, long, float:** Truncation may occur, and precision may be lost.
* **Other primitive types to double:** Conversion may involve widening, potentially leading to data loss or overflow.

8.

**Converting Primitive Types to Strings**

**Using Wrapper Class toString() Methods**

Java

public class PrimitiveToStringExample {

public static void main(String[] args) {

// Initialize variables of each primitive type

byte b = 10;

short s = 20;

int i = 30;

long l = 40L;

float f = 50.5f;

double d = 60.6;

char c = 'A';

boolean bool = true;

// Convert to strings using wrapper class toString() methods

String strB = Byte.toString(b);

String strS = Short.toString(s);

String strI = Integer.toString(i);

String strL = Long.toString(l);

String strF = Float.toString(f);

String strD = Double.toString(d);

String strC = Character.toString(c);

String strBool = Boolean.toString(bool);

// Print the results

System.out.println("Byte as String: " + strB);

System.out.println("Short as String: " + strS);

System.out.println("Integer as String: " + strI);

System.out.println("Long as String: " + strL);

System.out.println("Float as String: " + strF);

System.out.println("Double as String: " + strD);

System.out.println("Character as String: " + strC);

System.out.println("Boolean as String: " + strBool);

}

}

**Using String.valueOf()**

Java

public class PrimitiveToStringExample {

public static void main(String[] args) {

// Initialize variables of each primitive type

// ... (same as before)

// Convert to strings using String.valueOf()

String strB = String.valueOf(b);

String strS = String.valueOf(s);

String strI = String.valueOf(i);

String strL = String.valueOf(l);

String strF = String.valueOf(f);

String strD = String.valueOf(d);

String strC = String.valueOf(c);

String strBool = String.valueOf(bool);

// Print the results

// ... (same as before)

}

}

9.

**Default Values of Primitive Types**

**Instance Variables:**

* byte: 0
* short: 0
* int: 0
* long: 0L
* float: 0.0f
* double: 0.0
* char: '\u0000' (null character)
* boolean: false

**Static Variables:**

* byte: 0
* short: 0
* int: 0
* long: 0L
* float: 0.0f
* double: 0.0
* char: '\u0000' (null character)
* boolean: false

**Example:**

public class DefaultValuesExample {

// Instance variables

byte instanceByte;

short instanceShort;

int instanceInt;

long instanceLong;

float instanceFloat;

double instanceDouble;

char instanceChar;

boolean instanceBoolean;

// Static variables

static byte staticByte;

static short staticShort;

static int staticInt;

static long staticLong;

static float staticFloat;

static double staticDouble;

static char staticChar;

static   boolean staticBoolean;

public static void main(String[] args) {

// Create an instance of the class

DefaultValuesExample obj = new DefaultValuesExample();

// Print the default values

System.out.println("Instance Variables:");

System.out.println("byte: " + obj.instanceByte);

System.out.println("short: " + obj.instanceShort);

System.out.println("int: " + obj.instanceInt);

System.out.println("long: " + obj.instanceLong);

System.out.println("float: " + obj.instanceFloat);

System.out.println("double: " + obj.instanceDouble);

System.out.println("char: " + obj.instanceChar);

System.out.println("boolean: " + obj.instanceBoolean);

System.out.println("\nStatic Variables:");

System.out.println("byte: " + staticByte);

System.out.println("short: " + staticShort);

System.out.println("int: " + staticInt);

System.out.println("long: " + staticLong);

System.out.println("float: " + staticFloat);

System.out.println("double: " + staticDouble);

System.out.println("char: " + staticChar);

System.out.println("boolean: " + staticBoolean);

}

}

10.

public class CommandLineArithmetic {

public static void main(String[] args) {

if (args.length != 3) {

System.out.println("Usage: java CommandLineArithmetic <operand1> <operator> <operand2>");

return;

}

// Assuming valid input format

int operand1 = Integer.parseInt(args[0]);

int operand2 = Integer.parseInt(args[2]);

char operator = args[1].charAt(0);

switch (operator) {

case '+':

System.out.println(operand1 + " + " + operand2 + " = " + (operand1 + operand2));

break;

case '-':

System.out.println(operand1 + " - " + operand2 + " = " + (operand1 - operand2));

break;

case '\*':

System.out.println(operand1 + " \* " + operand2 + " = " + (operand1 \* operand2));

break;

case '/':

if (operand2 == 0) {

System.out.println("Error: Division by zero is not allowed.");

} else {

System.out.println(operand1 + " / " + operand2 + " = " + (operand1 / operand2));

}

break;

default:

System.out.println("Invalid operator. Please use +, -, \*, or /.");

}

}

}