Packages

1. java.lang: This package got primary classes and interfaces essential for Java program. It consists of wrapper classes (wrapper classes can be used to convert ordinary data into objects), Strings, Threads etc.

2. java.util: This package contains useful classes and interfaces like Stack, LinkedList, Arrays, ArrayList, List etc...

3. java.io: This package handles files and input output related tasks.

4. java.awt: This package helps to develop GUI. It consists of an important sub package java.awt.event to handle the events for GUI elements.

5. java.swing: This package helps to develop GUI like java.awt. In fact, this an extension to java.awt.

6. java.net: Client - server programming can be done using this package. This is a very important package to develop any web program like web browser or web server.

7. java.applet: Applets are programs which come from a server into a client and get executed on the client machine. This package was very well known to everyone before the development of servlets.

8. java.sql: This package helps us to connect to database like Oracle.

Data Types

There are 8 data types:

Data Type	Default Value	Default size
boolean	false	1 bit
char	'\u0000'	2 byte
byte	0	1 byte
short	0	2 byte
int	0	4 byte
long	0L	8 byte
float	0.0f	4 byte
double	0.0d	8 byte

Boolean data type : The Boolean data type is used to store only two possible values: true and false. This data type is used for simple flags that track true/false conditions. Ex boolean s= false;

Byte data type : It is an 8-bit signed two's complement integer. Its value-range lies between -128 to 127 (inclusive). Its minimum value is -128 and maximum value is 127. Its default value is 0. Ex byte b= -20;

Short data type : It is a 16-bit signed two's complement integer. Its value-range lies between -32,768 to 32,767 (inclusive). Its minimum value is -32,768 and maximum value is 32,767. Its default value is 0. Ex short s= 10;

Int data type : The int data type is a 32-bit signed two's complement integer. Its value-range lies between - 2,147,483,648 (-2^31) to 2,147,483,647 (2^31 -1) (inclusive). Its minimum value is - 2,147,483,648 and maximum value is 2,147,483,647. Its default value is 0.The int data type is generally used as a default data type for integral values unless if there is no problem about memory.
Ex int i = -200000

Long data type : The long data type is a 64-bit two's complement integer. Its value-range lies between -9,223,372,036,854,775,808(-2^63) to 9,223,372,036,854,775,807(2^63 -1)(inclusive). Its default value is 0. The long data type is used when you need a range of values more than those provided by int. Ex long a = 100000L;

Float data type : The float data type is a single-precision 32-bit IEEE 754 floating point.Its value range is unlimited. It is recommended to use a float if you need to save memory in large arrays of floating point numbers. The float data type should never be used for precise values, such as currency. Its default value is 0.0F.
Ex float f1 = 234.5f

Double data type : The double data type is a double-precision 64-bit IEEE 754 floating point. Its value range is unlimited. The double data type is generally used for decimal values just like float. The double data type also should never be used for precise values, such as currency. Its default value is 0.0d. Ex double d1 = 12.3

Char data type : The char data type is a single 16-bit Unicode character. Its value-range lies between '\u0000' (or 0) to '\uffff' (or 65,535 inclusive).The char data type is used to store characters. Ex char letterA= 'A'

Operators

Operator Type	Category	Precedence
Unary	postfix	*expr*++ *expr*--
	prefix	++*expr* --*expr* +*expr* -*expr* ~ !
Arithmetic	multiplicative	* / %
	additive	+ -
Shift	shift	<< >> >>>

Relational	comparison	< > <= >= instanceof
	equality	== !=
Bitwise	bitwise AND	&
	bitwise exclusive OR	^
	bitwise inclusive OR	|
Logical	logical AND	&&
	logical OR	||
Ternary	ternary	? :
Assignment	assignment	= += -= *= /= %= &= ^= |= <<= >>= >>>=

Example program for operators

```
public class Operator{

    public static void main(String []args){

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

        //Arithmetic operators

        System.out.println(a+b); //output is 30

        System.out.println(a-b);  //output is -10

        System.out.println(a*b); //output is 200

        System.out.println(a/b);  //output is 0

        System.out.println(a%b); //output is 10

        System.out.println(5*8/2+4-1*6/3);  //output is 22

        //shift operator

        System.out.println(10<<2); //output is 40

        System.out.println(10>>2); //output is 2

        //relational operator

        System.out.println(a>b); //output is false

        System.out.println(a<b); //output is true

        System.out.println(a==b); //output is false
```

```
//bitwise operator

System.out.println(a&b); //output is 0

System.out.println(a|b); //output is 30

System.out.println(a^b); //output is 30

//logical operator

System.out.println(a<b && a<c); //output is true

System.out.println(a<b||a<c); //output is true

//ternary operator

d=(a<b)?a:b;

System.out.println(d); //output is 10

//unary operator

System.out.println(a++); //output is 10

System.out.println(++b); //output is 21

//assignment operator

d+=10;

b=a;

System.out.println(d); //output is 20
```

```
        System.out.println(b); //output is 11

        }

}
```

Programs :

Exercise 1:

```
public class MyClass {

    public static void main(String args[]){

Int x=10;

      x++;      //the value of x incremented by 1

System.out.println(x);  }

}       // output is  11. No compilation error
```

Exercise 2:

```
int a=10,b=20,c=30;
b=a;c=b;       //  no error, a value copied into b and b value copied into c
sopln(c);      // output is 10
int a=b=c=10;    // compilation error, a=b=c=10 gives cannot find symbol error
sopln(c);
int a,b,c;
a=b=c=10;  // this is correct initialization
sopln(c);    //output is 10
```

Exercise 3:

```
char ch='a';
ch++;  // incrementing the ASCII value, no compilation error
sopln(ch); // output is character c because ch is a character type
```

Exercise-4:

```
double d=10.5;
d++;  // no compilation error
sopln(d); // output is 11.5
```

Exercise-5:

```
boolean  b=true;
b++;  //compilation error, error: bad operand type boolean for unary operator '++'
```

Exercise-6:

```
byte b=20;
byte b=b+1;  //compilation error, variable b is already defined
byte b=(byte)b+1 ; //error: incompatible types: possible lossy conversion from int to byte
sopln(c);  //error: cannot find  symbol
```

Exercise-7:

```
byte a=10;
byte b=20;
byte c=a+b;  // compilation error, type mismatch cannot convert integer to byte
byte c=byte(a+b); // compilation error, type mismatch cannot convert integer to byte
sopln(c);
```

Exercise-8:

```
sopln(10/0);  //Exception in thread "main" java.lang.ArithmeticException: / by zero
sopln(10/0.0); // no error, output is 'Infinity' because division by double zero is not throw
                    exception
```

Exercise-9:

```
sopln('a' + 'b'); // no compilation error, output is 195
sopln('a' + 1); //no compilation error,output is 98
sopln('a' + 1.2); //no compilation error,output is 98.2
```

Exercise-10:

```
String a="ashok";
int b=10 , c=20 , d=30 ;
a=b+c+d ; // compilation error, type mismatch: cannot convert int to String
a=a+b+c ; // no error, output for this is ashok1020
b=a+c+d ; // compilation error, type mismatch: cannot convert int to String
```

Exercise-11:

```
sopln(10 < 10.5); //output is true
sopln('a' > 100.5); // output is false
sopln('b'  >  'a'); //output is true
sopln(true > false); // compilation error, the operator > is undefined for the argument types
                        Boolean,boolean
```

Exercise-12:

```
sopln(10 == 20) ; // output is is false
sopln('a' == 'b' ); //output is false
sopln('a' == 97.0 ); //output is true
sopln(false == false) // output is true
```

Exercise-13:

```
Thread t1=new Thread( ) ;
Thread t2=new Thread( );
Thread t3=t1 ;
sopln(t1==t2);  // output is false
sopln(t1==t3); // output is true
```

Exercise-14:

```
sopln(true&false); //output is false
sopln(true|false);  //output is true
sopln(true^false); //output is true
```

Exercise-15:

```
sopln(4&5); //output is 4
sopln(4|5); //output is 5
sopln(4^5); //output is 1
```

Exercise-16:

sopln(~true); // compilation error, the operator ~ is undefined for the argument type boolean
sopln(~4); //output is -5

Exercise-17:

sopln(!false); ////output is true
sopln(!4); //compilation error, the operator ! is undefined for the argument type int

Exercise-18.1:

```
int x=10 , y=15 ;
if(++x < 10  ||  ++y > 15) {
 x++;
}
else {
          y++;
}
sopln(x+"----"+y);  //output is 12------16

int x='a';
sopln(x); output is 97
```

Exercise-18.2:

```
int x=10 , y=15 ;
if(++x < 10 &  ++y > 15) {
 x++;
}
else {
          y++;
}
sopln(x+"----"+y);  //output is 11------17
```

Exercise-18.3:

```
int x=10 , y=15 ;
if(++x < 10 &&  ++y > 15) {
 x++;
}
else {
          y++;
}
sopln(x+"----"+y);  //output is 11------16
```

Exercise-18.4:

```
int x=10 , y=15 ;
if(++x < 10 |  ++y > 15) {
 x++;
```

```
}
else {
        y++;
}
sopln(x+"----"+y);  //output is 12------16
```

Exercise-19:

```
int x=130;
byte b=(byte)x;  //no compilation error
sopln(b);   //output is -126
```

Exercise-20:

```
int x=150;
short s=(short)x; //no compilation error
byte b=(byte)x;  //no compilation error
sopln(s); //output is 150
sopln(b); //output is -106
```

Exercise-21:

```
double d=130.456 ;
int x=(int)d ; //no compilation error
sopln(x);  //output is 130
byte b=(byte)d ;  //no compilation error
sopln(b); //output is 126
```

Exercise-22:

```
int x=(10>20)?30:((40>50)?60:70);
sopln(x);  //output is 70
```

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;
        }
}
// output is 1 2 3 4 5 6 12
```