1) Define stream. write a program to copy the content of file 1 to file 2

Ⓐ A stream is a sequence of elements that can be processed in parallel (or) sequentially. It represents a pipline of data that can be operated upon to perform various transformations & computations.

~~Here~~
Stream allow for concise & expressive code especially when working with collections

Here some key characteristics

1. sequence of elements
2. functional operations.
3. Laziness & short - circuiting.
4. pipelining.
5. Intermediate & terminal operations

program :-

```
import Java.io.FileReader;
import Java.io.FileWriter;
import Java.io.IOException;

public class FileCopyExample {

    public static void main (string args[])
        string SourceFile = "file1.txt";
        string destinationFile = "file 2.txt";

    try ( FileReader reader = new FileReader (SourceFile);
        FileWriter writer = new FileWriter (destinationFile
                                           ))
    {
        int character;
        while ((character == reader.read()) != -1)
        {
```

```
    Writer.write (character);
 }
 System.out.println ("File Copy successful");
 }
catch (IO Exception e)
  {
    System.out.println ("An error occured during file Copy:"
                              + e.getMessage());
   }
  }
}.
```

② What is byte stream (binary IO)? List out various classes and methods of byte streams with an example. (File input Stream, File output stream example program).

Ⓐ Byte streams also known as binary I/O streams, are used for reading & writing raw binary data

Byte streams are typically used when working with files. (or) network sockets where the data is treated as a sequence of byte.

~~There are~~

File Input stream :- It is used to read data from a file as a sequence of bytes. It provides methods for reading data from file into a byte array (or) a single byte at a time.

File output stream :- It is used to write data to a file as a sequence of bytes. It provides methods for writing data from a byte array (or) a single byte at a time

Example :-

```
import java. io. File Input stream;
import java .io. File output stream;
import java .io. IOException :

public class Byte stream Example
{
    public static void main (string args [])
    {
        String source File = "file1.txt";
        string destination File = "file2.txt";
  try (File Input stream fis = new FileInputstream (source File)
                                                              ;
       Fileoutput stream fos = new Fileoutput stream
                                          (destinationFile))
     {
         byte [] buffer = new byte [1024];
         inte bytesRead ;
        while ((bytes Read = fis . read (buffer))! = -1)
         {
             fos. Write (buffer, 0, bytes Read);
         }
         system.out.println ("File Copy successful");
        }
       catch ( IOException e)
       {
        system .out. printh ("An error occurred during
                  file copy :" + e.getMessage ());
        }
     }
 }.
```

③ What is character stream (Text IO)?, List out Various classes & methods of character streams with an example (FileReader, FileWriter Example program).

Ⓐ Character streams also known as text I/O streams are used for reading & writing character-based data. character streams are typically used when working with text files (or) textual data.

Two commonly used classes for characters stream I/O are FileReader & FileWriter

FileReader :- It is used to read character data from a file. It provides methods for reading characters into character arrays (or) reading a single character at a time

FileWriter :- It is used to write character data to a file It provides methods for writing characters from character array (or) writing a single character at a time.

program :-
```
            import Java.io.FileReader;
            import java.io.FileWriter;
            import Java.io.IoException;

    public class characterstreamExample
     {
        public static void main (string [] args)
         {
            string sourceFile = "file1.txt".
           String destinationFile = "file2.txt";
        try (File Reader reader = new FileReader(sourceFile);
             File Writer Writer = new File Writer (destination
                                                    File))
        {
          char [] buffer = new char[1024];
```

```
int charsRead

 While ((charsRead = reader.read(buffer)) != -1)
   {
      Writer.write (buffer, 0, charsRead);
   }
   system.out.println ("File Copy successful!");
  }
   catch (IOException e)
   {
     system.out.println(" An error occured during file
                  Copy:" + e.getMessage());
       }
   }
}
```

5th question

program :-

```
import Java.io.FileReader;
import Java.io.FileWriter;
import.Java.io.IOException;

public class FileCopyExample{

    public static void main (String args[])

        String SourceFile = "file1.txt";
        String destinationFile = "file2.txt";

    try ( FileReader reader = new FileReader (SourceFile);
        FileWriter writer = new FileWriter (destinationFile
                                                        ))
      {
        int character;
        while ((character == reader.read()) != -1)
         {
```

5th question

```
   Writer.write (character);
  }
  System.out.println ("File Copy successful");
  }
  catch (IO Exception e)
   {
     System.out.println ("An error occured during file Copy:"
                               + e.getMessage());
     }
   }
  }.
```

4.

Ex:-

```
    import java.io.BufferedReader;
    Import java.io.FileReader;
    import java.io.IOException;

public class File Content Reader
 {
    public static void main (String args[])
    {
        String filepath = "path/to/your/file.txt";
  try (Buffered Reader reader = new Buffered reade
        r (new fileReader (filepath)))
        {
          String line;
        while ((line = reader.readLine()) ! = null)
        {
          System.out.println (line);
        }
    }
    catch (IO Exception e)
        {
          e.print stack Trace();
        }
    }
}
```

5) Diff b/w Txt & binary I/O.
6

Text I/O :-

1. Text I/O is designed for handling text-based data such as character strings.

* It uses character streams to read & write data

* Text I/O assumes that the data is encoded using specific character encoding.

* It is suitable for reading & writing text files where data is expected to be human readble & editable

binary I/O :-

* Binary I/O is designed for handling raw binary data such as Images, audio files.

2. It uses byte streams to read & write data as sequence of bytes.

3. Binary I/O treats the data as a stream of bytes without any assumptions.

4. It is suitable for reading & writing non text files (or) working with low level data.