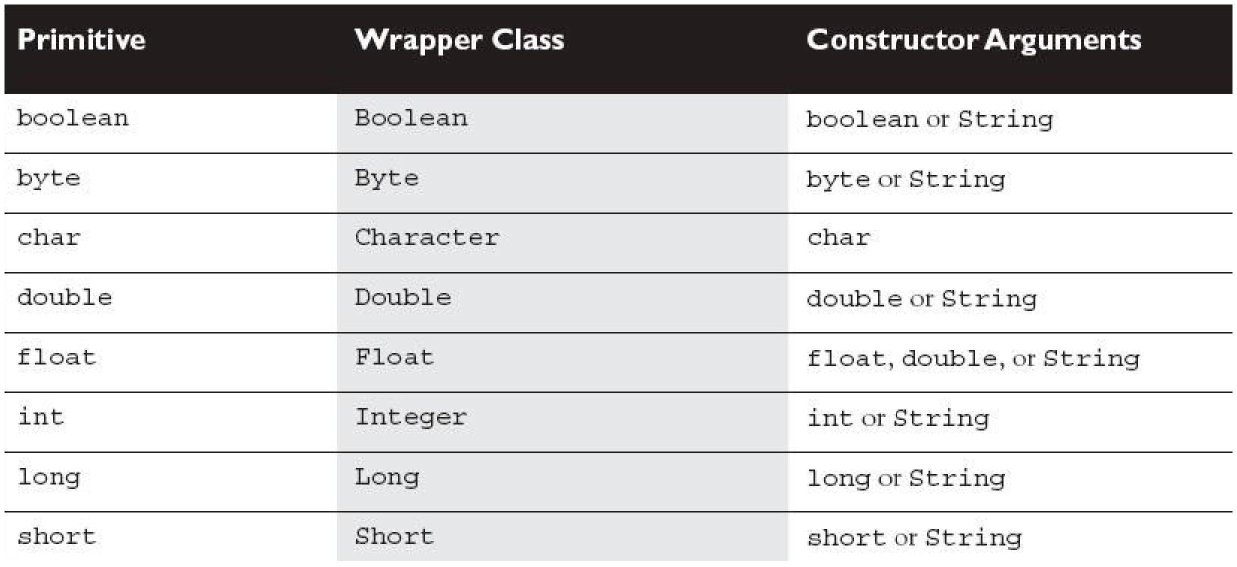
Topic: Wrapper Classes.
OOP concepts involved: Class Instantiation and Declaration, Class Members, Polymorphism, Constructor Methods, Class Modifiers, Interfaces, Objects, Getters and Setters.
Programming generic concepts involved: Functions, Data Types, Variables, Access Modifiers.

➢ Theoric introduction

WRAPPER CLASSES

A **Wrapper class** is a class whose object contains a primitive data type. This wrapper class wraps around the data type and gives it an object appearance, hence the name. In other words we can convert a primitive data type into an object.

Each primitive data type has a wrapper class attached to it, most of the times its named the same as the primitive data type but with the a capital first letter.



Primitive	Wrapper Class	Constructor Arguments
boolean	Boolean	boolean or String
byte	Byte	byte or String
char	Character	char
double	Double	double or String
float	Float	float, double, or String
int	Integer	int or String
long	Long	long or String
short	Short	short or String

WHY SHOULD I USE WRAPPER CLASSES?

Wrapper classes have multiple utilities, most of which might not seem obvious at first:

- ❏ They convert primitive data types into Objects, this is especially useful if we need to modify the arguments passed into a method (Since java passes primitive types by value)
- ❏ Allows access to object related packages like java.util
- ❏ Data structures in the Collection Framework, such as Arraylist and Vector, only store objects and not primitive types
- ❏ An object is needed to support synchronization in multithreading

AUTOBOXING AND UNBOXING

Now that we have talked about the uses and importance of Wrapper classes, how do we use them? Java introduced two automatic conversions called Autoboxing and Unboxing

Autoboxing is the automatic conversion of primitive types into their corresponding wrapper class. For example, conversion of int to Integer, long to Long, etc. This is done using the form below.

```
[primitive data type] pdt_identifier = value;
[Wrapper class] wc_identifier = pdt_identifier;

char ch = ‘a’;
Character wc = ch;
```

Unboxing is the reverse process of autoboxing. Automatically converting a wrapper class back into is corresponding primitive data type. For Example, Integer to int, Long to long, etc.

```
[Wrapper class] wc_identifier = value;
[primitive data type] pdt_identifier = wc_identifier;

Character wc = ‘a’;
char ch = wc;
```

➢ Statement

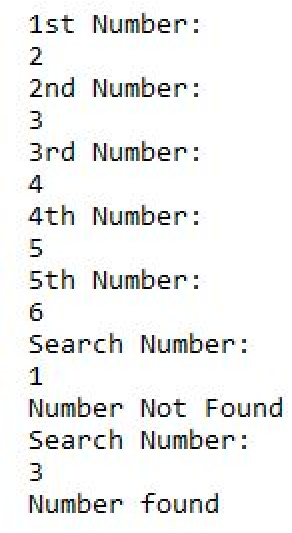
Create a program that captures 5 primitive data integers and uses Wrapper Classes to store them in an List object. After that the program should ask for two numbers and see if they are contained in the list.

➢ Program Code

Main.java

```java
import java.util.*;

public class Main{

        public static void main(String[] args) {
                ArrayList<Object> list = new ArrayList<Object>();
                Scanner sc = new Scanner(System.in);

                System.out.println("1st Number: ");
                list.add(Integer.parseInt(sc.nextLine()));
                System.out.println("2nd Number: ");
                list.add(Integer.parseInt(sc.nextLine()));
                System.out.println("3rd Number: ");
                list.add(Integer.parseInt(sc.nextLine()));
                System.out.println("4th Number: ");
                list.add(Integer.parseInt(sc.nextLine()));
                System.out.println("5th Number: ");
                list.add(Integer.parseInt(sc.nextLine()));

                System.out.println("Search Number: ");
                if(list.contains(Integer.parseInt(sc.nextLine())))
                        System.out.println("Number found");
                else
                        System.out.println("Number Not Found");

                System.out.println("Search Number: ");
                if(list.contains(Integer.parseInt(sc.nextLine())))
                        System.out.println("Number found");
                else
                        System.out.println("Number Not Found");
        }
}
```

➢ Program execution



```
1st Number:
2
2nd Number:
3
3rd Number:
4
4th Number:
5
5th Number:
6
Search Number:
1
Number Not Found
Search Number:
3
Number found
```

By making them wrapper classes instead of primitive data types, it allowed us the use of an Arraylist, which helped us a lot when searching and storing the numbers. Since the ArrayList can only store Objects, it would have been impossible to do this simple example without Wrapper Classes.

➢ Conclusions

Wrapper classes are a great tool in java. Since the language is Object oriented, most of the classes and methods require an object to work with. Wrapper classes serve the function of converting primitive data (the only data that's not object oriented), into objects.