Understanding Wrapper Classes

- In java technology if we want to represent a group of objects in the form of an object then we have to use "Collection objects", like
  - Array List
  - Vector
  - Stack
- In java applications collections objects are able to allow only group of other objects, not primitive data directly.
- If we want to store primitive data in collection objects, first we need to **convert the primitive data in object form** then we have to store, that object data in collection objects.

- Java Technology has provided the following '8' number of Wrapper classes w.r.t to '8' number of primitive data types.

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

Points to remember...

- All most all wrapper classes define 2 constructors one can take corresponding primitive as argument and the other can take String as argument.

(except Character)

- Character class defines only one constructor which can take char primitive as argument there is no String argument constructor.
- If we are passing String as an argument in Boolean wrapper class then :

---> If the argument is true then the result also will be true irrespective of the data and case sensitiveness

---> If the argument is false then the result also will be false irrespective of the data and case sensitiveness

---> Other than true/false any other data will give you the result as false.

Wrapper classes are present in java.lang package

The parse methods are present in wrapper classes.

Wrapper classes are nothing but

Integer Class

Byte Class

Short Class

Long Class

Float Class

Double Class

Character Class

Boolean Class

parse method is going to convert string value to other data types except character

for example,

String s=”10”; Should be number

If String s=”JAVA”; for this, we will get C.E for parse method

Parse methods are static methods we can call them by using class names (Integer.parseint(Sting value)).

There is no parse method for character

parseInt (Sting value) -------Integer Class (String value to int)

parseByte (Sting value) -------Byte Class (String value to byte)

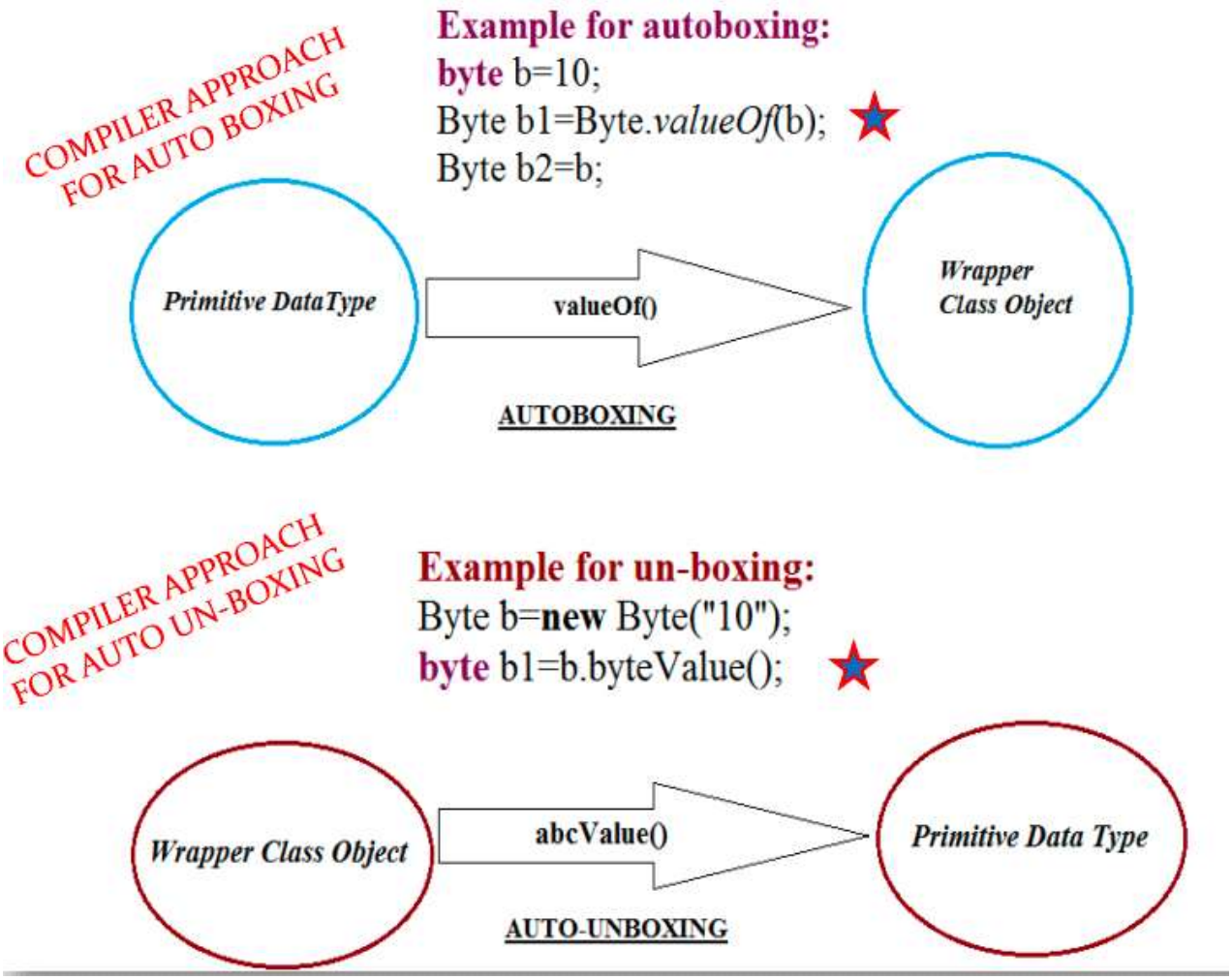
parseShort (Sting value) ---Short Class (String value to Short)

parseLong (Sting value) ----Long Class (String value to Long)

*parseFloat (*Sting value*) ----Float Class (String value to Float)*

*parseDouble (*Sting value*) ----Double Class (String value to Double)*

*parseBoolean (*Sting value*) -----Boolean Class (String value to Boolean)*



Auto boxing and auto un- boxing are automatically done by the compiler

```java
package com.pack1;

public class ClassA
{
    void meth1()
    {
        System.out.println("Performing Auto-Boxing");

        int i=10;
        Integer ival1=i; // 1st way
        Integer ival2=new Integer(i); // 2nd way
        Integer ival3=Integer.valueOf(i); // 3rd way

        System.out.println("\nint PDT i : "+i);
        System.out.println("Integer WCO ival1 : "+ival1);
        System.out.println("Integer WCO ival2 : "+ival2);
        System.out.println("Integer WCO ival3 : "+ival3);

        boolean flag=true;
        Boolean booleanval=Boolean.valueOf(flag);
        System.out.println("\nboolean PDT flag : "+flag);
        System.out.println("Boolean WCO booleanval : "+booleanval);


        float f=10.9f;
        Float fval=new Float(f);
        System.out.println("\nfloat PDT f : "+f);
        System.out.println("Float WCO fval : "+fval);
    }
    void meth2()
    {
        System.out.println("Implementing Auto-Unboxing");

        Integer ival=new Integer(500);
        int i1=ival; // 1st way
        int i2=ival.intValue();// 2nd way
        System.out.println("\nInteger WCO ival : "+ival);
        System.out.println("int PDT i1 : "+i1);
        System.out.println("int PDT i2 : "+i2);

```

```
40          Byte bval=new Byte((byte)10);
41          byte b1=bval.byteValue();
42          System.out.println("\nByte WCO bval : "+bval);
43          System.out.println("byte PDT b1 : "+b1);
44      }
45      void meth3()
46      {
47          System.out.println("meth3() called");
48          String s1="10";
49          //String s1="Java";
50          //int i=s1+90;// C.E because we cant perform addition operation with String & int
51          int i=Integer.parseInt(s1)+90;
52          System.out.println("i value : "+i);
53      /*
54          byte b=Byte.parseByte(s1);
55          System.out.println("b value : "+b);
56
57          short s=Short.parseShort(s1);
58          System.out.println("s value : "+s);
59
60          long l=Long.parseLong(s1);
61          System.out.println("l value : "+l);
62
63          float f=Float.parseFloat(s1);
64          System.out.println("f value : "+f);
65
66          double d=Double.parseDouble(s1);
67          System.out.println("d value : "+d);
68
69          //Character.parseCharacter(s1);// We will be getting C.E
70
71          boolean flag=Boolean.parseBoolean(s1);
72          System.out.println("flag value : "+flag);
73      */
74          System.out.println("Byte range is : "+ Byte.MIN_VALUE+" to "+Byte.MAX_VALUE);
75      }
76      public static void main(String[] args)
77      {
78          ClassA aobj=new ClassA();
79          //aobj.meth1();
80          //aobj.meth2();
81          aobj.meth3();
82      }
83  }
```

Assignment

1)Creating a Class, writing methods(Parameterized & Non-parameterized) with return types & without return types.
2) we have used the values which were returned by the methods
3) Object Class
    ===> hashCode()
    ===> equals()
    ===> getClass()
    ===> toString()
    ===> finalize()
4) Constructors(Parameterized & Default): Keypoints
5) Operators
    ===> Increment & Decrement
    ===> Arithematic
    ===> Relational
    ===> Logical

6) Static
    ===> Variable
    ===> Methods
    ===> Blocks
7) Variable
    ===> Instance
    ===> static
    ===> local
8) TypeCasting
    ===> Implict
    ===> Explicit
9) Wrapper
    ===> AutoBoxing
    ===> Auto Unboxing