DATE
PAGE No.

Lambda Expression

1) What is lambda expression?

Ans) It is an expression which allows you to write more succinct code in Java 8. For Eg (a,b) → a+b is a lambda expression

which ~~means~~. is equal to the follow code

```
Public int Value (~~a~~ int a, int b) {
    return a+b;
}
```

It is also called an anonymous function because you are writing the code you write in function but without name.

2) Can you pass lambda expression ~~into a~~ to a method? when?

Ans) Yes, you can pass a lambda expression to a method provided it is expecting a functional interface. For example, if a method is accepting a Runnable, Comparable or Comparator then you can pass a lambda expression to it because all these are function interface

3) What is functional interface in java 8?

Ans) A functional interface in java 8 is an interface with a single abstract method. For Example, Comparator which have just one abstract method called compare() or Runnable which has just one abstract method called run(). These are many more general purpose functional interfaces introduced in JDK on java.util.function package. They are also annotated with @FunctionalInterface but that's optional.

4) What is the benefit of Lambda expression?

Ans) That now it's easier to pass a code block to a method. Earlier, the only way to do this was wrapping the code inside an Anonymous class, which requires a lot of boilerplate code.

5) Is it mandatory for a lambda expression to have parameters?

Ans) No, it is not mandatory to have parameter you can define a lambda expression without parameters as shown below.

```
() -> System.out.println("Hi");
```

you can pass this code to any method which accepts a functional interface.