Introduction to Wrapper class:

Let us first understand why wrapper class?

Java is an impure object-oriented programming language because it supports **primitive data type**.

You may be wondering; does it really matter if java is pure or impure object oriented.

To understand this, you must know an important fact about primitive data types which is Primitive data types are not treated as objects in java. ~~object~~

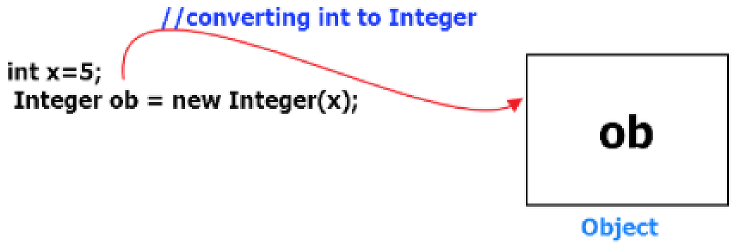
However, 100% pure object-oriented programs can be written in java using **Wrapper Class.**

Now you understood why wrapper class let us understand what is Wrapper class.

What is Wrapper class?

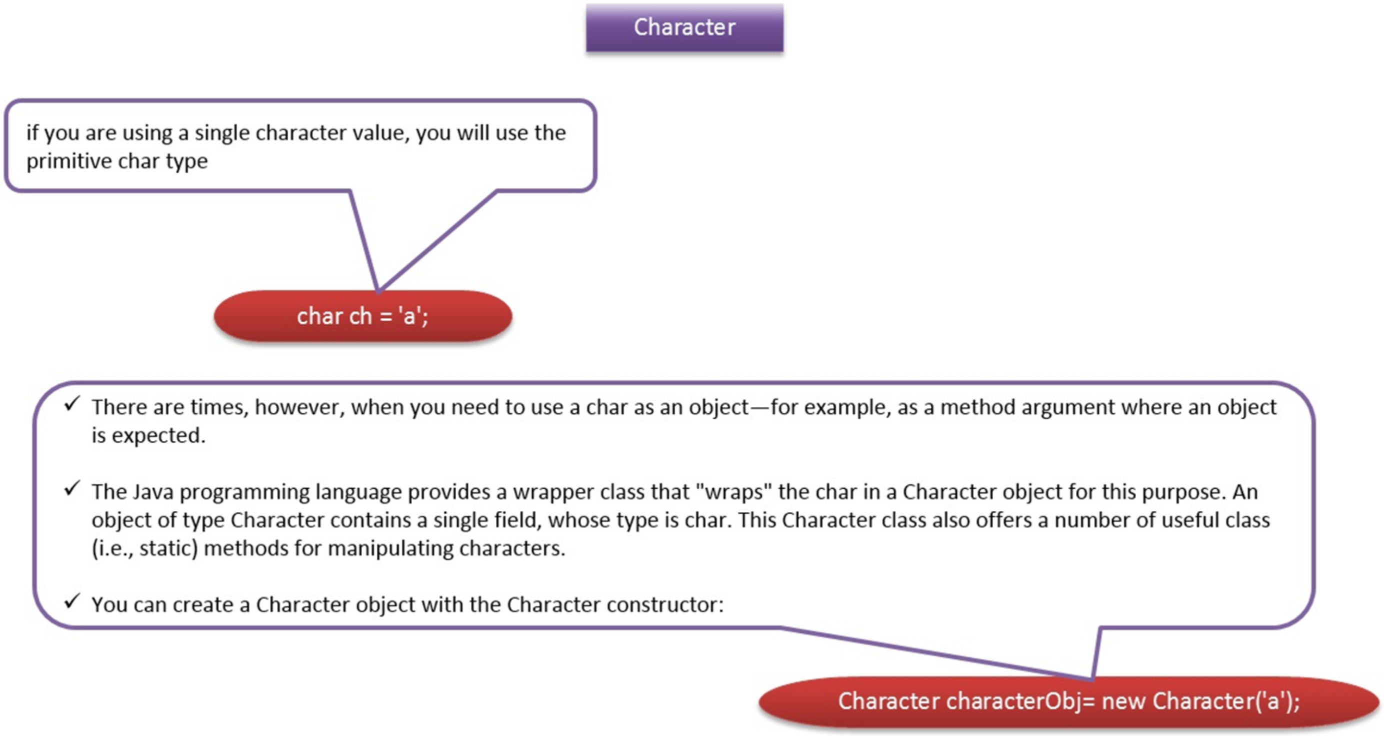
The **wrapper class in Java is a class** provides the mechanism ***to convert primitive data types into object and object into primitive*** **data type**.

Well how if you wonder, consider the below example:



```
int x=5;
 Integer ob = new Integer(x);
```

In the above example **int** is a **primitive data** type and **Integer** is a **wrapper class**. Let us have a look at one more example of character data type.



Character

if you are using a single character value, you will use the primitive char type

char ch = 'a';

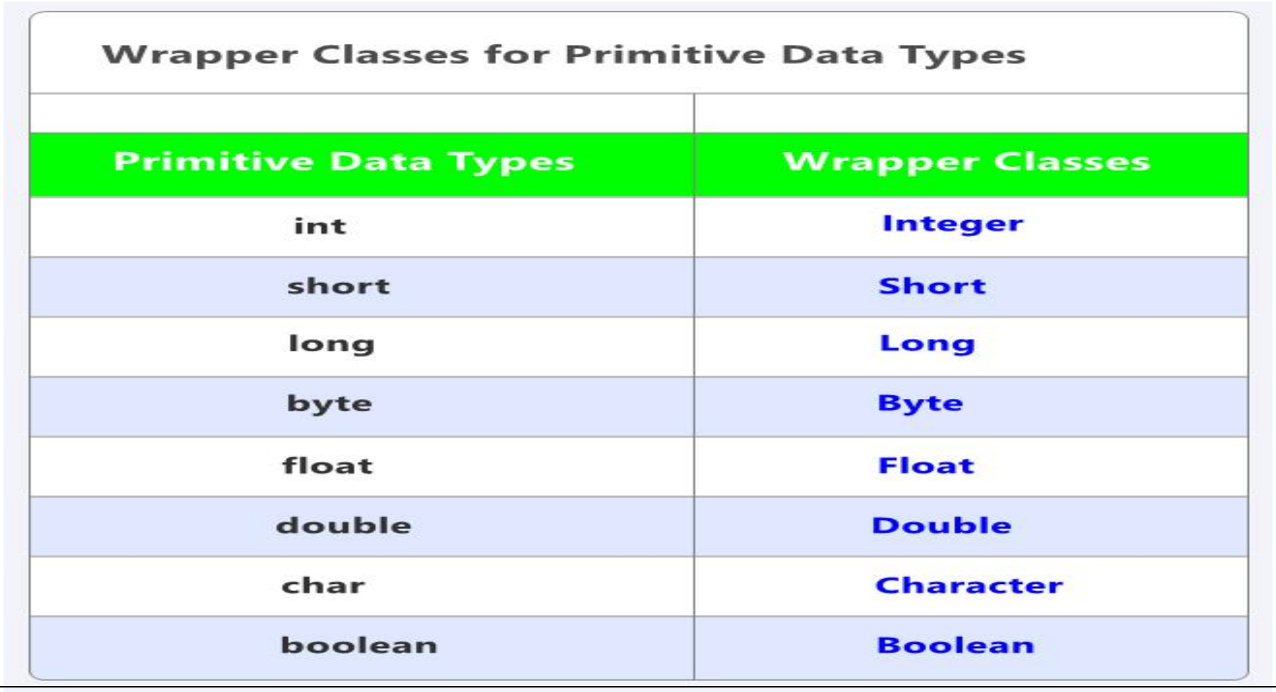
- ✓ There are times, however, when you need to use a char as an object—for example, as a method argument where an object is expected.
- ✓ The Java programming language provides a wrapper class that "wraps" the char in a Character object for this purpose. An object of type Character contains a single field, whose type is char. This Character class also offers a number of useful class (i.e., static) methods for manipulating characters.
- ✓ You can create a Character object with the Character constructor:

Character characterObj= new Character('a');

Advantage of Wrapper class: The program becomes pure object oriented.
Disadvantage of wrapper class: Execution speed decreases.

Advantage of Primitive data type: Faster in execution.
Disadvantage of Primitive data type: The program becomes impure object oriented.

Different wrapper classes for different primitive data types is given below:

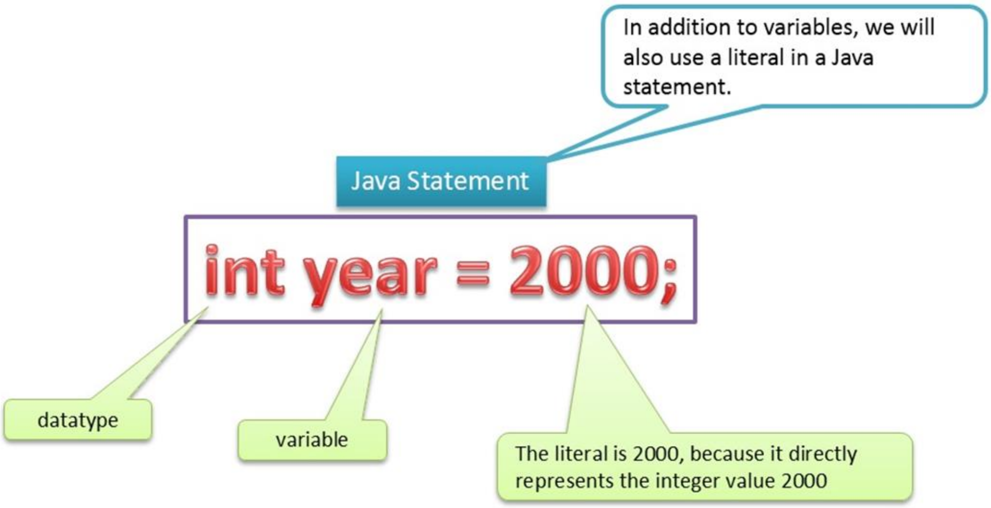


Wrapper Classes for Primitive Data Types

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

Literals in java:

Any constant value which can be assigned to the variable is called as literal. Consider the example shown below:


In addition to variables, we will also use a literal in a Java statement.
Java Statement
int year = 2000;
datatype
variable
The literal is 2000, because it directly represents the integer value 2000

Let us have a look at some valid and invalid syntaxes of variable names and literals.

Variable Names:

The symbols that are allowed in variable names are → ***_ and $***

Valid Syntax: int rooman_ = 99;
int _rooman_ = 99;
int _r_ooman_ = 99;
int roman$ = 99;
int $rooman_ = 99;
int r_o$$oman_ = 99;

Invalid Syntax: int &rooman_ = 99;
int _roo%man$ = 99;
int roo man = 99;
int roo " man = 99;

Literals:

The only symbol that is allowed in literal is ⇨ **"_"(underscore)**

Valid Syntax:

```
int rooman_ = 9_9;
int _rooman_ = 9__9;
int _r_ooman_ = 9____9;
```

Invalid Syntax:

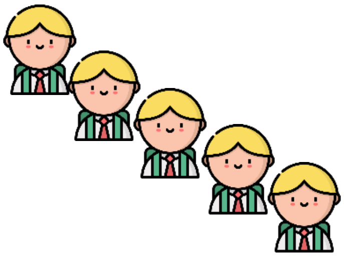
```
int rooman_ = 99_;
int rooman = _99;
float rooman = 99._9f;
float rooman = 99.9_9f;
float rooman = 99.99_f;
```

Introduction to Arrays:

In Java array is an object which contains elements of a similar data type.

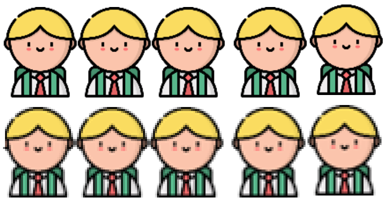
After getting to know the definition of array, let us now understand why array-based approach was introduced.
To understand this, let us first understand the limitations of variable approach by considering three different cases:

Case i) Write code to store the ages of 5 students.



```
int a;
int b;
int c;
int d;
int e;
```

Caseii) Write code to store the ages of 10 students.



```
int a;
int b;
int c;
int d;
int e;
int f;
int g;
int h;
int i;
int j;
```

Caseiii) Write code to store the ages of 20 students.

```
int a;
int b;
int c;
int d;
int e;
int f;
int g;
int h;
int i;
int j;
int h;
int i;
int j;
int k;
int l;
int m;
int n;
int o;
int p;
int q;
int r;
int s;
int t;
```

After considering three different cases, if I consider case iii and ask you which variable stores the marks of 18th student, you'll now have to check each line to answer this question which is one of the limitations of variable approach.

The variable approach has two limitations:

1) ***Creation is difficult.***

2) ***Remembering multiple names and accessing them is difficult.***

Due to these limitations, array-based approach was invented.

Arrays are of two types:

1)Regular array.

2)Jagged array.

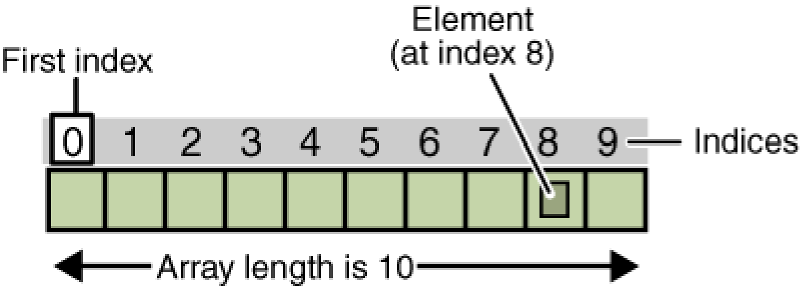
1)Regular array: Regular array is further divided into 1-Dimensional, 2-Dimensional, 3-Dimensional array and so on.

1-Dimensional regular array/Rectangular array.

Let us learn the creation of 1-D array using an example.

Ex: Create an array to store the ages of 10 students:

```
int a[  ] = new int[10];
```



2-Dimensional regular array/Rectangular array.

Let us learn the creation of 2-D array using an example.

Ex: Create an array to store the ages of students belonging to 2 classrooms with 5 students each.

```
int a[  ][  ] = new int[2][5];
a[1][0] = 5;
```

2 5

Rows Columns

	0	1	2	3	4
0	0	0	0	0	0
1	5	0	0	0	0

Array length is 10