Assignment Questions 4

Q1.1. Write a program to show Interface Example in java?

ans:-

```
interface printable{
void print();
}
class A6 implements printable{
public void print(){System.out.println("Hello");}

public static void main(String args[]){
A6 obj = new A6();
obj.print();
 }
}
```

Q2.Write a program a Program with 2 concrete method and 2 abstract method in java ?

ans:-

```
abstract class AbstractExample {

   // Abstract method
   abstract void display();
```

```
    // Concrete method
    void show()
    {
        System.out.println("Concrete method of abstract class");
    }
}

// Subclass of abstract class
class SubClass extends AbstractExample {

    // Implementing the abstract method
    void display()
    {
        System.out.println("Abstract method implemented");
    }
}

/**
 * Main class
 */
public class AbstractClass{

    public static void main(String args[])
    {
```

```
        // Creating an object of the subclass
        SubClass obj = new SubClass();

        // Calling the abstract method
        obj.display();

        // Calling the concrete method
        obj.show();
    }
}
```

Q3.Write a program to show the use of functional interface in java?

ans:-

```
class Test {
        public static void main(String args[])
        {
                // create anonymous inner class object
                new Thread(new Runnable() {
                        @Override public void run()
                        {
                                System.out.println("New thread created");
                        }
```

```
            }).start();
        }
}
```

Q4.What is an interface in Java?

ans:-

In Java, an interface specifies the behavior of a class by providing an abstract type. As one of Java's core concepts, abstraction, polymorphism, and multiple inheritance are supported through this technology. Interfaces are used in Java to achieve abstraction.

Q5.What is the use of interface in Java?

ans:-

Provides communication − One of the uses of the interface is to provide communication. Through interface you can specify how you want the methods and fields of a particular type.

Multiple inheritance − Java doesn't support multiple inheritance, using interfaces you can achieve multiple inheritance .

Q6.What is the lambda expression of Java 8?

ans:- Lambda Expressions were added in Java 8. A lambda expression is a short block of code which takes in parameters and returns a value. Lambda expressions are similar to methods, but they do not need a name and they can be implemented right in the body of a method.

Q7.Can you pass lambda expressions to a method? When?

ans:-

lambda expression passed in a method that has an argument of type of functional interface. If we need to pass a lambda expression as an argument, the type of parameter receiving the lambda expression argument must be of a functional interface type.

```
interface TestInterface {
  boolean test(int a);
}
class Test {
  // lambda expression can be passed as first argument in the check() method
  static boolean check(TestInterface ti, int b) {
    return ti.test(b);
  }
}
public class LambdaExpressionPassMethodTest {
  public static void main(String arg[]) {
    // lambda expression
    boolean result = Test.check((x) -> (x%2) == 0, 10);
    System.out.println("The result is: "+ result);
  }
}
```

Q8.What is the functional interface in Java 8?

ans:-

A functional interface is an interface that consists of one abstract method. These interfaces can show only one functionality. Beyond Java 8, lambda expressions can be used to represent the instance of a functional interface. Functional Interfaces can contain any number of static and default methods. Consumer, Predicate, Function, Unary Operator, Binary Operator are some of the examples of predefined functional interfaces.

Q9.What is the benefit of lambda expressions in Java 8?

ans:-

Fewer Lines of Code − One of the most benefits of a lambda expression is to reduce the amount of code. We know that lambda expressions can be used only with a functional interface. For instance, Runnable is a functional interface, so we can easily apply lambda expressions.

Sequential and Parallel execution support by passing behavior as an argument in methods − By using Stream API in Java 8, the functions are passed to collection methods. Now it is the responsibility of collection for processing the elements either in a sequential or parallel manner.

Higher Efficiency − By using Stream API and lambda expressions, we can achieve higher efficiency (parallel execution) in case of bulk operations on collections. Also, lambda expression helps in achieving the internal iteration of collections rather than external iteration.

Q10.Is it mandatory for a lambda expression to have parameters?

ans:-

Lambda Expressions are anonymous functions. These functions do not need a name or a class to be used. Lambda expressions are added in Java 8. Lambda expressions basically express instances of functional interfaces An interface with a single abstract method is called a functional interface. One example is java.lang.Runnable.