



## Lambda (λ) Expressions

- Lambda calculus is a big change in mathematical world which has been introduced in 1930. Because of benefits of Lambda calculus slowly this concepts started using in programming world. "LISP" is the first programming which uses Lambda Expression.
- ★ The other languages which uses lambda expressions are:
  - C#.Net
  - C Objective
  - **-** (
  - C++
  - Python
  - Ruby etc. and finally in Java also.
- \* The Main Objective of Lambda Expression is to bring benefits of functional programming into Java.

## What is Lambda Expression ( $\lambda$ ):

- Lambda Expression is just an anonymous (nameless) function. That means the function which doesn't have the name, return type and access modifiers.
- Lambda Expression also known as anonymous functions or closures.

- If the type of the parameter can be decided by compiler automatically based on the context then we can remove types also.
- The above Lambda expression we can rewrite as (a,b) → sop (a+b);





## **Ex: 3**

```
public String str(String str) {
    return str;
}

(String str) → return str;

(str) → str;
```

## **Conclusions:**

1) A lambda expression can have zero or more number of parameters (arguments).

```
Ex:

() → sop("hello");

(int a ) → sop(a);

(inta, int b) → return a+b;
```

2) Usually we can specify type of parameter. If the compiler expects the type based on the context then we can remove type. i.e., programmer is not required.

```
Ex:

(inta, int b) \rightarrow sop(a+b);

(a,b) \rightarrow sop(a+b);
```

- 3) If multiple parameters present then these parameters should be separated with comma (,).
- 4) If zero number of parameters available then we have to use empty parameter [like ()].

```
Ex: () \rightarrow sop("hello");
```

5) If only one parameter is available and if the compiler can expect the type then we can remove the type and parenthesis also.

```
(int a) \rightarrow sop(a);
(a) \rightarrow sop(a);
A \rightarrow sop(a);
```

- 6) Similar to method body lambda expression body also can contain multiple statements. If more than one statements present then we have to enclose inside within curly braces. If one statement present then curly braces are optional.
- 7) Once we write lambda expression we can call that expression just like a method, for this functional interfaces are required.