FILE HANDLING CONCEPT

- exists() : file is a non-static method present in file class, It will check whether the file exist in given path, if yes it returns boolean value true else false.

```
import java.io.File;

public class A {
    public static void main(String[] args) {

        String str ="C:\\Users\\MERAZ\\Desktop\\New Text Document.txt";

        File file = new File(str);
        boolean exists = file.exists();
        System.out.println(exists);

    }}
```

- delete(): – delete is a non-static method present in file class whose return value is boolean value, if the file exist it will delete and return true, but if not, there is given path, then it cannot delete and hence it returns false.

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

        String str ="C:\\Users\\MERAZ\\Desktop\\sam.txt";

        File file = new File(str);
        boolean exists = file.delete();
        System.out.println("deleted successfully !"+exists);

    }}
```

- createNewFile():- Create is a non-static method present in file class. If the file does not exist in the given path, then we create a new file and return boolean value true. If the file already existed in the given path then it will not override, create a new file hence it return false.

```
import java.io.File;
import java.io.IOException;

public class A {
    public static void main(String[] args) throws IOException {
```

```
            String str ="C:\\Users\\MERAZ\\Desktop\\sam.txt";

            File file = new File(str);
            boolean newFile = file.createNewFile();
            if(newFile==false){
                System.out.println("you have already created file ");
            }

        }}
```

- mkdir():- Folder – directory, if it creates folder, then return boolean value true, if it already exists then return false.

```
public class A {
    public static void main(String[] args) throws IOException {

        String str ="C:\\Users\\MERAZ\\Desktop\\folder1";

        File file = new File(str);
        boolean newFolder = file.mkdir();
        System.out.println("successfully created !"+newFolder);

    }}
```

- FILE WRITER CLASS:- it is used for writing streams of characters to files.it is best suitable for writing text rather than binary data.

  Write():- it help to write inside the file there are some override method of it.

```
public class A {
    public static void main(String[] args) {
        try {
            FileWriter fw = new
FileWriter("C:\\Users\\MERAZ\\Desktop\\sam.txt");
           String str ="sahil";
            char[]ch = {'x','y','z'};
            fw.write(ch);
            fw.write(str);
            fw.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }}
```

- FILE READER CLASS:- file reader class is used for reading streams of characters from files, it is suitable for reading text files.

  Read():- it helps us to read the file content, there are some override method of it.

```
public class A {
    public static void main(String[] args) {
        try {
            File f = new File("C:\\Users\\MERAZ\\Desktop\\sam.txt");
            FileReader fr = new FileReader(f);
            char [] ch = new char[(int)f.length()];
            fr.read(ch);
            for (char c: ch) {
                System.out.print(c);
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

- BUFFEREDWRITTER :- Improve the file writer, Write the file content with the best performance, It improves file writing performance ,It has newline method, that help us to write the content in new line.

  Write()&newline():-

```
public class A {
    public static void main(String[] args) {
        try {
            FileWriter fw = new
                    FileWriter("C:\\Users\\MERAZ\\Desktop\\sam.txt");
            BufferedWriter bw = new BufferedWriter(fw);
            bw.write("testing");
            bw.newLine();
            char [] ch = {'a','q','f'};
            bw.write(ch);
            bw.close();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }}
```

- BUFFEREDREADER : File reading Improve file reader, Read the file content with the best performance ,It helps us improve file reading performance ,It has readLine method which can read the content from the file, line by line.

readLine():-

```
public class A {
    public static void main(String[] args) {
        try {
            FileReader fr = new
FileReader("C:\\Users\\MERAZ\\Desktop\\sam.txt");
            BufferedReader br = new BufferedReader(fr);
            System.out.println(br.readLine());
            System.out.println(br.readLine());
            System.out.println(br.readLine());
        } catch (Exception e) {
            e.printStackTrace();
        }
    }}
```