PART-5

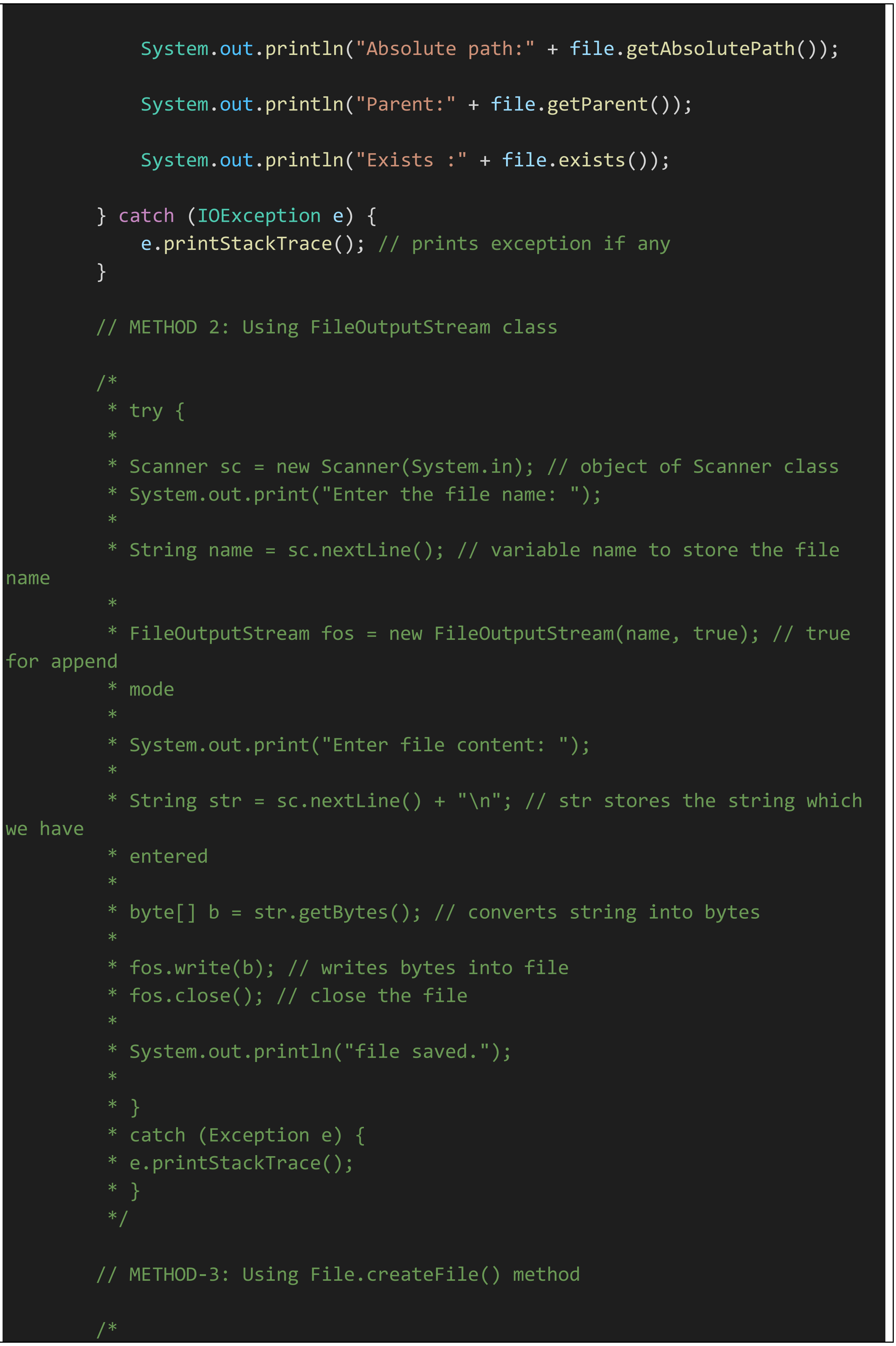
File Handling & Streams

GitHub Repository Link: https://github.com/21ce114/JAVA-Practicals.git

Question 1:	WAP to show how to create a file with different mode and methods of File class to find path, directory etc.

Answer:

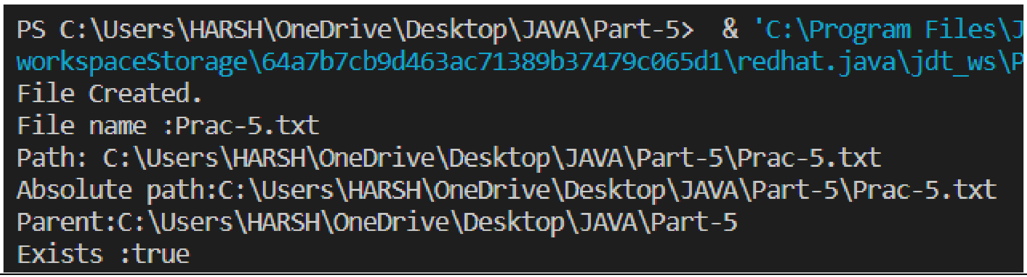
```
/*ID: 21CE114
Name: Harsh Rana
Git Repository Link: https://github.com/21ce114/JAVA-Practicals.git
AIM : WAP to show how to create a file with different mode and methods
of File class to find path, directory etc.*/


import java.io.File;
import java.io.IOException;
import java.io.FileOutputStream; // USED IN METHOD 2
import java.util.Scanner; // USED IN METHOD 2
import java.nio.file.*; // USED IN METHOD 3


public class Practical5_1 {
    public static void main(String[] args) {
        // METHOD 1: Using File.createNewFile() method

        // initialize File object and passing path as argument
        File file = new
File("C:\\Users\\HARSH\\OneDrive\\Desktop\\JAVA\\Part-5\\Prac-5.txt");
        boolean result;

        try {
            result = file.createNewFile(); // creates a new file
            if (result) {
                System.out.println("File Created.");
            } else {
                System.out.println("File was not created.");
            }

            // applying File class methods on File object

            System.out.println("File name :" + file.getName());

            System.out.println("Path: " + file.getPath());
```



```
            System.out.println("Absolute path:" + file.getAbsolutePath());

            System.out.println("Parent:" + file.getParent());

            System.out.println("Exists :" + file.exists());

        } catch (IOException e) {
            e.printStackTrace(); // prints exception if any
        }

        // METHOD 2: Using FileOutputStream class

        /*
         * try {
         *
         * Scanner sc = new Scanner(System.in); // object of Scanner class
         * System.out.print("Enter the file name: ");
         *
         * String name = sc.nextLine(); // variable name to store the file
name
         *
         * FileOutputStream fos = new FileOutputStream(name, true); // true
for append
         * mode
         *
         * System.out.print("Enter file content: ");
         *
         * String str = sc.nextLine() + "\n"; // str stores the string which
we have
         * entered
         *
         * byte[] b = str.getBytes(); // converts string into bytes
         *
         * fos.write(b); // writes bytes into file
         * fos.close(); // close the file
         *
         * System.out.println("file saved.");
         *
         * }
         * catch (Exception e) {
         * e.printStackTrace();
         * }
         */

        // METHOD-3: Using File.createFile() method

        /*
```

```
         * Path path =
         * Paths.get("C:\\Users\\HARSH\\OneDrive\\Desktop\\JAVA\\Part-
5\\Prac-5.txt");
         * // creates Path instance
         *
         * try {
         *
         * Path p = Files.createFile(path); // creates file at specified
location
         * System.out.println("File Created at Path: " + p);
         *
         * }
         * catch (IOException e) {
         *
         * e.printStackTrace();
         *
         * }
         */

    }
}
```

OUTPUT :



```
PS C:\Users\HARSH\OneDrive\Desktop\JAVA\Part-5>  & 'C:\Program Files\J
workspaceStorage\64a7b7cb9d463ac71389b37479c065d1\redhat.java\jdt_ws\P
File Created.
File name :Prac-5.txt
Path: C:\Users\HARSH\OneDrive\Desktop\JAVA\Part-5\Prac-5.txt
Absolute path:C:\Users\HARSH\OneDrive\Desktop\JAVA\Part-5\Prac-5.txt
Parent:C:\Users\HARSH\OneDrive\Desktop\JAVA\Part-5
Exists :true
```

Question 2:

When to use Character Stream over Byte Stream? When to use Byte Stream over Character Stream? Give example.

Answer:

Byte streams are used to perform input and output of 8-bit bytes. Byte streams are useful when we want to read/write binary data. Character stream is used to perform input and output operations of 16-bit Unicode. Character streams are used to read/write characters.

- Character streams are used when we want to process text files.
- Byte streams are used to process raw data like binary files.

Example:

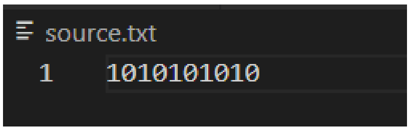
Byte Stream:

```
/*ID: 21CE114
Name: Harsh Rana
```

```
Git Repository Link: https://github.com/21ce114/JAVA-Practicals.git
AIM : When to use Character Stream over Byte Stream? When to use
Byte Stream over Character Stream? Give example..*/

import java.io.*;
public class Practical5_2
{
    public static void main(String[] args) throws IOException
    {
        FileInputStream sourceStream = null;
        FileOutputStream targetStream = null;

        try
        {
            sourceStream = new
FileInputStream("C:\\Users\\HARSH\\OneDrive\\Desktop\\JAVA\\Part-
5\\source.txt");
            targetStream = new FileOutputStream
("C:\\Users\\HARSH\\OneDrive\\Desktop\\JAVA\\Part-5\\destination.txt");

            // Reading source file using read method
            // and write to file byte by byte using write method
            int temp;
            while ((temp = sourceStream.read()) != -1)
                targetStream.write((byte)temp);
        }
        finally
        {
            if (sourceStream != null){
                sourceStream.close();
            }
            if (targetStream != null){
                targetStream.close();
            }
        }
    }
}
```

Files:

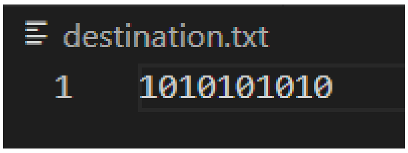
Source.txt



source.txt

```
1010101010
```

Destination.txt



destination.txt

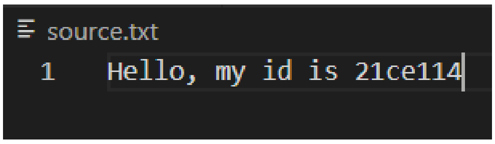
```
1010101010
```

Character Stream:

```
import java.io.*;
public class Practical5_2b {

   public static void main(String args[]) throws IOException {
      FileReader in = null;
      FileWriter out = null;

       // Reading source file using read method
        // and write to file using write method
      try {
         in = new FileReader("C:\\Users\\HARSH\\OneDrive\\Desktop\\JAVA\\Part-5\\source.txt");
         out = new FileWriter("C:\\Users\\HARSH\\OneDrive\\Desktop\\JAVA\\Part-5\\destination.txt");
         int c;
         while ((c = in.read()) != -1) {
            out.write(c);
         }
      }
       finally {
         if (in != null) {
            in.close();
         }
         if (out != null) {
            out.close();
         }
      }
   }
}
```

Files:

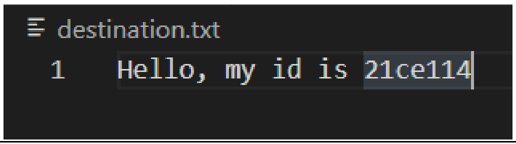
Source.txt



source.txt

```
Hello, my id is 21ce114
```

Destination.txt



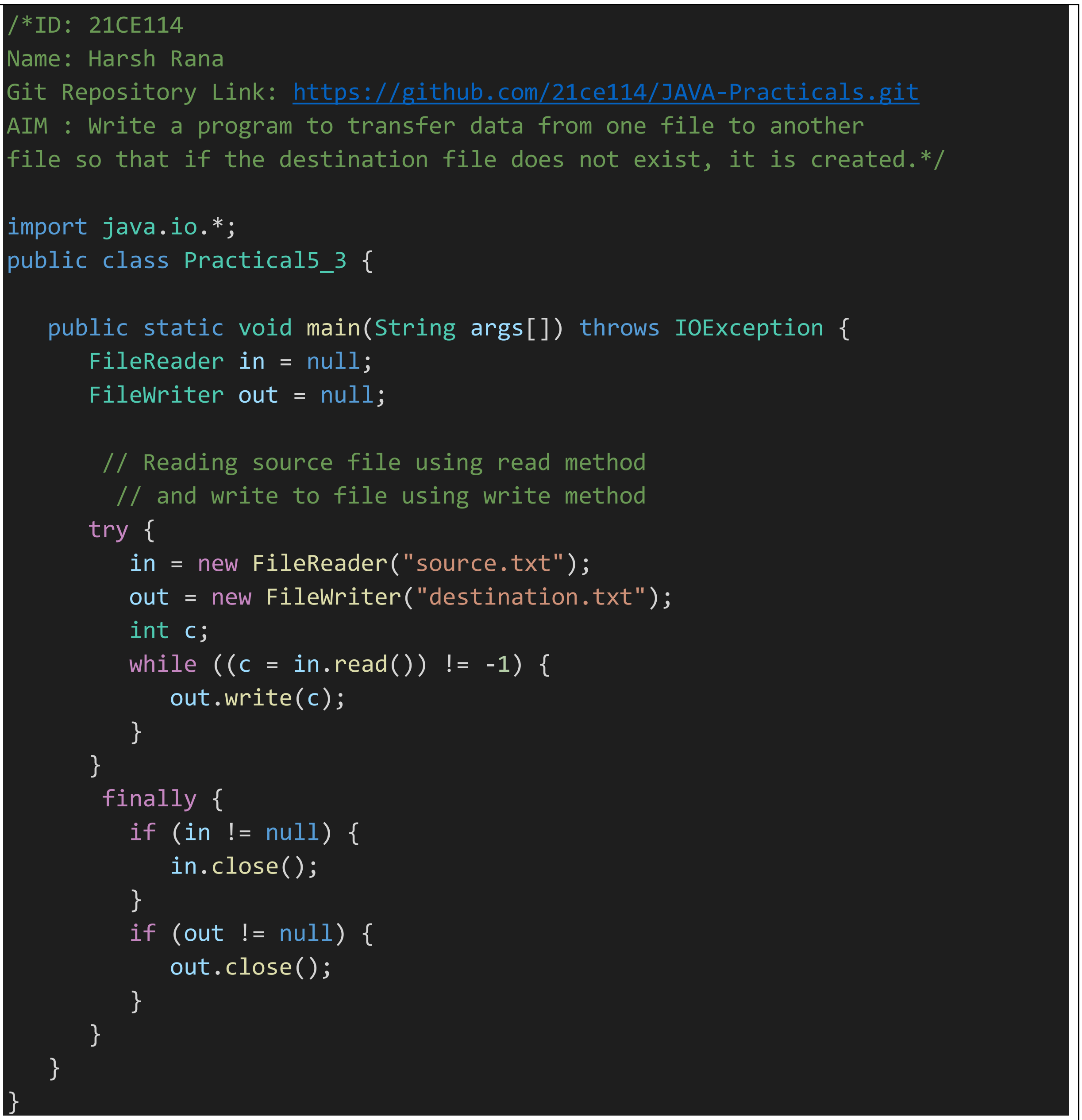
destination.txt

```
Hello, my id is 21ce114
```

Question 3:

Write a program to transfer data from one file to another file so that if the destination file does not exist, it is created.

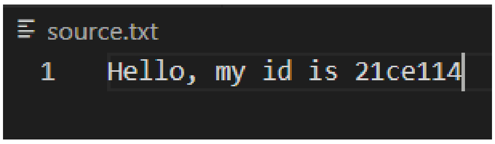
Answer:



```
/*ID: 21CE114
Name: Harsh Rana
Git Repository Link: https://github.com/21ce114/JAVA-Practicals.git
AIM : Write a program to transfer data from one file to another
file so that if the destination file does not exist, it is created.*/

import java.io.*;
public class Practical5_3 {

   public static void main(String args[]) throws IOException {
      FileReader in = null;
      FileWriter out = null;

       // Reading source file using read method
        // and write to file using write method
      try {
         in = new FileReader("source.txt");
         out = new FileWriter("destination.txt");
         int c;
         while ((c = in.read()) != -1) {
            out.write(c);
         }
      }
       finally {
         if (in != null) {
            in.close();
         }
         if (out != null) {
            out.close();
         }
      }
   }
}
```

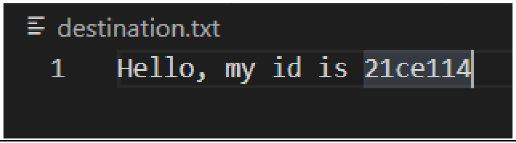
Files:

Source.txt



```
Hello, my id is 21ce114
```

It did not exist before running the code: Destination.txt



```
Hello, my id is 21ce114
```

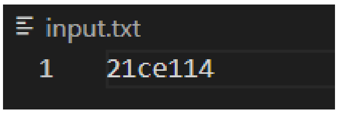
Question 4:

WAP to show use of character and byte stream.

Answer:

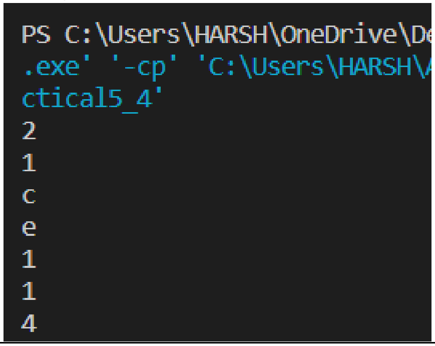
```java
//Examples of Character and Byte stream are already Mentioned in Question 2
//This is another example.
/*ID: 21CE114
Name: Harsh Rana
Git Repository Link: https://github.com/21ce114/JAVA-Practicals.git
AIM : WAP to show use of character and byte stream.*/


import java.io.*;

public class Practical5_4 {
    public static void main(String[] args) throws IOException {
        FileReader sourceStream = null;
        try {
            sourceStream = new
FileReader("C:\\Users\\HARSH\\OneDrive\\Desktop\\JAVA\\Part-5\\input.txt");
            int temp;
            while ((temp = sourceStream.read()) != -1)
                System.out.println((char) temp);
        } finally {
            if (sourceStream != null)
                sourceStream.close();
        }

    }
}
```

File:



```
≡ input.txt
  1    21ce114
```

Output:

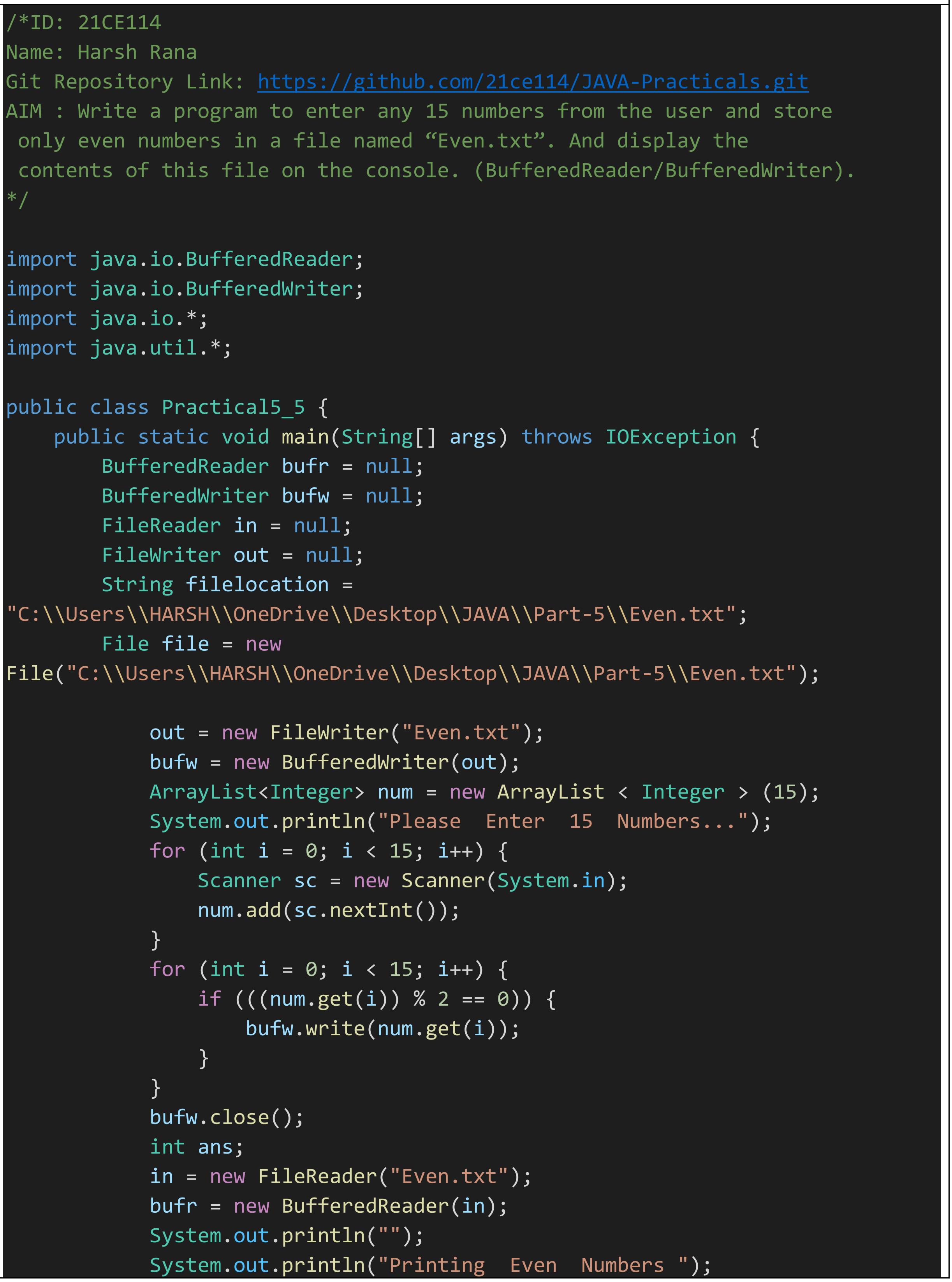


```
PS C:\Users\HARSH\OneDrive\De
.exe' '-cp' 'C:\Users\HARSH\
ctical5_4'
2
1
c
e
1
1
4
```

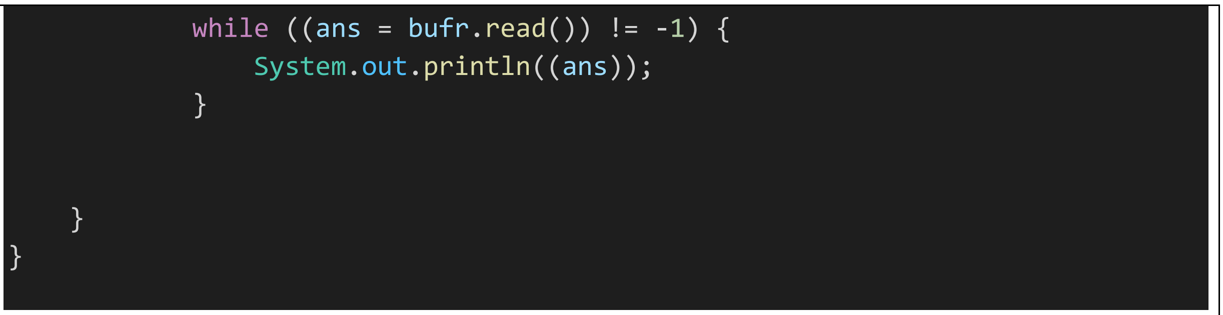
Question 5:

Write a program to enter any 15 numbers from the user and store only even numbers in a file named “Even.txt”. And display the contents of this file on the console. (BufferedReader / BufferedWriter).

Answer:

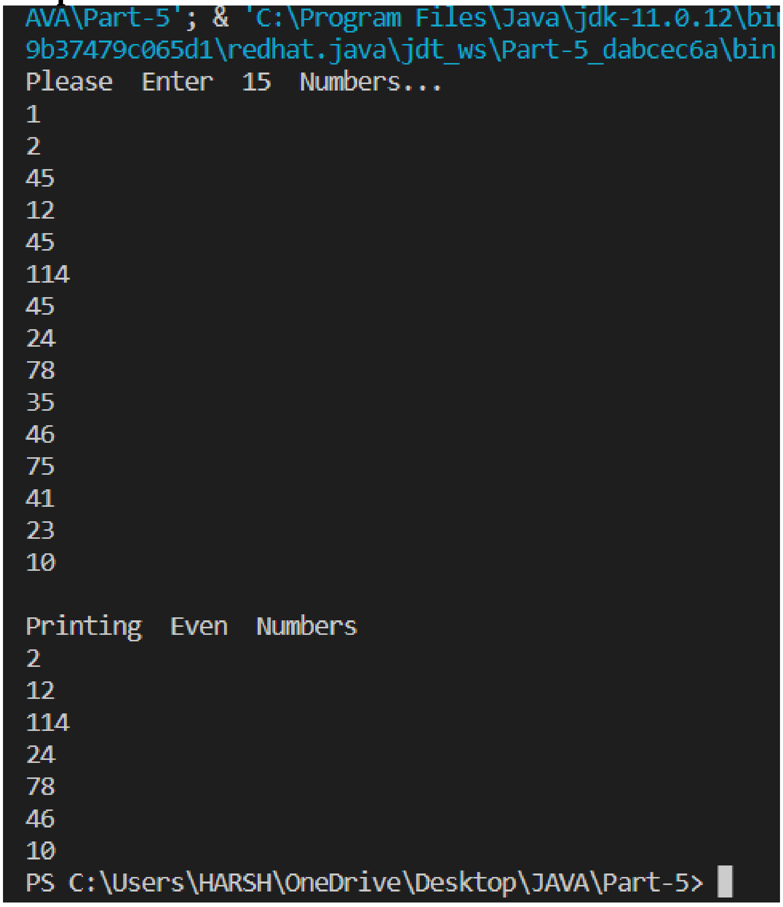


```
/*ID: 21CE114
Name: Harsh Rana
Git Repository Link: https://github.com/21ce114/JAVA-Practicals.git
AIM : Write a program to enter any 15 numbers from the user and store
 only even numbers in a file named “Even.txt”. And display the
 contents of this file on the console. (BufferedReader/BufferedWriter).
*/

import java.io.BufferedReader;
import java.io.BufferedWriter;
import java.io.*;
import java.util.*;

public class Practical5_5 {
    public static void main(String[] args) throws IOException {
        BufferedReader bufr = null;
        BufferedWriter bufw = null;
        FileReader in = null;
        FileWriter out = null;
        String filelocation =
"C:\\Users\\HARSH\\OneDrive\\Desktop\\JAVA\\Part-5\\Even.txt";
        File file = new
File("C:\\Users\\HARSH\\OneDrive\\Desktop\\JAVA\\Part-5\\Even.txt");

            out = new FileWriter("Even.txt");
            bufw = new BufferedWriter(out);
            ArrayList<Integer> num = new ArrayList < Integer > (15);
            System.out.println("Please  Enter  15  Numbers...");
            for (int i = 0; i < 15; i++) {
                Scanner sc = new Scanner(System.in);
                num.add(sc.nextInt());
            }
            for (int i = 0; i < 15; i++) {
                if (((num.get(i)) % 2 == 0)) {
                    bufw.write(num.get(i));
                }
            }
            bufw.close();
            int ans;
            in = new FileReader("Even.txt");
            bufr = new BufferedReader(in);
            System.out.println("");
            System.out.println("Printing  Even  Numbers ");
```



```
            while ((ans = bufr.read()) != -1) {
                System.out.println((ans));
            }


    }
}
```

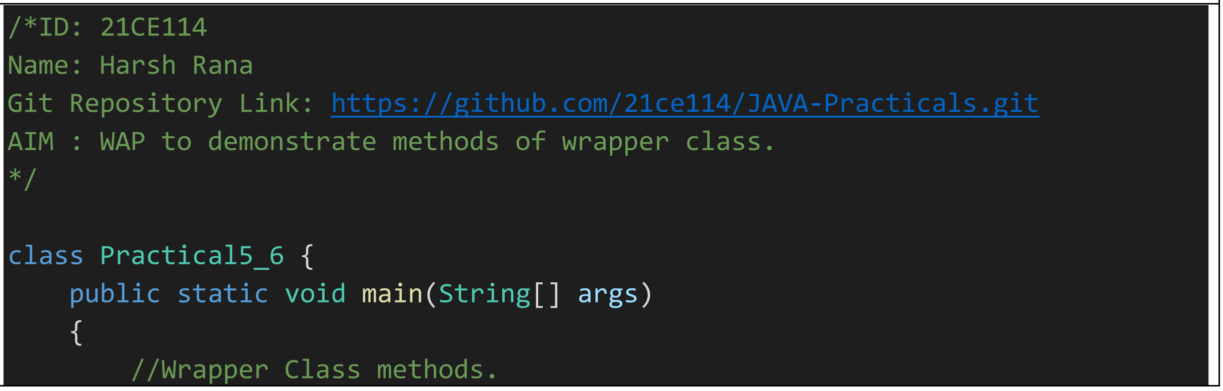
Output:



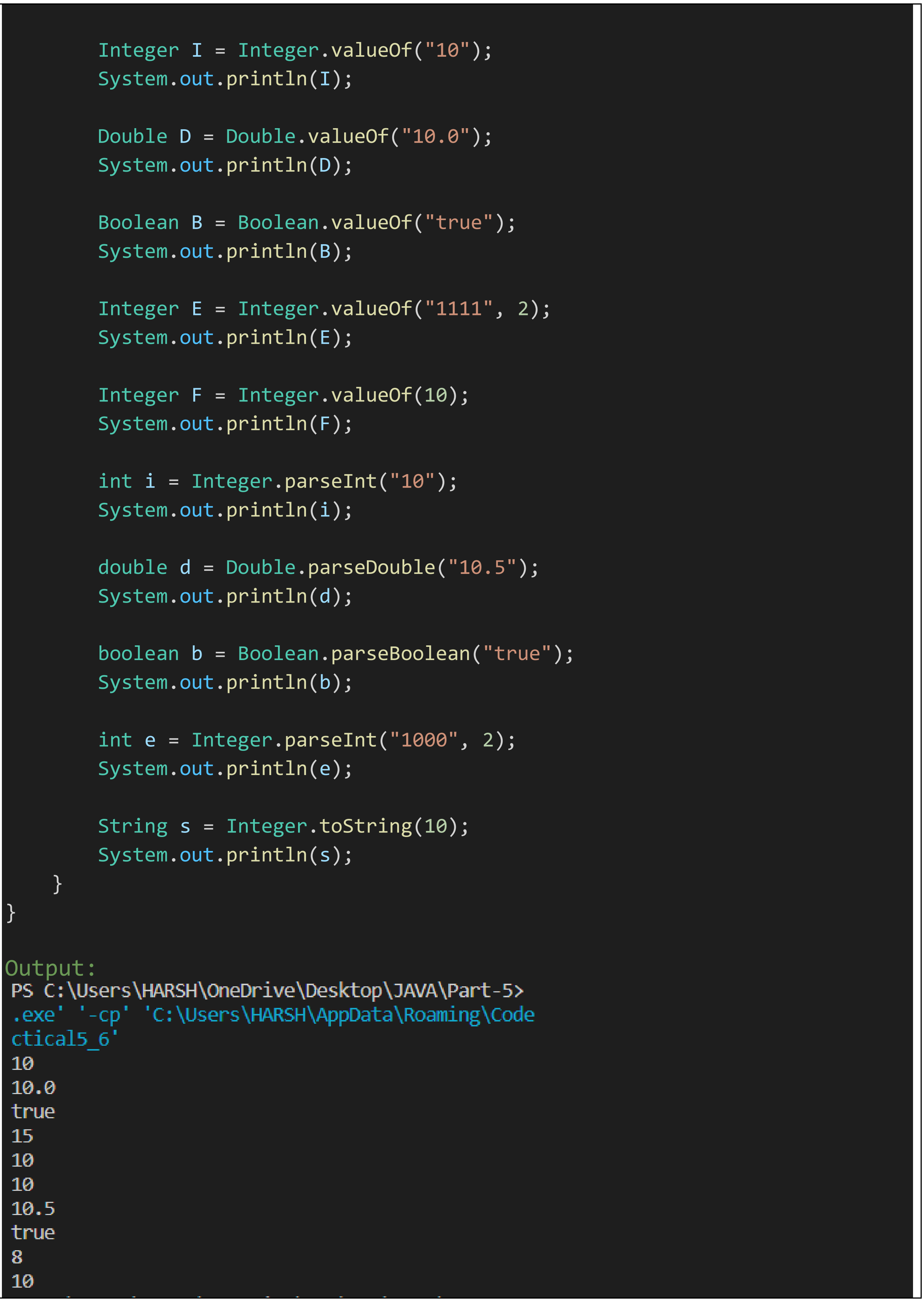
```
AVA\Part-5'; & 'C:\Program Files\Java\jdk-11.0.12\bi
9b37479c065d1\redhat.java\jdt_ws\Part-5_dabcec6a\bin
Please  Enter  15  Numbers...
1
2
45
12
45
114
45
24
78
35
46
75
41
23
10

Printing  Even  Numbers
2
12
114
24
78
46
10
PS C:\Users\HARSH\OneDrive\Desktop\JAVA\Part-5>
```

Question: 6

WAP to demonstrate methods of wrapper class.



```
/*ID: 21CE114
Name: Harsh Rana
Git Repository Link: https://github.com/21ce114/JAVA-Practicals.git
AIM : WAP to demonstrate methods of wrapper class.
*/

class Practical5_6 {
    public static void main(String[] args)
    {
        //Wrapper Class methods.
```



```
        Integer I = Integer.valueOf("10");
        System.out.println(I);

        Double D = Double.valueOf("10.0");
        System.out.println(D);

        Boolean B = Boolean.valueOf("true");
        System.out.println(B);

        Integer E = Integer.valueOf("1111", 2);
        System.out.println(E);

        Integer F = Integer.valueOf(10);
        System.out.println(F);

        int i = Integer.parseInt("10");
        System.out.println(i);

        double d = Double.parseDouble("10.5");
        System.out.println(d);

        boolean b = Boolean.parseBoolean("true");
        System.out.println(b);

        int e = Integer.parseInt("1000", 2);
        System.out.println(e);

        String s = Integer.toString(10);
        System.out.println(s);
    }
}

Output:
 PS C:\Users\HARSH\OneDrive\Desktop\JAVA\Part-5>
 .exe' '-cp' 'C:\Users\HARSH\AppData\Roaming\Code
 ctical5_6'
 10
 10.0
 true
 15
 10
 10
 10.5
 true
 8
 10
```