COMP 6481: Programming and Problem Solving

Tutorial 5:

Files input/output & Serialization

Question 1

- Read the whole file “foo.txt” line by line using Scanner object. At the same time count the number of lines in file.

```
File text = new File("foo.txt");
Scanner sc = new Scanner(text);

        int lineNumber = 1;
        while(sc.hasNextLine()){
            String line = sc.nextLine();
            System.out.println("line " + lineNumber + " :" + line);
            lineNumber++;
        }
```

Question 2

- Read file “foo.txt” line by line using BufferedReader object and count the number of lines in file. Report exception thrown by BufferedReader.

```
try{
      br = new BufferedReader(new FileReader("foo.txt"));
     String contentLine = br.readLine();
     int lineNumber = 1;
     while (contentLine != null) {
        System.out.println(contentLine);
        contentLine = br.readLine();
        lineNumber++;
     }
     catch (IOException e)
      {
         e.printStackTrace();
      }
```

What is difference between BufferedReader and Scanner

- Scanner is not thread safe whereas in a thread-safe environment BufferedReader is thread safe.
- Scanner buffer is 1024 chars, whereas BufferedReader has a default buffer memory of 8192 chars, and that too can be extended.
- Scanner class uses regular expression in parsing the strings, by default, white space is set as a delimiter, but you can set any other delimiter using useDelimiter() of scanner.
- BufferedReader is only used for reading data, whereas Scanner class is used for reading as well as parsing of data.

Is it possible to serialize Employee objects?

```
class Employee implements Serializable{
    Int empId;
    String name;
    Address address;
    //getters and setters
}

class Address{
    String street;
    String city;
    String pin;
}
```

- No. All the member object must also implement the Serializable. Additionally, if you would like to skip any member to be skipped for serialization, then it can be made transient.

  private transient int foo;

Read binary file input.bin and write the content to output.bin by reading each byte using FileInputStream and FileOutputStream

```
String inputFile = “input.bin”;
String outputFile = “output.bin”;
try (
    InputStream inputStream = new FileInputStream(inputFile);
    OutputStream outputStream = new FileOutputStream(outputFile);
) {
    int byteRead;
    while ((byteRead = inputStream.read()) != -1) {
        outputStream.write(byteRead);
    }
} catch (IOException ex) {
    ex.printStackTrace();
}
```

Assume that the file size is big then the previous solution might not work. Now modify the previous problem so that it can copy block of size 4KB at a time.

```
String inputFile = "input.bin";
String outputFile = "output.bin";

try (
    InputStream inputStream = new FileInputStream(inputFile);
    OutputStream outputStream = new FileOutputStream(outputFile);
) {
    byte[] buffer = new byte[4096];
    while (inputStream.read(buffer) != -1) {
        outputStream.write(buffer);
    }
} catch (IOException ex) {
    ex.printStackTrace();
}
```

What is the output of the following program?

```
Scanner scanner = new Scanner(System.in);
System.out.println(scanner.nextInt());
System.out.println(scanner.nextLine());
```

Assume the user input is:
12
Hello

Output is:
12

References

- https://java2blog.com/difference-between-scanner-bufferreader-java/