Name : Pratyay Dhond

Department : IT

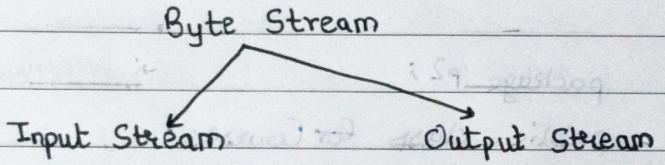
Roll No : 1907011

Assignment No. 04

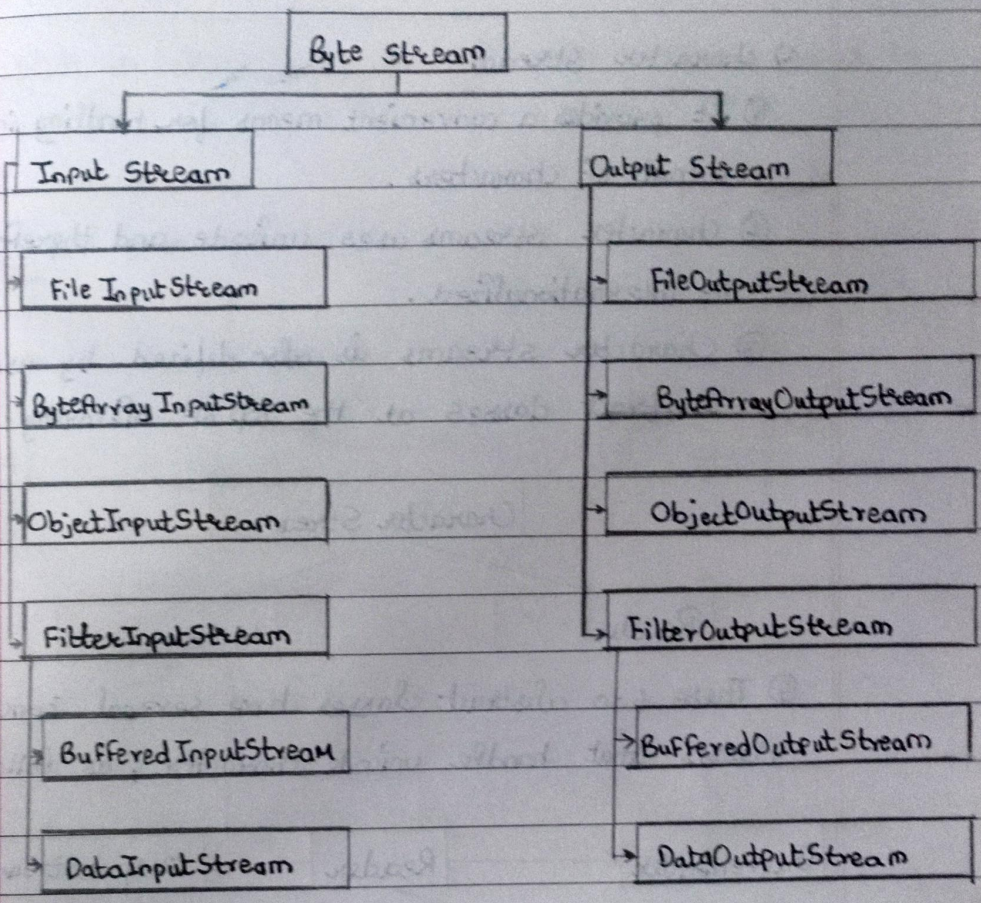
2> Explain Other Stream classes in detail.

1) Java performs I/O through the streams
2) Java encapsulates stream under java.io package.
3) Java defines two types of streams:
   1) Byte stream
   2) Character stream

1) Byte stream :



① Byte streams provide a convenient means for handling input and output of bytes.
② Byte stream is defined by using two abstract classes at the top of hierarchy, they are InputStream and OutputStream.
③ These two abstract classes have several concrete classes that handle various devices such as disk files, network connection, etc.
④ The full hierarchy of classes is as follows:



Full hierarchy of Byte Stream classes

~~2) Cha~~

~~2) Character Stream.~~

Various methods of byteStream classes are :

int read()

int read(byte [] buffer)

int readObject(obj)

void reset()

void close()

void write()

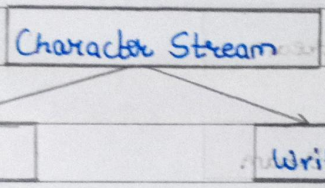
void writeObject

2) Character Stream

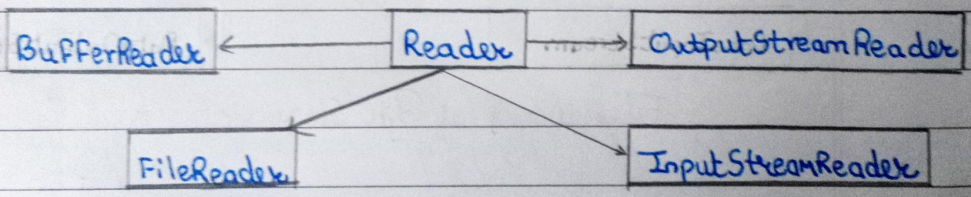
① It provides a convenient means for handling input and output of characters.

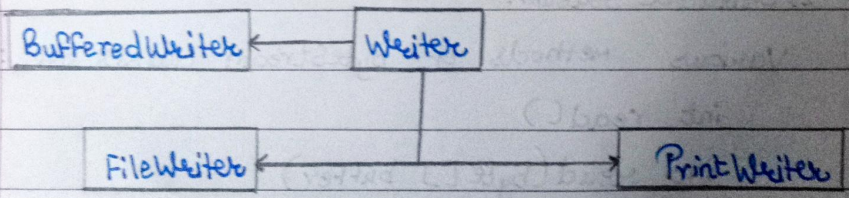
② Character streams uses unicode and therefore can be internationalized.

③ Character streams is also defined by using two abstract classes at the top of hierarchy.



④ These two abstract classes have several concrete classes that handle unicode character, as follows:





⑤ Various functions of character stream classes are:

① read()
② write()
③ open()
④ close()

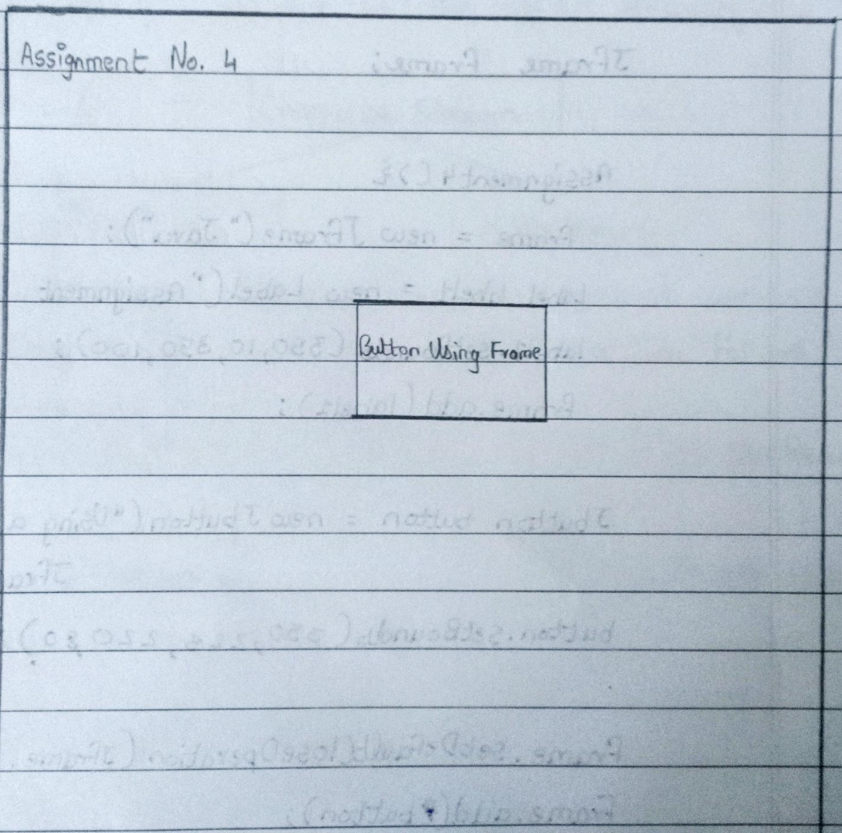
2) Write a simple program to create a frame object in swing.

→

```
import java.swing.*;

public class Assignment4 {

    JFrame frame;

    Assignment4 (){
      frame = new JFrame("Java");
      Label label1 = new Label("Assignment No. 4");
      label1.setBounds (350,10,350,100);
      frame.add (label1);

    Jbutton button = new Jbutton("Using a Jbutton in
                                  JFrame");
    button.SetBounds( 350,225,220,80);

    Frame.setDefaultCloseOperation (JFrame.EXIT_ON-CLOSE);
    Frame.add(button);
    Frame.setSize (900,900);
    Frame.setLayOut (null);
    Frame.setVisible(true);
}
```

```
public static void main (String[] args){
    new Assignment4();
}
}
```

Output :



3. What is concatenation and buffering of files? Explain with examples.

→

Concatenation is the process of concatanating two or more files and saving it in different files.
In Java, by using SequenceInputStream class we can concatenate two or more files.

Buffering:

In java we can create a buffer to store temporary data that is read from and written to a stream and this process known as I/O buffer operation.

A buffer can be created by using following classes:

BufferedInputClass
BufferedOutputClass

Example:

~~class~~
import ~~Javax~~ java.io.*;

public class FileConcat {
public static void main(String[] args){
FileInputStream f1 = new FileInputStream("1.txt");
FileInputStream f2 = new FileInputStream("2.txt");

```
SequenceInputStream si = new SequenceInputStream(f1,f2);

BufferedInputStream ibs = new BufferedInputStream(si);
BufferedOutputStream obs = new BufferedOutputstream(System.out);

int c;
while(( c = ibs.read()) != -1) {
    obs.write((char) c);
  }
 ibs.close();
 obs.close();
 f1.close();
 f2.close();
}
}
```

1.txt

Hello, I am Pratyay Dhond.

2.txt

Java Assignment.

Output on cmd:

```
javac FileConcat.java
java FileConcat
    Hello, I am Pratyay Dhond.
    Java Assigment.
```

4. How to access interface variables ? Justify with answer.

→

An interface in java is a specification of method prototypes. Wherever you need to guide the programmer or, make a contract specifying how the methods and fields ~~on a~~ of a type should be you can define an interface. By default:

1) All the ~~interfaces of an~~ members of an interface are public.
2) All the methods in an interface are public and abstract (except static and default)
3) All the fields of an interface are public, static and, final by default.

Example:

```
interface Trial {
    final float PI = 3.14f;
    int move();
}
```

In the above example, the above variable PI is the interface variable.

5. What are uncaught exception? Give example.

→

The uncaught exceptions are the exceptions that are not caught by the compiler but automatically caught and handled by the java built-in exception handler.

The java programming language has a very strong exception handling mechanism. It also allows us to handle the exception use the keywords like try, catch, finally, throw and throws.

When an uncaught exception occurs, the JVM calls a special private method known dispatchUncaughtException(), on the thread class in which the exception occurs and terminates the thread.

The division by zero exception is one of the example for uncaught ~~exceptions.~~ exceptions.

Example:

```
import java.util.Scanner;

public class Example{
    public static void main(String[] args){
     Scanner sc = new Scanner(System.in);
     System.out.println("Enter the a and b values: ");
     int a = read.nextInt();
     int b = read.nextInt();
```

```
        int c = a/b;
        System.out.println(a + "/" + b + " = " + c);
    }
}
```

CMD

```
Enter a and b values:
25
0
Exception in thread "main" java.lang.Arithmetic: / by zero
        at UncaughtExceptionExample.main
                (UncaughtExceptionExample.java :11)
```