

## Most used

**Print with new line** -> `System.out.println("Hello World");`

**Scanner object** -> `Scanner s = new Scanner(System.in);`

**Find the length (int)** -> `msg.length()`

**To lower/uppercase** -> `msg.toLowerCase()/toUpperCase()`

**Replace a string** -> `msg.replaceAll(String a, String b)`

**charAt(int index)** -> char value at the specified index

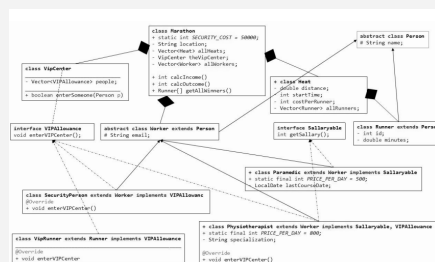
**equals(String/Object)** -> if strings values are the same

**toCharArray()** -> Convert string to character array

## Switch

```
switch (num) {
    case 1: doSomething();
    break;
    default: doThis();
    break;
}
```

## Q1



## bubble sort

```
public static void swap(int[] arr, int i, int j) {
    int temp = arr[i];
    arr[i] = arr[j];
    arr[j] = temp;
}

public static void sortAscending(int arr[]) {
    for (int i=arr.length-1; i > 0; i--) {
        for (int j=0; j < i; j++) {
            if (arr[j] > arr[j+1])
                swap(arr, j, j+1);
        }
    }
}
```

## binary search

```
public static int binarySearch(int a[], int val) {
    int low=0, high=a.length-1, middle;
    while (low <= high) {
        middle = (low + high)/2;
        if (val == a[middle])
            return middle;
        else if (val < a[middle])
            high = middle - 1;
        else
            low = middle + 1;
    }
    return -1;
}
```

## Exception Handling in Java

<b>ArithmeticException</b>	Errors because of an arithmetic expression.
<b>ZeroDivisionException</b>	Dividing a value with 0.
<b>ArrayIndexOutOfBoundsException</b>	No such index number.
<b>FileNotFoundException</b>	Accessing a file that does not exist.
<b>IOException</b>	Input or output error.
<b>NullPointerException</b>	Referencing a null object.
<b>NumberFormatException</b>	When typecasting does not work.
<b>StringIndexOutOfBoundsException</b>	When there is no index for a particular string.

## try.....catch Statement in Java

```
class Exception_Handling
{
    public static void main(String args[])
    {
        try
        {
            int x = 10/0;
        }
        catch (ArithmeticException e)
        {
            System.out.println("Division by zero");
        }
    }
}
```

## File Reading and Writing Process

```
// Create a file
File Obj = new File("file.txt");

// Write in the file
import java.io.*;
FileWriter write_file = new
FileWriter("file.txt");
write_file.write(" ") //
write in the file
write_file.close() //close
file

//Read from a File:
File read_file = new File("file.txt");
System.out.println(read_file.length());
```

### File Reading and Writing Process (cont)

```
> read_file.close();
```

```
import java.io.File --> Create or Open File --
```

```
> Read or Write in to file --> Close the file
```

### Inheritance

object of a parent class cannot be cast down to object of the child class

```
One one = new Two(); -> Two two = (Two) one;
```

Polymorphism for static methods does not work in the same way as instance methods.

In Java, a child class can make the method abstract which is inherited from the parent class.

If a reference of a parent class is pointing to the object of a child class, it can only call methods defined by the parent class.

"The method print() is undefined for the type One".

Unlike methods, constructors are not inherited by the child class and hence they cannot be overridden.

method needs a return type

Java constructors do not have return type.

error "The constructor One(String) is undefined".

### Interface

```
interface Sun{
    void rays();
}

class Mercury implements Sun{
    public void rays() {System.out.println("Mercury");}
}

class Earth implements Sun{
    public void rays() {System.out.println("Earth");}
}

class Sunlight{
    public static void main(String args[]){
        Sun e = new Earth();
        e.rays();
    }
}
```

An interface in Java helps to implement the concept of abstraction. In an interface class, we only have abstract methods and no method body. With interfaces, we can also achieve multiple inheritance in Java.

### Method Overloading

When a class has two or more methods with the same name but different parameters, it is method overloading.

#### Example of Overload:

```
public printer(String x){}
public printer(String x, String y){}
```

If the input is 2 string, it will go to the second method instead of first one.

**But you cannot overload by using the same input type sequence. For example:**

```
public printer(String x){}
public printer(String x, String y){} // conflict
public printer(String y, String x){} // conflict
```

Java will not allow this to be run, because it cannot determine the value.

### Override

When you have inherit some of the class from parents, but you want to do something different. In override feature, all the subclass/class object will use the newer method. To make sure JDK knows what you are doing, type @Override in front of the public name. If the override is unsuccessful, JDK will returns error.

#### Example of overridden helloWorld() method :

```
Class Student
public void helloWorld(){
    System.out.println("Hello");
}

Class GradStudent extends Student
@Override
public void helloWorld(){
    System.out.println("Hello World");
}
```

#### Rules of Overridden methods:

1. Access modifier priority can only be narrower or same as superclass
2. There is the same name method in superclass / libraries



By COOKIE1234

[cheatography.com/cookie1234/](https://cheatography.com/cookie1234/)

Not published yet.

Last updated 4th June, 2022.

Page 2 of 2.

Sponsored by [ApolloPad.com](https://apollopadd.com)

Everyone has a novel in them. Finish Yours!

<https://apollopadd.com>