**Java Packages**

**Java.lang:** it contains useful classes, interfaces like wrapper classes, Strings, Threads etc.

**Java.io:** it handles files and input-output related tasks.

**Java.util:** it contains useful classes, interfaces like Stack, LinkedList, List etc.

**Java.awt:** it helps to develop GUI, it consists of an important sub package java.awt.event to handle the events for GUI elements

**Java.swing:** it is also used for developing GUI. it is an extension of the java.awt package.

**Java.net:** it is used for developing Client-Server programming. This is an important package to develop web programs like a web browser or web server.

**Java.applet:** applets are the programs which come from a server into a client and get executed on the client machine.

**Java.sql:** this package helps us to connect to the database like Oracle.

**Java datatypes**

In Java the datatypes are divided into two types:

**\*Primary datatypes:** byte, short, int, long, float, double, char, boolean.

**\*Non-primary datatypes:** Strings, Arrays, Classes.

**\*imp:** Finding the range of a **byte, short, int, long** by using the formula: -

**pow(2,n-1) to pow(2,n-1)-1**

**Memory based table**

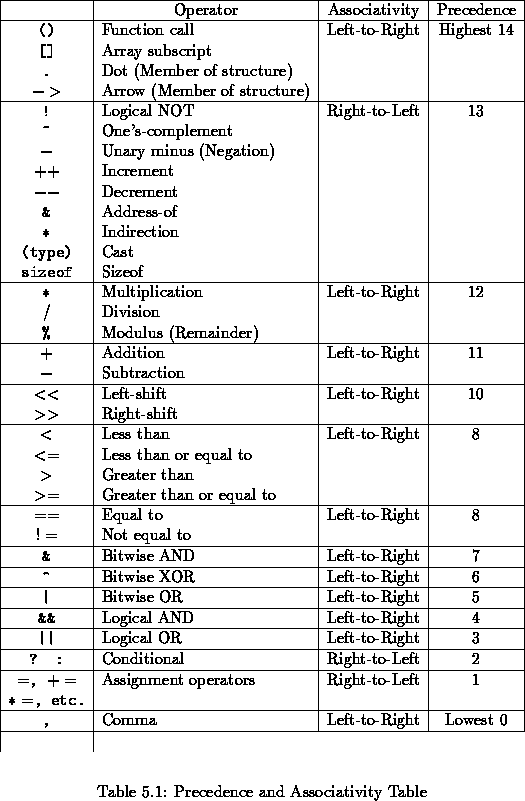
|  |  |  |
| --- | --- | --- |
| **Data Type** | **Size** | **Description** |
| byte | 1 byte | Stores whole numbers from -128 to 127 |
| short | 2 bytes | Stores whole numbers from -32,768 to 32,767 |
| int | 4 bytes | Stores whole numbers from -2,147,483.648 to 2,147,483,647 |
| long | 8 bytes | Stores whole numbers from -9,223.372,036.854,775.808 to 9,223.372,036,854,775,808 |
| float | 4 bytes | Stores fractional numbers from 3.4e−038 to 3.4e+038. Sufficient for storing 6 to 7 decimal digits |
| double | 8 bytes | Stores fractional numbers from 1.7e−308 to 1.7e+038. Sufficient for storing 15 decimal digits |
| boolean | 1 byte | Stores true or false values |
| char | 2 bytes | Stores a single character/letter |

**Operators**

**Types of Operators:-**

**Arithmetic Operators:** +, -, \*, /, %, ++, --.

**Relational Operators:** ==, !=, >, < , >=, <=.  
**Bitwise Operators:**&, |, ^(xor), ~(bitwise compliment), <<, >>, >>>(zero fill rightshift)  
**Logical Operators:**&&, ||,!,  
**Assignment Operators:** ==, +=, -= etc  
**Misc Operators:** ? : (ternary operator)



**Exercise**

**Exercise-1:**

int x=10;

x++; **//pre increment**

sopln(x);

**Description:** Here x++ is pre-increment so x=10 becomes 11.

**Exercise-2:**

int a=10,b=20,c=30;

b=a;c=b;

sopln(c);

**Ans:10**

**Description:** first a value is assign to the b then b value is assign to the c so c=10.

int a=b=c=10; //it gives compilation error.

sopln(c);

**Description:** b cannot be resolved to a variable  
 c cannot be resolved to a variable

int a,b,c;

a=b=c=10;

sopln(c);

**Ans:10**

**Description:**we are declare the variables then we intialize the vaues.

**Exercise-3:**

char ch='a';

ch++;

sopln(ch);

**Ans:b**

**Description:** where a is a character and while increasing the char value so it will be incremented as ASCII value.

**Exercise-4:**

double d=10.5;

d++; //it is pre-increment operation

sopln(d);

**Ans: 11.5**

**Description:**the value of d is incremented by 1.

**Exercise-5:**

boolean b=true;

b++; //compilation error

**Description:** Type mismatch: cannot convert from boolean to int

**Exercise-6:**

byte b=20;

byte b=b+1; // Type mismatch: cannot convert from int to byte

byte b=(byte)b+1 ;//syntax error

sopln(c); // c cannot be resolved to a variable

**Description:** can’t convert from int to byte in 2nd line.in the 3rd line we need perform typecasting properly.

**Exercise-7:**

byte a=10;

byte b=20;

byte c=a+b; //can’t convert from int to byte.

byte c=byte(a+b); //syntax error

sopln(c);

**Description:** here byte is having memory of 1byte but int have 4bytes so can’t convert from int to bye.

In the typecasting, there is the syntax error. the correct way is **byte c=(byte)(a+b)**

**Exercise-8:**

sopln(10/0); **//Exception in thread "main" java.lang.ArithmeticException: / by zero**

**Description:** divided by zero gives the exception.

sopln(10/0.0);

**Ans: Infinity**

**Description:** public static final float POSITIVE\_INFINITY  
A constant holding the positive infinity of type float. It is equal to the value returned by Float.intBitsToFloat(0x7f800000).

**Exercise-9:**

sopln('a' + 'b'); //ans 195

sopln('a' + 1); //ans 98

sopln('a' + 1.2); //ans 98.2

**Description:** there is no error because the characters are internally integers.

**Exercise-10:**

String a="ashok";

int b=10 , c=20 , d=30 ;

a=b+c+d ; //complie error:Type mismatch: cannot convert from int to String

a=a+b+c ;//ans:ashok1020

b=a+c+d ;//compile error:Type mismatch: cannot convert from String to int

**Exercise-11:**

sopln(10 < 10.5);//ans true

sopln('a' > 100.5);//ans false

sopln('b' > 'a');//ans true

sopln(true > false);//compile eror:**The operator > is undefined for the argument type(s) boolean, boolean**

**Exercise-12:**

sopln(10 == 20) ; //ans :false

sopln('a' == 'b' );//ans: false

sopln('a' == 97.0 ) // syntax error:semicolon missing

sopln(false == false) // syntax error:semicolon missing

**Exercise-13:**

Thread t1=new Thread( ) ;

Thread t2=new Thread( );

Thread t3=t1 ;

sopln(t1==t2); //ans false  
sopln(t1==t3); //ans true

**Description: t1 value:** Thread[Thread-0,5,main]

**t2 value:**Thread[Thread-1,5,main]

**t3 value:**Thread[Thread-0,5,main]

**Exercise-14:**

sopln(true&false);//ans false

sopln(true|false); //ans true

sopln(true^false);//ans true

**Description:** According to truth tables the above ans will come.

**Exercise-15:**

sopln(4&5); //ans 4

sopln(4|5); //ans 5

sopln(4^5); //ans 1

**Description:** According to truth tables the above ans will come.

**Exercise-16:**

sopln(~true); //The operator ~ is undefined for the argument type(s) boolean

sopln(~4); //ans -5

**Description:**   
The bitwise complement is equal to the two's complement of the value minus one. If two's complement arithmetic is used, then NOT x = -x − 1 . For unsigned integers, the bitwise complement of a number is the "mirror reflection" of the number across the half-way point of the unsigned integer's range

**Exercise-17:**

sopln(!false); //ans true

sopln(!4);//ans The operator ! is undefined for the argument type(s) int

**Exercise-18:**

int x=10 , y=15 ;

if(++x < 10 || ++y > 15) { //instead of || using &,&&, | operators

x++;

}

else {

y++;

}

sopln(x+"----"+y);

int x='a'; //**Duplicate local variable x**

sopln(x);

**Ans for ||:** 12……..16

**Ans for &&:** 11…….16

**Ans for &:** 11----17

**Ans for |:**12----16

**Description:** according to&& first statement will be evaluate if it is true then second will be evaluate in the if statements

According to bitwise operators, both statements will be evaluated.

**Exercise-19:**

int x=130;

byte b=(byte)x;

sopln(b);

**Ans: -126**

**Description:** byte range from -128 to 127 so after 127 it will count from -128.

**Exercise-20:**

int x=150;

short s=(short)x;

byte b=(byte)x;

sopln(s);//ans 150

sopln(b);//ans -106

**Description:** byte range from -128 to 127 so after 127 it will count from -128.

**Exercise-21:**

double d=130.456 ;

int x=(int)d ;

sopln(x); //ans 130

byte b=(byte)d ;

sopln(b); //ans -126

**Exercise-22:**

int x=(10>20)?30:((40>50)?60:70);

sopln(x);//ans 70

**Description:** by the ternary operation this will be performed

**Exercise-23:**

class OperatorsDemo {

public static void main(String[] args) {

sopln(m1(1)+m1(2)\*m1(3)/m1(4)\*m1(5)+m1(6));

}

public static int m1(int i) {

sopln(i);

return i;

}

}