

File Handling

File handling in Java implies reading from and writing data to a file. The File class from the java.io package, allows us to work with different formats of files. In order to use the File class, you need to create an object of the class and specify the filename or directory name.

- Create an object of file and check whether the file exists or not

```
import java.io.File;

public class Alpha {

    public static void main(String[] args) {

        //      Specify the directory in which file exists
        String path =
"C:\\Users\\Megha\\Desktop\\MyFiles\\data.txt";

        //      Create an object of a file
        File file = new File(path);

        //      exists() will return true with the file exists
        //      in that directory else it will return false
        System.out.println(file.exists());
    }
}
```

Now let's see different methods available in File class

- **canRead() and canWrite():**

canRead() method will check whether the file is readable or not. It returns true if it is readable else false.

canWrite() method will check whether the file is writable or not. It returns true if it is readable else false.

```
import java.io.File;

public class Alpha {

    public static void main(String[] args) {

        //      Specify the directory in which file exists
        String path =
"C:\\Users\\Megha\\Desktop\\MyFiles\\data.txt";

        //      Create an object of a file
        File file = new File(path);

        //      canRead() will check whether the file is
        //      readable or not
        System.out.println(file.canRead());

        //      canWrite() will check whether the file is
        //      readable or not
        System.out.println(file.canWrite());
    }
}
```

- **getName():**

This method returns the Name of the given file object. The function returns a string object which contains the Name of the given file object.

```
import java.io.File;

public class Alpha {

    public static void main(String[] args) {

        //      Specify the directory in which file exists
        String path =
"C:\\Users\\Megha\\Desktop\\MyFiles\\data.txt";

        //      Create an object of a file
        File file = new File(path);

        //      getName() will return the name of the file
        System.out.println(file.getName());
    }

}
```

- **getParent():**

This method returns a String which denotes the pathname string of the parent directory named by this abstract pathname, or null if this pathname does not name a parent.

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

```
public class Alpha {  
  
    public static void main(String[] args) {  
  
        //      Specify the directory in which file exists  
        String path =  
        "C:\\Users\\Megha\\Desktop\\MyFiles\\data.txt";  
  
        //      Create an object of a file  
        File file = new File(path);  
  
        //      getName() will return the name of the file  
        System.out.println(file.getParent());  
    }  
}
```

Output:

```
C:\Users\Megha\Desktop\MyFiles
```

- **getAbsolutePath():**

The **getAbsolutePath()** method is a part of the file class. This function returns the absolute pathname of the given file object. If the pathname of the file object is absolute then it simply returns the path of the current file object.

```
import java.io.File;  
  
public class Alpha {  
  
    public static void main(String[] args) {  
  
        //      Specify the directory in which file exists
```

```
String path =  
"C:\\Users\\Megha\\Desktop\\MyFiles\\data.txt";  
  
//      Create an object of a file  
File file = new File(path);  
  
System.out.println(file.getAbsolutePath());  
}  
  
}
```

Output:

```
C:\\Users\\Megha\\Desktop\\MyFiles\\data.txt
```

- **isFile()**

The **isFile()** function is a part of the File class in Java. This function determines whether the is a file or Directory denoted by the abstract filename is File or not. The function returns true if the abstract file path is File else returns false.

```
import java.io.File;  
  
public class Alpha {  
  
    public static void main(String[] args) {  
  
        //      Specify the directory in which file exists  
        String path =  
"C:\\Users\\Megha\\Desktop\\MyFiles\\data.txt";  
  
        //      Create an object of a file  
        File file = new File(path);  
  
        System.out.println(file.isFile());  
    }  
}
```

```
    }  
}
```

Output:

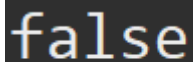
```
true
```

- **isDirectory()**

This function determines whether the is a file or directory denoted by the abstract filename is Directory or not. The function returns true if the abstract file path is Directory else returns false.

```
import java.io.File;  
  
public class Alpha {  
    public static void main(String[] args) {  
        //      Specify the directory in which file exists  
        String path =  
        "C:\\\\Users\\Megha\\Desktop\\MyFiles\\data.txt";  
  
        //      Create an object of a file  
        File file = new File(path);  
  
        System.out.println(file.isDirectory());  
    }  
}
```

Output:



- **createNewFile():**

The createNewFile() method creates a new and empty file with a specified name. This operation succeeded when the name did not yet exist. Checking for the existence of the file and creation of the file are atomic operations.

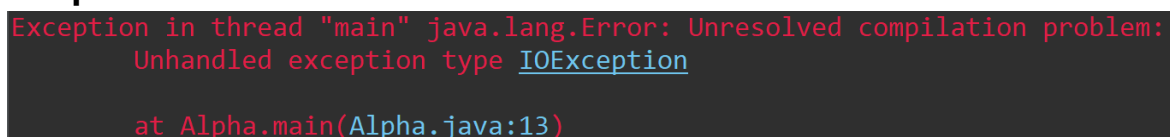
```
import java.io.File;
public class Alpha {
    public static void main(String[] args) {

        //      Specify the directory in which file exists
        String path =
"C:\\Users\\Megha\\Desktop\\MyFiles\\aaa.txt";

        //      Create an object of a file
        File file = new File(path);

        file.createNewFile();
    }
}
```

Output:



Here you are getting IOException because the createNewFile() will throw IOException so you need to handle this exception using throw or try-catch

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

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

        //          Specify the directory in which file exists
        String path =
"C:\\Users\\Megha\\Desktop\\MyFiles\\aaa.txt";

        //          Create an object of a file
        File file = new File(path);

        try {
            file.createNewFile();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

- **mkdir():**

The mkdir() function is used to create a new directory denoted by the abstract pathname. The function returns true if the directory is created else returns false.

```
import java.io.File;
public class Alpha {

    public static void main(String[] args) {

        //          Specify the directory in which file exists
        String path =
"C:\\Users\\Megha\\Desktop\\MyFiles\\aaa.txt";

        //          Create an object of a file
```



```
        File file = new File(path);

        System.out.println(file.mkdir());
    }
}
```

- **list()**

Returns an array of strings naming the files and directories in the directory denoted by this abstract pathname.

```
import java.io.File;

public class Alpha {

    public static void main(String[] args) {

        //      Specify the directory in which file exists
        String path = "C:\\Users\\Megha\\Desktop\\MyFiles";

        //      Create an object of a file
        File file = new File(path);
        String[] list = file.list();
        for (String string : list) {
            System.out.println(string);
        }
    }
}
```

Output:

```
aaa.txt
data.txt
```

- **delete()**

Deletes the file or directory denoted by this abstract pathname. If this pathname denotes a directory, then the directory must be empty in order to be deleted.

```
import java.io.File;

public class Alpha {

    public static void main(String[] args) {

        //          Specify the directory in which file exists
        String path =
"C:\\Users\\Megha\\Desktop\\MyFiles\\aaa.txt";

        //          Create an object of a file
        File file = new File(path);
        System.out.println(file.delete());
    }
}
```

Output:

```
true
```

File Writer:

Java FileWriter class is used to write character-oriented data to a file. It is a character-oriented class which is used for file handling in java.

- Create an object of file writer
- Call write() and pass string to be written inside the file
- Now call the flush() to push the data to the file from the stream

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

public class Alpha {
    public static void main(String[] args) {
        // Specify the directory in which file exists
        String path =
"C:\\Users\\Megha\\Desktop\\MyFiles\\data.txt";

        // Create an object of a file
        File file = new File(path);

        FileWriter writer;

        // Create an object of file writer
        try {
            writer = new FileWriter(file);
            // call the write() and pass the string to be written inside
            the file
            writer.write("Hello World");

            //
            writer.flush();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

Now Hello World String is written inside the data.txt file

- Write a program to take three words from the user and write that inside the file

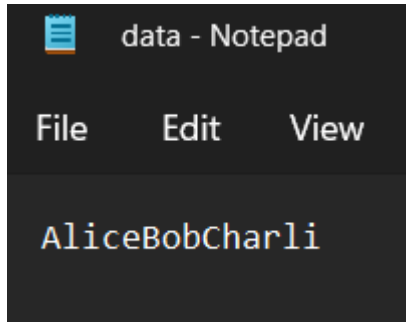
```
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.util.Scanner;

public class Alpha {
    public static void main(String[] args) {
        // Specify the directory in which file exists
        String path =
"C:\\Users\\Megha\\Desktop\\MyFiles\\data.txt";

        // Create an object of a file
        File file = new File(path);
        Scanner scan = new Scanner(System.in);
        FileWriter writer;
        // Create an object of file writer
        try {
            int n1 = scan.nextInt();
            int n2 = scan.nextInt();
            int n3 = scan.nextInt();
            writer = new FileWriter(file);
            writer.write(n1);
            writer.write(n2);
            writer.write(n3);
            writer.flush();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

```
}
```

Output:



The user entered data is stored in a data.txt file.

Now if you run the code again the data which is present will be overridden by the new data.

To overcome this problem you need to append the values to the filewriter, to do this you need to pass one more parameter true which will allow you to append

Alpha.java

```
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.util.Scanner;

public class Alpha {
    public static void main(String[] args) {
        // Specify the directory in which file exists
        String path =
            "C:\\Users\\Megha\\Desktop\\MyFiles\\data.txt";

        // Create an object of a file
        File file = new File(path);
        Scanner scan = new Scanner(System.in);
        FileWriter writer;
```

```
//      Create an object of file writer
try {
    String s1 = scan.next();
    String s2 = scan.next();
    String s3 = scan.next();
    writer = new FileWriter(file, true);
    writer.write(s1);
    writer.write(s2);
    writer.write(s3);
    writer.flush();
} catch (IOException e) {
    e.printStackTrace();
}
finally{
    scan.close();
    writer.close();
}
}
```

Output:

```
AliceBobCharliGameOfThrones
```

Now if you observe from the output, the new input is appended to the existing data.

File Reader:

Java FileReader class is used to read data from the file. It returns data in byte format like FileInputStream class.

It is a character-oriented class which is used for file handling in java.

- Store path of the file in a string
- Create an object of file reader and this will throw an exception of FileNotFoundException if the file is not present in that directory so need to handle that exception
- Call read() which will read the characters present in that file and this method will throw IO Exception need to handle that

```
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
public class Alpha {
    public static void main(String[] args) {
        // Specify the directory in which file exists
        String path =
"C:\\Users\\Megha\\Desktop\\MyFiles\\data.txt";

        FileReader reader = null;

        // Create an object of file reader
        try {
            reader = new FileReader(path);
            System.out.println(reader.read());
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        }
        catch(IOException e1) {
            e1.printStackTrace();
        }
    }
}
```

Output:

65

Here if you observe from the above output, it is giving the ascii value of the first character present inside the file because it reads the character and converts it to integer value. If you want the character instead of integer then you need to do explicit type cast as shown below.

```
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
public class Alpha {
    public static void main(String[] args) {
        // Specify the directory in which file exists
        String path =
"C:\\Users\\Megha\\Desktop\\MyFiles\\data.txt";

        FileReader reader = null;

        // Create an object of file reader
        try {
            reader = new FileReader(path);
            System.out.println((char)reader.read());
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        }
        catch(IOException e1) {
            e1.printStackTrace();
        }
    }
}
```

Output:

A

To read all the string present inside the file you need to create an array using which you can read

```
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
public class Alpha {
    public static void main(String[] args) {
//        Specify the directory in which file exists
        String path =
"C:\\Users\\Megha\\Desktop\\MyFiles\\data.txt";
        FileReader reader = null;
        char[] ar = new char[15];

//        Create an object of file reader
        try {
            reader = new FileReader(path);
            System.out.println(reader.read(ar));
            System.out.println(ar);
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        }
        catch(IOException e1) {
            e1.printStackTrace();
        }
    }
}
```

Output:

13 GameOfThrones

Now as you can see from the above example here you need to define the size of the array in which the data present in the file is stored but if you don't know the number of characters present in the file then you cannot read the complete data present inside the file.

Now to read complete data present in the file make use of `read()` and do this operation repeatedly unless you encounter last character

```
reader = new FileReader(path);
int c = reader.read();
while(c!=-1) {
    System.out.print((char)c);
    c = reader.read();
}
```

Alpha.java

```
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
public class Alpha {
    public static void main(String[] args) {
        // Specify the directory in which file exists
        String path =
            "C:\\Users\\Megha\\Desktop\\MyFiles\\data.txt";
        FileReader reader = null;

        // Create an object of file reader
```

```
try {  
    reader = new FileReader(path);  
    int c = reader.read();  
    while(c!=-1) {  
        System.out.print((char)c);  
        c = reader.read();  
    }  
} catch (FileNotFoundException e) {  
    e.printStackTrace();  
}  
catch(IOException e1) {  
    e1.printStackTrace();  
}  
}  
}
```

Output:

```
ABCDEFGHIJKLMNOPQRS  
INDIA IS MY COUNTRY  
JAVA  
PYTHON  
HTML  
SQL
```

- Now let's create one more file and copy all the data from one file to another file
 - Create an object of filewriter and filereader
 - Read the character present inside one file using read() and write to one more file using write()

```
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;
public class Alpha {
    public static void main(String[] args) {
//        Specify the directory in which file exists
        String path =
"C:\\Users\\Megha\\Desktop\\MyFiles\\data.txt";
        String path1 =
"C:\\Users\\Megha\\Desktop\\MyFiles\\exmp.txt";
        FileReader reader = null;
        FileWriter writer = null;

//        Create an object of file reader and writer
        try {
            reader = new FileReader(path);
            writer = new FileWriter(path1);

//        Read the character from data file and write to exmp file
            int c = reader.read();
            while(c!=-1) {
                writer.write(c);
                c = reader.read();
            }
            writer.flush();
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        }
        catch(IOException e1) {
            e1.printStackTrace();
        }
    }
}
```

```

    }
}
}

```

The data is present in the data file to the exmp file, but here if you observe here we are reading each character one by one which is not efficient when there are billions of characters present in the file.

- Now let's see how to read the data from the file using `BufferedReader`. `BufferedReader` will read line by line.
 - Create an object of file reader
 - Convert file reader to bufferedreader by creating an object of buffered reader
 - Call `readLine()` to read the each line present inside the file

```

import java.io.BufferedReader;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;

public class Alpha {
    public static void main(String[] args) {
        // Specify the directory in which file exists
        String path =
"C:\\Users\\Megha\\Desktop\\MyFiles\\data.txt";
        FileReader reader = null;
        BufferedReader reader2 = nul
        try {
            reader = new FileReader(path);
            reader2 = new BufferedReader(reader);
            String line = reader2.readLine();
            System.out.println(line);
            System.out.println(reader2.readLine());
            System.out.println(reader2.readLine());

```

```
        System.out.println(reader2.readLine());  
        System.out.println(reader2.readLine());  
        System.out.println(reader2.readLine());  
    } catch (FileNotFoundException e) {  
        e.printStackTrace();  
    }  
    catch(IOException e1) {  
        e1.printStackTrace();  
    }  
}  
}
```

Output:

```
ABCDEFGHIJKLMNOPS  
INDIA IS MY COUNTRY  
JAVA  
PYTHON  
HTML  
SQL
```

Here we have called `readLine()` multiple times to read all the lines present inside the file. Instead of that we can make use of a loop and read all lines until the end of the file.

```
import java.io.BufferedReader;
```

```
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
public class Alpha {
    public static void main(String[] args) {
        // Specify the directory in which file exists
        String path =
"C:\\Users\\Megha\\Desktop\\MyFiles\\data.txt";
        FileReader reader = null;
        BufferedReader reader2 = null;

        try {
            reader = new FileReader(path);
            reader2 = new BufferedReader(reader);
            String line = reader2.readLine();
            while(line != null) {
                System.out.println(line);
                line = reader2.readLine();
            }
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        }
        catch(IOException e1) {
            e1.printStackTrace();
        }
    }
}
```

Output:

```
ABCDEFGHIJKLMNOPQRS
INDIA IS MY COUNTRY
JAVA
PYTHON
HTML
SQL
```


- Write a program to count number of lines in the file

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

public class Alpha {
    public static void main(String[] args) {
        // Specify the directory in which file exists
        String path =
"C:\\Users\\Megha\\Desktop\\MyFiles\\data.txt";
        FileReader reader = null;
        BufferedReader reader2 = null;

        try {
            reader = new FileReader(path);
            reader2 = new BufferedReader(reader);
            String line = reader2.readLine();
            int count = 0;
            while(line != null) {
                System.out.println(line);
                count++;
                line = reader2.readLine();
            }
            System.out.println(count);
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        }
        catch(IOException e1) {
            e1.printStackTrace();
        }
    }
}
```

Output:

```
ABCDEFGHIJKLMNOPQRSTUVWXYZ  
INDIA IS MY COUNTRY  
JAVA  
PYTHON  
HTML  
SQL  
6
```

- Write a program to count number of characters in the file

```
import java.io.BufferedReader;  
import java.io.FileNotFoundException;  
import java.io.FileReader;  
import java.io.IOException;  
  
public class Alpha {  
    public static void main(String[] args) {  
        // Specify the directory in which file exists  
        String path =  
"C:\\Users\\Megha\\Desktop\\MyFiles\\data.txt";  
        FileReader reader = null;  
        BufferedReader reader2 = null;  
  
        try {  
            reader = new FileReader(path);  
            reader2 = new BufferedReader(reader);  
            String line = reader2.readLine();  
            int count = 0;  
            int sum = 0;  
            while(line != null) {  
                count++;  
                int l = line.length();  
            }  
        }  
    }  
}
```

```
        sum +=1;
        line = reader2.readLine();
    }
    System.out.println(count);
    System.out.println(sum);
} catch (FileNotFoundException e) {
    e.printStackTrace();
}
catch(IOException e1) {
    e1.printStackTrace();
}
}
}
```

Scenario 1: Write a program to read the name from the file and phone number from another file, now merge two data and write into file in a format shown below

name : Phone Number

- Create a file to write the **name : Phone Number** combination
- create an object of file reader and buffered reader to read each line from the file
- Run a loop which will help to read all lines until end of the file

```
import java.io.BufferedReader;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.FileWriter;
import java.io.IOException;

public class Alpha {
    public static void main(String[] args) {
        // Specify the directory in which file exists
        String path =
"C:\\Users\\Megha\\Desktop\\MyFiles\\name.txt";
        String path1 =
"C:\\Users\\Megha\\Desktop\\MyFiles\\phone.txt";
        String path2 =
"C:\\Users\\Megha\\Desktop\\MyFiles\\phone_book.txt";

        // Creating a reference of FileReader and
        // BufferedReader
        FileReader reader = null;
        BufferedReader reader1 = null;

        FileReader reader2 = null;
        BufferedReader reader3 = null;

        // Creating a reference of FileWriter
```

```
FileWriter writer = null;

try {
    reader = new FileReader(path);
    reader1 = new BufferedReader(reader);
    reader2 = new FileReader(path1);
    reader3 = new BufferedReader(reader2);

    writer = new FileWriter(path2);

    String name = reader1.readLine();
    String phone = reader3.readLine();

    while(name != null && phone != null) {
        writer.write(name + " : " + phone +
"\n");

        name = reader1.readLine();
        phone = reader3.readLine();
    }
    writer.flush();

} catch (FileNotFoundException e) {
    e.printStackTrace();
}
catch(IOException e1) {
    e1.printStackTrace();
}

}
```

Now let's see how you can write into the file using buffered writer

- First Create a path of the file

```
String path =  
"C:\\Users\\Megha\\Desktop\\MyFiles\\data.txt";
```

- Create an object of File Writer

```
FileWriter writer = new FileWriter(path);
```

- Create an object of BufferedWriter

```
BufferedWriter bf = new BufferedWriter(writer);
```

- Using bufferedWriter call write() to write the data into the text

```
bf.write("India");
```

Alpha.java

```
import java.io.BufferedWriter;  
import java.io.FileNotFoundException;  
import java.io.FileWriter;  
import java.io.IOException;  
  
public class Alpha {  
    public static void main(String[] args) {  
        // Specify the directory in which file exists  
        String path =  
"C:\\Users\\Megha\\Desktop\\MyFiles\\data.txt";  
  
        // Create a reference of file writer and buffered  
        writer  
        FileWriter writer = null;  
        BufferedWriter bf = null;
```

```
    try {  
        writer = new FileWriter(path);  
        bf = new BufferedWriter(writer);  
        bf.write("India");  
        bf.flush();  
  
    } catch (FileNotFoundException e) {  
        e.printStackTrace();  
    }  
    catch(IOException e1) {  
        e1.printStackTrace();  
    }  
}  
}
```

- Now let's see to take integer value from the user and write into the file

```
import java.io.BufferedWriter;  
import java.io.FileNotFoundException;  
import java.io.FileWriter;  
import java.io.IOException;  
import java.util.Scanner;  
  
public class Alpha {  
    public static void main(String[] args) {  
        Scanner scan = new Scanner(System.in);  
        int n = scan.nextInt();  
  
        // Specify the directory in which file exists  
        String path =  
        "C:\\Users\\Megha\\Desktop\\MyFiles\\data.txt";  
  
        // Create a reference of file writer and buffered
```

```
writer

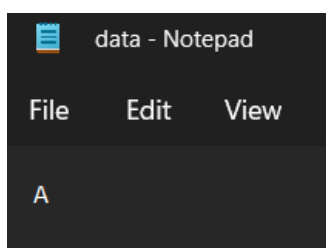
    FileWriter writer = null;
    BufferedWriter bf = null;

    try {
        writer = new FileWriter(path);
        bf = new BufferedWriter(writer);
        bf.write(n);
        bf.flush();

    } catch (FileNotFoundException e) {
        e.printStackTrace();
    }
    catch(IOException e1) {
        e1.printStackTrace();
    }
}

}
```

Output:



As you can see from the above output, the user has given 65 but in the file ascii character of the number is stored, it is because write() inside the buffered writer will write the character into the file.

If you try to read float value from the user and try to write into the file using buffered writer and file writer you will get an error. It is because it can read int, String and character array.

To overcome this you can make use of PrintWriter()

- Create an object of PrintWriter and call print() to write the value into the file

```
import java.io.FileNotFoundException;
import java.io.IOException;
import java.io.PrintWriter;
import java.util.Scanner;

public class Alpha {
    public static void main(String[] args) {
        Scanner scan = new Scanner(System.in);
        float f = scan.nextFloat();

        // Specify the directory in which file exists
        String path =
"C:\\\\Users\\Megha\\Desktop\\MyFiles\\data.txt";

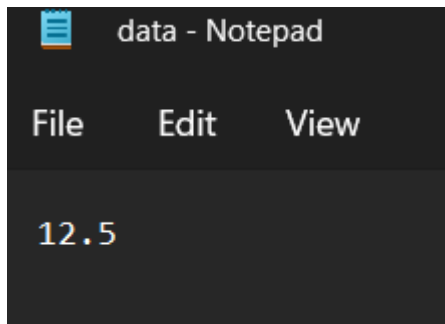
        // Create a reference of file writer and buffered
        writer

        PrintWriter writer = null;

        try {
            writer = new PrintWriter(path);
            writer.print(f);
            writer.flush();
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        }
        catch(IOException e1) {
            e1.printStackTrace();
        }
    }
}
```

```
}  
  
}
```

Output:



As you can see from the above output, successfully float value is written inside the file.

Using `PrintWriter` you can write all different types of values inside the file by calling `print()`.

- Now let's read integer, float, boolean value from the user and write into the file

```
import java.io.FileNotFoundException;  
import java.io.IOException;  
import java.io.PrintWriter;  
import java.util.Scanner;  
  
public class Alpha {  
    public static void main(String[] args) {  
        Scanner scan = new Scanner(System.in);  
        float f = scan.nextFloat();  
        int n = scan.nextInt();  
        boolean b = scan.nextBoolean();  
  
        // Specify the directory in which file exists  
        String path =
```

```
"C:\\Users\\Megha\\Desktop\\MyFiles\\data.txt";

//      Create a reference of file writer and buffered
writer

    PrintWriter writer = null;
    try {
        writer = new PrintWriter(path);
        writer.println(f);
        writer.println(n);
        writer.println(b);
        writer.flush();
    } catch (FileNotFoundException e) {
        e.printStackTrace();
    }
    catch(IOException e1) {
        e1.printStackTrace();
    }
}

}
```

Output:

