:: double colon operator, this an act as reference to static methods, instance methods, constructor using new

```
//Acessing ArrayList in different ways
// First Approach
ArrayList<String> list = new ArrayList<>();
list.add("Admin01");
list.add("Admin02");
list.add("Admin03");
list.add("Admin04");
list.add("Admin05");
System.out.println(list); // [Admin01, Admin02, Admin03, Admin04, Admin05]
```

```
// Second Approach
// <Class name>::<method name
list.forEach(System.out::println);
Admin01
Admin02
Admin03
Admin04
Admin05
```

```
// Third Approach
// Using Iteration
Iterator<String> iterator = list.iterator();
while (iterator.hasNext()) {
System.out.println(iterator.next());
}
Admin01
Admin02
Admin03
Admin04
Admin05
```

```
// Fourth Approach
// Iterate over each element of arraylist using forEach() method
// parameter -> expression
list.forEach(s -> System.out.println(s));
```

```
Admin01
Admin02
Admin03
Admin04
Admin05
```

```
//Static Method
public class Eg2 {

private static void add() {
int a = 10;
int b = 20;
System.out.println(a+b);
}

public static void main(String[] args) {
//Direct Approach
Eg2.add(); // 30

//Method refrence Approach
//ClassName :: StaticMethodName
Thread t1= new Thread(Eg2 :: add);
t1.start(); // 30

//Lambda Approach
Thread t2 = new Thread(()-> add());
t2.start(); // 30
}
}
```

```java
//Instance Method
public class Eg3 {

public void add() {
int a = 10;
int b = 20;
System.out.println(a + b);
}

public static void main(String[] args) {
// Direct Approach
Eg3 e = new Eg3();
e.add(); // 30
// Method refrence Approach
// class Name :: methodName
Thread t1 = new Thread(new Eg3()::add);
t1.start(); // 30
// Lambda Approach
Eg3 ee = new Eg3();
Thread t2 = new Thread(() -> ee.add());
t2.start(); // 30
}
}
```

```
//Method reference to a constructor
@FunctionalInterface
interface StudentIn {
Student id(int id, String name);
}

class Student {

public Student(int id, String name) {
System.out.print(id + " " + name);
}
}
```

```
public class Eg4 {

public static void main(String[] args) {

// Method reference to a constructor
StudentIn ref = Student::new;
ref.id(101, "hari \n");
ref.id(102, "mahi \n");
ref.id(103, "bunny \n");
ref.id(104, "sunny");
}
}
```

```
101 hari
102 mahi
103 bunny
104 sunny
```