1. **Explain the below**

**Java.Data.LocalTime , LocalDate, LocalDateTime , LocalDateTimeFormatter , Period**

1. **Explain list.removeif() in java 1.8 ?**

In java1.8, there is a new default method introduced in Collection interface. Which will remove the elements from the list based on the predicate condition given.

Ex:

List<Employee> empList=new Arraylist<Employee>();

Emplist.add(new Employee(“Vasantha”,20)

empList.add(new Employee(“sumathi”,35)

// remove an employee who has age more than 30

empList.removeIf((Employee emp)->emp.age>30)

now the will have only employee details of “Vasantha”.

This is the replacement method for the Iterator.remove().

By using the iterator , we have to iterate over the list and check the condition then remove it. It requires more code but the removeIf() do the same functions in single line using the Lambda expression as predicate.

1. **How it will behave ?**

**ArrayList a=new ArrayList();**

**a.add(“fg”)**

**a.add(“Sd”);**

**String strArry[]=new String[3];**

**String str2[]=a.toArray(strArry); -How it will behave ?**

**What will the value in strArry[] and str2[] ?**

**Ans:**

Both strArry and str will have the value of list.

**a.toArray(strArry) :-**  this line convert the list to an array and poy the values to strArray

**String str2[]=a.toArray(strArry); :-** the converted array also assigned to str2 .

So both of the array will have the same values.

1. **What is the difference between toArray() and toArray(T[] t) in ArrayList?**

The main difference is with their signature.

Public Object[] toArray() :- It returns an Object array . it convert the list to array with all element from the list.

Public T[] toArray(T[] t) :- it returns the specified type of array which is passed as an argument. It converts the list to array with all the element from the list . It also copy the returned array to its argument variable of specified array type.

1. **What is AtomicLong ? and where it is used?**

It is an long value that can be updated automatically.(atomically) . This is used in application where the sequence number to be generated atomically. This cannot replace the Long . Which is available in java.util.concurrent.atomic package

**Constructor :**

**AtomicLong () :** Creates the new AtomicLong with intial value zero.

**AtomicLong(int initialvalue) :** creates the new atomic long with specified initial value

**Methods:**

1. **addAndGet(long value) :**  add the given value to the current value
2. **compareAndGet:**