**APIs and Annotation**

**Assignment**

1. **Program to display current date and time in java?**

Ans: **package** Annotation;

**import** java.time.\*;

**public** **class** Current\_Date\_and\_Time

{

**public** **static** **void** main(String[] args)

{

LocalDate date=LocalDate.*now*();

System.***out***.println(date);

LocalTime time=LocalTime.*now*();

System.***out***.println(time);

}

}

//Output:

// 2023-08-13

// 12:20:12.913261700

1. **Write a program to convert a date to a string in the format "MM/dd/yyyy"?**

Ans:

**package** Annotation;

**import** java.time.LocalDate;

**import** java.time.format.DateTimeFormatter;

**public** **class** Date\_To\_String

{

**public** **static** **void** main(String[] args)

{

LocalDate date=LocalDate.*of*(2023, 8, 13);

DateTimeFormatter formatter = DateTimeFormatter.*ofPattern*("MM/dd/yyyy");

String formattedDate = date.format(formatter);

System.***out***.println("Formatted Date:"+formattedDate);

}

}

//Output:Formatted Date:08/13/202

1. **What is the difference between collections and streams? Explain with an Example?**

Ans:

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr. No.** | **Key** | **Collections** | **Streams** |
| 1 | Basic | It is used for storing and manipulating group of data | Stream API is only used for processing group of data |
| 2 | Package | All the classes and interfaces of this API is in the java.util package | All the classes and interfaces of this API is in the java.util.stream  package |
| 3 | Eager/Lazy | All the elements in the collections are computed in the beginning. | In streams, intermediate operations are lazy. |
| 4. | Data Modification | In collections, we can remove or add elements. | We can’t modify streams. |
| 5 | External /Internal iterator | Collections perform iteration over the collection. | Stream perform iteration   internally. |

**Example of Collection:-**

**import** java.util.\*;;

**public** **class** Example\_of\_Collection

{

**public** **static** **void** main(String[] args)

{

List<String> laptopList = **new** ArrayList<>();

laptopList.add("HCL");

laptopList.add("Apple");

laptopList.add("Dell");

Comparator<String> com = (String o1, String o2)->o1.compareTo(o2);

Collections.*sort*(laptopList,com);

**for** (String name : laptopList)

{

System.***out***.println(name);

}

}

}

//Output:

// Apple

// Dell

// HCL

**Example of Streams:-**

**package** StreamAPI;

**import** java.util.\*;

**public** **class** Example\_of\_Streams

{

**public** **static** **void** main(String[] args)

{

List<String> laptopList = **new** ArrayList<>();

laptopList.add("HCL");

laptopList.add("Apple");

laptopList.add("Dell");

laptopList.stream().sorted().forEach(System.***out***::println);

}

}

//Output:

// Apple

// Dell

// HCL

1. **What is enums in java? explain with an example?**

Ans: An enum is a special "class" that represents a group of constants (unchangeable variables, like final variables).To create an enum, use the enum keyword (instead of class or interface), and separate the constants with a comma.

Example:**package** Enum\_In\_Java;

**enum** Level

{

***LOW***,

***MEDIUM***,

***HIGH***;

}

**public** **class** Example\_of\_enum

{

**public** **static** **void** main(String[] args)

{

Level myVar = Level.***LOW***;

**switch**(myVar)

{

**case** ***LOW***:

System.***out***.println("Low level");

**break**;

**case** ***MEDIUM***:

System.***out***.println("Medium level");

**break**;

**case** ***HIGH***:

System.***out***.println("High level");

**break**;

}

}

}

//Output:Low level

1. **What are in built annotations in java?**

## Ans: Built-In Java Annotations  are:

* @Override
* @SuppressWarnings
* @Deprecated
* @Target
* @Retention
* @Inherited
* @Documented