**JAVA CHAPTER 2**

**Java Variables**

A variable is a container which holds the value while the [Java program](https://www.javatpoint.com/simple-program-of-java) is executed.

Variable is a name of memory location.

There are three types of variables in java

* local variables
* instance variables
* static variables

A variable is created with a data type. We can call it as variable declaration.

There are two types of [data types in Java](https://www.javatpoint.com/java-data-types)

primitive data types: These are basic and not derived from any other types. For these types no methods are available.

Example: byte, short, int, long, float, double, boolean, char

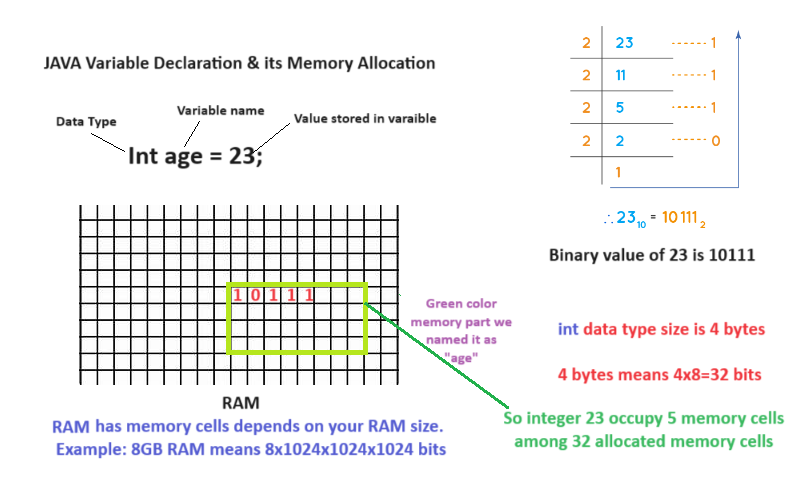
non-primitive data types: These are derived from primitive types. For these types methods are available for each type.

Example: [String](https://www.w3schools.com/java/java_strings.asp), [Arrays](https://www.w3schools.com/java/java_arrays.asp) and [Classes](https://www.w3schools.com/java/java_classes.asp)(Integer, Float, Double, Long, Student, Employee etc)

Explain primitive data types in java?

|  |  |  |
| --- | --- | --- |
| **Data Type** | **Size** | **Description** |
| byte | 1 byte | Stores whole numbers from -128 to 127 |
| short | 2 bytes | Stores whole numbers from -32,768 to 32,767 |
| int | 4 bytes | Stores whole numbers from -2,147,483,648 to 2,147,483,647 |
| long | 8 bytes | Stores whole numbers from -9,223,372,036,854,775,808 to 9,223,372,036,854,775,807 |
| float | 4 bytes | Stores fractional numbers. Sufficient for storing 6 to 7 decimal digits |
| double | 8 bytes | Stores fractional numbers. Sufficient for storing 15 decimal digits |
| boolean | 1 bit | Stores true or false values |
| char | 2 bytes | Stores a single character/letter or ASCII values |

What happened when we declare an integer variable as int age = 23?



Write a program to display byte variable?

public class Main {

public static void main(String[] args) {

byte myNum = 100;

System.out.println(myNum);

}

}

Output:

100

Write a program to display short variable?

public class Main {

public static void main(String[] args) {

short myNum = 5000;

System.out.println(myNum);

}

}

Output:

5000

Write a program to display int variable?

public class Main {

public static void main(String[] args) {

int myNum = 100000;

System.out.println(myNum);

}

}

Output:

100000

Write a program to display long variable?

public class Main {

public static void main(String[] args) {

long myNum = 15000000000L;

System.out.println(myNum);

}

}

Output:

15000000000

Write a program to display float variable?

public class Main {

public static void main(String[] args) {

float myNum = 5.75f;

System.out.println(myNum);

}

}

Output:

5.75

Write a program to display double variable?

public class Main {

public static void main(String[] args) {

double myNum = 19.99d;

System.out.println(myNum);

}

}

Output:

19.99

Write a program to display scientific numbers?

public class Main {

public static void main(String[] args) {

float f1 = 35e3f;

double d1 = 12E4d;

System.out.println(f1);

System.out.println(d1);

}

}

Output:

35000.0  
120000.0

Write a program to display Boolean values?

public class Main {

public static void main(String[] args) {

boolean isJavaFun = true;

boolean isFishTasty = false;

System.out.println(isJavaFun);

System.out.println(isFishTasty);

}

}

Output:

true  
false

Write a program to display characters?

public class Main {

public static void main(String[] args) {

char myGrade = 'B';

System.out.println(myGrade);

}

}

Output:

B

Write a program to display certain characters by its ASCII values?

public class Main {

public static void main(String[] args) {

char myVar1 = 65, myVar2 = 66, myVar3 = 67;

System.out.println(myVar1);

System.out.println(myVar2);

System.out.println(myVar3);

}

}

Output:

A  
B  
C

Write a program to display String?

public class Main {

public static void main(String[] args) {

String greeting = "Hello World";

System.out.println(greeting);

}

}

Output

Hello World

**Java Type Casting**

Type casting is a process happens when you assign a value of one primitive data type to another type.

In Java, there are two types of casting:

**Widening Casting** (automatically casting):

Widening casting is done automatically when passing a smaller size type to a larger size type.   
byte -> short -> char -> int -> long -> float -> double

**Narrowing Casting** (manually casting):

Narrowing casting must be done manually by placing the type in parentheses in front of the value. This casting is generally converting a larger type to a smaller size type.  
double -> float -> long -> int -> char -> short -> byte

Write a program for widening casting between int and double?

public class Main {

public static void main(String[] args) {

int myInt = 9;

double myDouble = myInt; // Automatic casting: int to double

System.out.println(myInt);

System.out.println(myDouble);

}

}

Output:

9  
9.0

Write a program for narrowing casting between double and int?

public class Main {

public static void main(String[] args) {

double myDouble = 9.78d;

int myInt = (int) myDouble; // Explicit casting: double to int

System.out.println(myDouble);

System.out.println(myInt);

}

}

Output:

9.78  
9

Write a program to find ASCII value of any character?

public class Main {

public static void main(String[] args) {

char ch = 'A';

int asciiValue = (int) ch; // Convert character to ASCII value

System.out.println(ch + " -> " + asciiValue);

}

}

Output:

A->65

## **Java Wrapper Classes**

Wrapper classes provide a way to use primitive data types (int, boolean, etc..) as objects.

Sometimes we must use wrapper classes to work with Collection objects, such as ArrayList etc, because the list can store objects only. primitive types cannot be supported there.

The table below shows the primitive type and the equivalent wrapper class:

|  |  |
| --- | --- |
| **Primitive Data Type** | **Wrapper Class** |
| byte | Byte |
| short | Short |
| int | Integer |
| long | Long |
| float | Float |
| double | Double |
| boolean | Boolean |
| char | Character |

Write a program to use wrapper classes and display their values?

public class Main {

public static void main(String[] args) {

Integer myInt = 5;

Double myDouble = 5.99;

Character myChar = 'A';

System.out.println(myInt);

System.out.println(myDouble);

System.out.println(myChar);

}

}

Output

5  
5.99  
A

Write a program to change primitive variable as object?

public class Main {

public static void main(String[] args) {

int x = 5;

double y = 5.99;

char z = 'A';

String st = "45";

Integer p = new Integer(x);

Double q = new Double(y);

Character r = new Character(z);

Integer s = new Integer(st);

System.out.println(p);

System.out.println(q);

System.out.println(r);

System.out.println(s);

}

}

Output:

5

5.99

A

45

Write a program to convert object to primitive type?

public class Main {

public static void main(String[] args) {

Integer myInt = 5;

Double myDouble = 5.99;

Character myChar = 'A';

int x = myInt.intValue();

double y = myDouble.doubleValue();

char z = myChar.charValue();

System.out.println(x);

System.out.println(y);

System.out.println(z);

}

}

Output

5

5.99

A

Write a program to find sizes of each data type in java?

class SizePrimitiveTypes

{

public static void main (String[] args)

{

System.out.println("Size of byte: " + (Byte.SIZE/8) + " bytes.");

System.out.println("Size of short: " + (Short.SIZE/8) + " bytes.");

System.out.println("Size of int: " + (Integer.SIZE/8) + " bytes.");

System.out.println("Size of long: " + (Long.SIZE/8) + " bytes.");

System.out.println("Size of char: " + (Character.SIZE/8) + " bytes.");

System.out.println("Size of float: " + (Float.SIZE/8) + " bytes.");

System.out.println("Size of double: " + (Double.SIZE/8) + " bytes.");

}

}

Output:

Size of byte: 1 bytes.

Size of short: 2 bytes.

Size of int: 4 bytes.

Size of long: 8 bytes.

Size of char: 2 bytes.

Size of float: 4 bytes.

Size of double: 8 bytes.

Write a program to find binary value of number?

public class Main {

public static void main(String[] args) {

int x = 23;

System.out.println("Binary value of "+x+" is : "+Integer.toBinaryString(x));

}

}

Output:

Binary value of 23 is : 10111

Write a program that take a number and convert that number to string , binary, hexa and octa strings?

public class Main {

public static void main(String[] args) {

int x = 23;

System.out.println("X value is = " + Integer.toString(x));

System.out.println("Binary value of "+x+" is : "+Integer.toBinaryString(x));

System.out.println("Octa value of "+x+" is : " + Integer.toOctalString(x));

System.out.println("Hexa value of "+x+" is : " + Integer.toHexString(x));

}

}

Output:

X value is = 23

Binary value of 23 is : 10111

Octa value of 23 is : 27

Hexa value of 23 is : 17

# Autoboxing and Unboxing:

The automatic conversion of primitive data types into its equivalent Wrapper type is known as boxing.

The automatic conversion of wrapper class type into corresponding primitive type, is known as Unboxing

This is the new feature of Java5. So java programmer doesn't need to write the conversion code.

## Advantage of Autoboxing and Unboxing:

|  |
| --- |
| No need of conversion between primitives and Wrappers manually so less coding is required. |

Example for Boxing: assigning primitive value to wrapper class object.

class BoxingExample1{

  public static void main(String args[]){

    int a=50;

        Integer a2=new Integer(a);//Boxing

        Integer a3=5;//Boxing

        System.out.println(a2+" "+a3);

 }

}

Example for unboxing: assigning wrapper class object to primitive variables.

class UnboxingExample1{

  public static void main(String args[]){

    Integer i=new Integer(50);

        int a=i;

        System.out.println(a);

 }

}

What is parsing in Java?

In Java, parsing refers to the process of converting data from one data type or format into another. It is commonly used to convert textual data (such as strings) into numerical types (like integers or doubles) or to parse structured data (like JSON or XML) into Java objects.

Write a program parse string value to integer, double and float?

public class Main {

public static void main(String[] args) {

String numberStr = "123";

int x = Integer.parseInt(numberStr);

float y = Float.parseFloat(numberStr);

double z = Double.parseDouble(numberStr);

System.out.println(x);

System.out.println(y);

System.out.println(z);

}

}

Output:

123

123.0

123.0

Write a program to use sum, min and max methods in Integer wrapper class?

public class Main {

public static void main(String[] args) {

System.out.println("Sum is " + Integer.sum(2,4));

System.out.println("Biggest in 5 and 7 is " + Integer.max(5,7));

System.out.println("Smallest in 5 and 7 is " + Integer.min(5,7));

}

}

Output:

Sum is 6

Biggest in 5 and 7 is 7

Smallest in 5 and 7 is 5

## **Java Operators**

Operators are used to perform operations on variables and values.

Java divides the operators into the following groups:

* Arithmetic operators
* Assignment operators
* Comparison operators
* Logical operators
* Bitwise operators

## **Arithmetic Operators**

Arithmetic operators are used to perform common mathematical operations.

|  |  |  |  |
| --- | --- | --- | --- |
| **Operator** | **Name** | **Description** | **Example** |
| + | Addition | Adds together two values | x + y |
| - | Subtraction | Subtracts one value from another | x - y |
| \* | Multiplication | Multiplies two values | x \* y |
| / | Division | Divides one value by another | x / y |
| % | Modulus | Returns the division remainder | x % y |
| ++ | Increment | Increases the value of a variable by 1 | ++x |
| -- | Decrement | Decreases the value of a variable by 1 | --x |

## **Java Assignment Operators**

Assignment operators are used to assign values to variables.

A list of all assignment operators:

|  |  |  |
| --- | --- | --- |
| **Operator** | **Example** | **Same As** |
| = | x = 5 | x = 5 |
| += | x += 3 | x = x + 3 |
| -= | x -= 3 | x = x - 3 |
| \*= | x \*= 3 | x = x \* 3 |
| /= | x /= 3 | x = x / 3 |
| %= | x %= 3 | x = x % 3 |
| &= | x &= 3 | x = x & 3 |
| |= | x |= 3 | x = x | 3 |
| ^= | x ^= 3 | x = x ^ 3 |
| >>= | x >>= 3 | x = x >> 3 |
| <<= | x <<= 3 | x = x << 3 |

## **Java Comparison Operators**

Comparison operators are used to compare two values (or variables).

The return value of a comparison is either true or false.

A list of all comparison operators:

|  |  |  |
| --- | --- | --- |
| **Operator** | **Name** | **Example** |
| == | Equal to | x == y |
| != | Not equal | x != y |
| > | Greater than | x > y |
| < | Less than | x < y |
| >= | Greater than or equal to | x >= y |
| <= | Less than or equal to | x <= y |

## **Java Logical Operators**

You can also test for true or false values with logical operators.

Logical operators are used to determine the logic between variables or values:

|  |  |  |  |
| --- | --- | --- | --- |
| **Operator** | **Name** | **Description** | **Example** |
| && | Logical and | Returns true if both statements are true | x < 5 &&  x < 10 |
| || | Logical or | Returns true if one of the statements is true | x < 5 || x < 4 |
| ! | Logical not | Reverse the result, returns false if the result is true | !(x < 5 && x < 10) |

Programs for operators concept you can try in bellow link:

https://www.w3schools.com/java/java\_operators.asp