1. What is the Lambda Expression in Java 8:

A lambda expression in Java 8 is a concise way to represent an anonymous function (a function without a name) that can be used to implement a method defined by a functional interface. It allows for functional programming in Java, making the code more readable and expressive.

Example:

```
(int a, int b) -> a + b;
```

2. Can You Pass Lambda Expressions to a Method? When:

Yes, you can pass lambda expressions to a method when the method parameter is of a functional interface type. A functional interface is an interface with a single abstract method, and a lambda expression provides the implementation for that method.

Example:

```
@FunctionalInterface
interface Operation {
    int execute(int a, int b);
}

public class LambdaExample {
    public static void main(String[] args) {
        Operation add = (a, b) -> a + b; // Lambda expression
        int result = performOperation(5, 3, add);
        System.out.println("Result: " + result); // Output: Result: 8
    }

    static int performOperation(int a, int b, Operation op) {
        return op.execute(a, b);
    }
}
```

3. What is the Functional Interface in Java 8:

A functional interface in Java 8 is an interface that has exactly one abstract method. It can have multiple default or static methods, but only one abstract method. The `@FunctionalInterface` annotation is used to mark an interface as functional, although it's optional.

Example:

```
@FunctionalInterface
interface Calculator {
    int calculate(int x, int y);
}

public class LambdaExample {
    public static void main(String[] args) {
        Calculator add = (a, b) -> a + b;
        System.out.println("Sum: " + add.calculate(10, 20)); // Output: Sum: 30
    }
}
```

4. Why Do We Use Lambda Expressions in Java:

- **Conciseness:** Lambdas allow you to write less code by removing boilerplate code such as anonymous class declarations.

- **Readability:** They make the code more readable and expressive.

- **Functional Programming:** Enables functional programming techniques in Java, such as passing functions as arguments, treating functions as first-class citizens, and improving code reusability.

- **Parallelism:** Lambda expressions are often used in conjunction with streams, which can be easily parallelized for better performance.

5. Is It Mandatory for a Lambda Expression to Have Parameters:

No, it is not mandatory for a lambda expression to have parameters. A lambda expression can have zero, one, or multiple parameters. If there are no parameters, empty parentheses `()` are used.

Examples:

- No Parameters: `() -> System.out.println("Hello, World!");`

- One Parameter: `name -> System.out.println("Hello, " + name);`

- Multiple Parameters: `(a, b) -> a + b;`