1. Why Lambda?

- Enables functional programming.

- Readable and concise code.

- Enables to write easy to use APIs and libraries.

- Enables support for parallel programming.

1. What is Lambda Expression?

It is like an anonymous function.

- without name

- without modifiers

- without return type

1. Difference between object-oriented programming and functional programming?

In oops a function/code blocks can’t exist on its own. It needs to be associated with a class. However, in functional programming function can exist without a class.

Public void greet(action){

action();

}

Lambda is out there to achieve this. We can pass the lambda function as value and execute the lambda.

1. What is Functional Interface?

An interface which contains **single abstract method** is called Function Interface. If you want to invoke Lambda expression, then function interface is required. Functional Interface job is to call the Lambda expression.

Eg : Runnable, Callable, Comparable, Comparator, ActionListener

1. Difference between Anonymous and Lambda?

Anonymous can be used for the interfaces which have more than 1 abstract methods. However, Lambda can be used only if the interface has only one abstract methods.

1. What are pre-defined functional interfaces in Java?

**Predicate**

interface Predicate<T>{

public boolean test(T t);

}

**Function**

interface Function<T,R>{

public R apply(T t);

}

**Consumer**

interface Consumer<T>{

public void accept(T t);

}

**Supplier**

interface Supplier<R>{

public R get();

}

**BiPredicate**

interface BiPredicate<T1,T2>{

public boolean test(T1 t1,T2 t2);

}

**BiFunction**

interface BiFunction<T,U,R>{

public R apply(T t,U u);

}

**BiConsumer**

interface BiConsumer<T,U>{

public void accept(T t, U u);

}

1. What is Method Reference?

Method Reference is used for code reusability. If the same block of code is being repeated, then it’s good to make a method and call it using the method reference.

ClassName::MethodName

ObjectName::MethodName

**Only restriction is that method arguments should match with the Functional Interface abstract method. It can have any access modifier or return type.**

1. What is Constructor Reference?

If the Functional Interface abstract method returns an object, then in that case we can use Contructor Reference.

ClassName::new

Test::new

1. What is Stream?

A sequence of elements supporting sequential and parallel aggregate operations. The following example illustrates an aggregate operation using Stream and IntStream:

int sum = widgets.stream()

.filter(w -> w.getColor() == RED)

.mapToInt(w -> w.getWeight())

.sum();

1. What is flatmap?
2. What is interface default method?

**Default Method** is also known as **Virtual Extension Method and Defender Method**.

**Without affecting the implementation classes if you want to add a method then we can use the default method**. In java 1.8, streams have been added to the List interface and it has been defined rather than just declaring it. By this approach, default method helps to provide the backward compatibility without breaking the implementation classes. Since List has many implementation classes it could have easily broke all the implementation classes, it has been just declared.

“default” should only be used inside the interface, there is a different meaning inside a class.

Object class method can’t be declared as default method inside the interface. Because object class by default is already available to all the classes in java.

There is a diamond problem associated with default method because it can lead to multiple inheritance problem.

1. What is interface static method?

If you have only static methods in a class then it’s better to go for interface which is less expensive in comparison to a class. Moreover, static methods are not related to object so it’s better to keep it inside an interface.