Java I/O File Handling -

1. Write a program to create a new text file named test.txt.

A.
```
package File_handling;
import java.io.File;
public class CreateNew_file {
	public static void main(String[] args) {
		File f=new File("test.txt");
		try
		{
			f.createNewFile();
			System.out.println(f.setWritable(true));
		}
		catch(Exception e)
		{
			System.out.println("File not found");
			System.out.println(e);
		}
	}

}
```

2. Write a program to check whether a file exists at a given path.

A.
```
package File_handling;
import java.io.File;
public class fileDetails {
//access existing file
	public static void main(String[] args) {
		File f=new File("test.txt");
		if(f.exists())
		{
			System.out.println("File name:"+f.getName());
			System.out.println("File
Location:"+f.getAbsolutePath());
			System.out.println("File size:"+f.length());
```

```
                System.out.println("File readable:"+f.canRead());
                System.out.println("File Writable:"+f.canWrite());

            }
            else
            {
                System.out.println("File not found");
            }

        }

}
```

3. Write a Java program to write "Hello, World!" into a file using FileWriter.

A.
```
package File_handling;
import java.io.FileWriter;
import java.io.IOException;
public class WriteFile {
    public static void main(String[] args) {
        try {
            FileWriter writer = new FileWriter("test.txt");
            writer.write("Hello, World!");
            writer.close();
            System.out.println("Write successful");
```

```
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

4. Write a program to read the content of a file line by line using BufferedReader.

A.
```
package File_handling;
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;
public class ReadFile {
   public static void main(String[] args) {
      try (BufferedReader r = new BufferedReader(new FileReader("sample.txt"))) {
         String line;
         while ((line = r.readLine()) != null) {
            System.out.println(line);
         }
      } catch (IOException e) {
         System.out.println("Error reading the file.");
         e.printStackTrace();
      }
```

```
    }
}
```

5. Write a program to append a line of text to an existing file.

A.
```
package File_handling;
import java.io.BufferedWriter;
import java.io.FileWriter;
import java.io.IOException;
public class AppendToFile {
    public static void main(String[] args) {
        try (BufferedWriter bw = new BufferedWriter(new FileWriter("sample.txt", true))) {
            bw.write("This is the appended line.");
            bw.newLine();
            System.out.println("Line appended successfully.");
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

6. Write a program to count the number of lines, words, and characters in a file.

A.
```
package File_handling;
import java.io.BufferedReader;
import java.io.FileReader;
import java.io.IOException;
public class CountFileDetails {
    public static void main(String[] args) {
        int lines = 0, words = 0, chars = 0;
        try (BufferedReader br = new BufferedReader(new FileReader("sample.txt"))) {
            String line;
            while ((line = br.readLine()) != null) {
                lines++;
                String[] wordList = line.trim().split("\\s+");
```

```
            words += wordList.length;
            chars += line.length();
        }
        System.out.println("Lines: " + lines);
        System.out.println("Words: " + words);
        System.out.println("Characters: " + chars);
    } catch (IOException e) {
        e.printStackTrace();
    }
  }
}
```

7. Write a program to copy content from one file to another using FileReader and FileWriter.

A.
```
package File_handling;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
public class CopyFile {
   public static void main(String[] args) {
      try (FileReader fr = new FileReader("source.txt");
           FileWriter fw = new FileWriter("destination.txt")) {
         int ch;
         while ((ch = fr.read()) != -1) {
            fw.write(ch);
         }
         System.out.println("File copied successfully.");
      } catch (IOException e) {
         e.printStackTrace();
      }
   }
}
```

8. Write a program that lists all the files in a directory.

A.
```
package File_handling;
import java.io.File;
public class ListFilesInDirectory {
```

```
    public static void main(String[] args) {
        File dir = new File("your_directory_path_here");
        if (dir.isDirectory()) {
            File[] files = dir.listFiles();
            if (files != null) {
                for (File f : files) {
                    if (f.isFile()) {
                        System.out.println(f.getName());
                    }
                }
            } else {
                System.out.println("No files found.");
            }
        } else {
            System.out.println("Not a directory.");
        }
    }
}
```

9. Write a program to filter and display only .txt files from a folder using FilenameFilter.

A.
```
package File_handling;
import java.io.File;
import java.io.FilenameFilter;
public class TxtFilesFilter {
    public static void main(String[] args) {
        File dir = new File("your_directory_path_here");
        FilenameFilter filter = new FilenameFilter() {
            @Override
            public boolean accept(File dir, String name) {
                return name.endsWith(".txt");
            }
        };
        String[] txtFiles = dir.list(filter);
        if (txtFiles != null) {
            for (String file : txtFiles) {
```

```
            System.out.println(file);
        }
    } else {
        System.out.println("No .txt files found.");
    }
  }
}
```

10. Write a program to serialize and deserialize a Student object to and from a file.

A.
```
package File_handling;
import java.io.*;
class Student implements Serializable {
   private static final long serialVersionUID = 1L;
   int id;
   String name;
   Student(int id, String name) {
      this.id = id;
      this.name = name;
   }
   @Override
   public String toString() {
      return "Student[id=" + id + ", name=" + name + "]";
   }
}
public class StudentSerialization {
   public static void main(String[] args) {
      Student s = new Student(101, "Suma");
      // Serialization
      try (ObjectOutputStream oos = new ObjectOutputStream(new
FileOutputStream("student.ser"))) {
         oos.writeObject(s);
         System.out.println("Student object serialized.");
      } catch (IOException e) {
         e.printStackTrace();
      }
```

```
        // Deserialization
        try (ObjectInputStream ois = new ObjectInputStream(new
FileInputStream("student.ser"))) {
            Student deserializedStudent = (Student) ois.readObject();
            System.out.println("Deserialized Student: " +
deserializedStudent);
        } catch (IOException | ClassNotFoundException e) {
            e.printStackTrace();
        }
    }
}
```

11. Write a program to read a file using Scanner and display the tokens.

A.
```
package File_handling;
import java.io.File;
import java.io.FileNotFoundException;
import java.util.Scanner;
public class ReadFileWithScanner {
    public static void main(String[] args) {
        try {
            File file = new File("sample.txt");
            Scanner scanner = new Scanner(file);
            while (scanner.hasNext()) {
                String token = scanner.next();
                System.out.println(token);
            }
            scanner.close();
        } catch (FileNotFoundException e) {
            System.out.println("File not found");
            e.printStackTrace();
        }
    }
}
```

12. Write a program to search for a specific word in a file and count its occurrences.

A. package File_handling;

```
import java.io.File;
import java.io.FileNotFoundException;
import java.util.Scanner;

public class WordCountInFile {
    public static void main(String[] args) {
        String searchWord = "Java";
        int count = 0;

        try {
            File file = new File("sample.txt");
            Scanner scanner = new Scanner(file);

            while (scanner.hasNext()) {
                String word = scanner.next();
                if (word.equalsIgnoreCase(searchWord)) {
                    count++;
                }
            }
            scanner.close();
            System.out.println("The word '" + searchWord + "' occurs "
+ count + " times.");
        } catch (FileNotFoundException e) {
            System.out.println("File not found");
            e.printStackTrace();
        }
    }
}
```

13. Write a program to create, move, and delete a file using Files and Paths.

A.
```
package File_handling;
import java.io.IOException;
import java.nio.file.*;
public class FileOperations {
    public static void main(String[] args) {
        Path source = Paths.get("testfile.txt");
        Path target = Paths.get("newfolder/movedfile.txt");
        try {
            // Create a file
            if (!Files.exists(source)) {
                Files.createFile(source);
                System.out.println("File created: " + source);
            }
            // Create target directory if not exists
            if (!Files.exists(target.getParent())) {
                Files.createDirectories(target.getParent());
            }
            // Move file
            Files.move(source, target,
StandardCopyOption.REPLACE_EXISTING);
            System.out.println("File moved to: " + target);
            // Delete file
            Files.delete(target);
            System.out.println("File deleted: " + target);
        } catch (IOException e) {
            System.out.println("Error occurred:");
            e.printStackTrace();
        }
    }
}
```

14. Write a program to read all lines of a file using Files.readAllLines() and print them.

A.
```
package File_handling;
```

```
import java.io.IOException;
import java.nio.file.*;
import java.util.List;
public class ReadAllLinesExample {
   public static void main(String[] args) {
      Path path = Paths.get("sample.txt");
      try {
         List<String> lines = Files.readAllLines(path);
         for (String line : lines) {
            System.out.println(line);
         }
      } catch (IOException e) {
         System.out.println("Error reading file");
         e.printStackTrace();
      }
   }
}
```

15. Write a program to write data into a file using Files.write() and append using StandardOpenOption.APPEND.

A.
```
package File_handling;
import java.io.IOException;
import java.nio.file.*;
import java.util.Arrays;
public class WriteAndAppendFile {
   public static void main(String[] args) {
      Path path = Paths.get("sample.txt");
      String dataToWrite = "This is a new line.\n";
      try {
         // Write initial data (overwrites if file exists)
         Files.write(path, Arrays.asList("Initial line in file."), StandardOpenOption.CREATE);
         // Append new data
         Files.write(path, dataToWrite.getBytes(), StandardOpenOption.APPEND);
```

```
            System.out.println("Data written and appended
successfully.");
        } catch (IOException e) {
            System.out.println("Error writing/appending to file");
            e.printStackTrace();
        }
    }
}
```

16. Write a program to walk through a directory tree and display file names using Files.walk().

A.
```
package File_handling;
import java.io.IOException;
import java.nio.file.*;
import java.util.stream.Stream;
public class WalkDirectoryTree {
    public static void main(String[] args) {
        Path start = Paths.get(".");
        try (Stream<Path> stream = Files.walk(start)) {
            stream.filter(Files::isRegularFile)
                  .forEach(System.out::println);
        } catch (IOException e) {
            System.out.println("Error walking through directory");
            e.printStackTrace();
        }
    }
}
```

17. Write a program to copy a file using Files.copy() with REPLACE_EXISTING option.

A.
```
package File_handling;
import java.io.IOException;
import java.nio.file.*;
public class CopyFileExample {
    public static void main(String[] args) {
        Path source = Paths.get("source.txt");
        Path destination = Paths.get("destination.txt");
```

```
        try {
            Files.copy(source, destination,
StandardCopyOption.REPLACE_EXISTING);
            System.out.println("File copied successfully.");
        } catch (IOException e) {
            System.out.println("Error copying file");
            e.printStackTrace();
        }
    }
}
```

18. Write a program to check and print the size of a file in bytes using Files.size().

A.
```
package File_handling;
import java.io.IOException;
import java.nio.file.*;
public class FileSizeExample {
    public static void main(String[] args) {
        Path path = Paths.get("sample.txt");
        try {
            long size = Files.size(path);
            System.out.println("Size of the file in bytes: " + size);
        } catch (IOException e) {
            System.out.println("Error getting file size");
            e.printStackTrace();
        }
    }
}
```

19. Write a program to serialize a class Employee and store it in employee.ser.

A.
```
package File_handling;
import java.io.*;
class Employee implements Serializable {
    private static final long serialVersionUID = 1L;
    int id;
```

```
    String name;
    double salary;
    public Employee(int id, String name, double salary) {
        this.id = id;
        this.name = name;
        this.salary = salary;
    }
}
public class SerializeEmployee {
    public static void main(String[] args) {
        Employee emp = new Employee(101, "Alice", 75000);
        try (ObjectOutputStream oos = new ObjectOutputStream(new
FileOutputStream("employee.ser"))) {
            oos.writeObject(emp);
            System.out.println("Employee object serialized.");
        } catch (IOException e) {
            System.out.println("Error serializing object");
            e.printStackTrace();
        }
    }
}
```

20. Write a program to deserialize the employee.ser file and display the object data.

A.
```
package File_handling;
import java.io.*;
public class DeserializeEmployee {
    public static void main(String[] args) {
        try (ObjectInputStream ois = new ObjectInputStream(new
FileInputStream("employee.ser"))) {
            Employee emp = (Employee) ois.readObject();
            System.out.println("Employee Details:");
            System.out.println("ID: " + emp.id);
            System.out.println("Name: " + emp.name);
            System.out.println("Salary: " + emp.salary);
```

```
        } catch (IOException | ClassNotFoundException e) {
            System.out.println("Error deserializing object");
            e.printStackTrace();
        }
    }
}
```