CEG – 4350

OS Internals and Design

SPRING 2018

Name: Aman Ali Pogaku

UID: U00878439

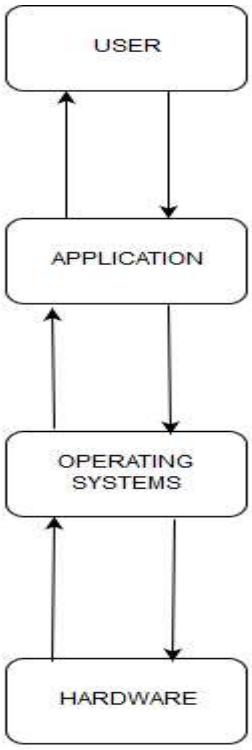
Email: pogaku.6@wright.edu

INDEX

INTRODUCTION

BACKGROUND

An Operating System is computer program that acts as an interface between a user of a computer and the computer hardware. It is system software that is responsible for managing of hardware and software, and helps the whole workflow by providing common services for computer programs.



Operating Systems

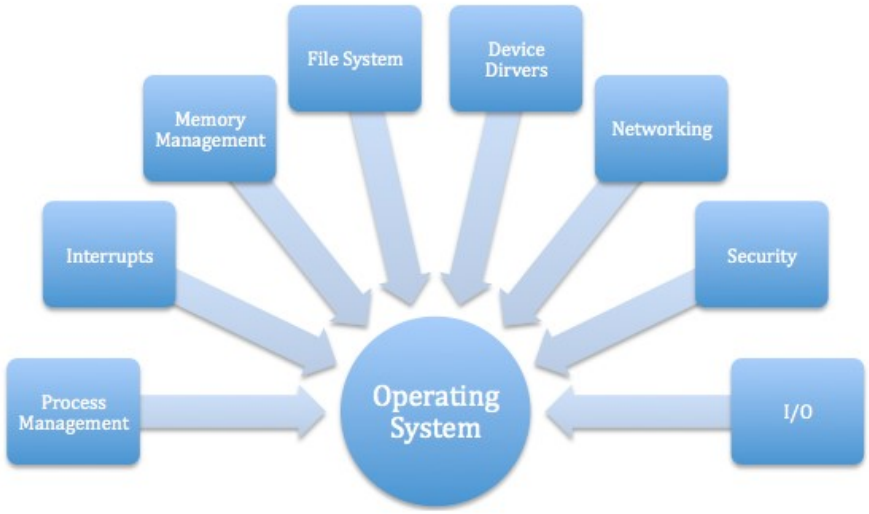
There are three classical views of an operating system. They are:

1) Resource manager: Resource manager manages and protects multiple computer resources: CPU Processes, Internal/External memory, Tasks etc, and Handles and allocates resources to multiple users or multiple programs running at the same time and space.

2) Control program: Control program handles all the components of a complex computer system in an integrated manner and controls the execution of user programs and I/O devices to prevent errors and improper use of computer resources.

3) Command Executer: This interfaces between user and Machine and supplies services and utilities to users.

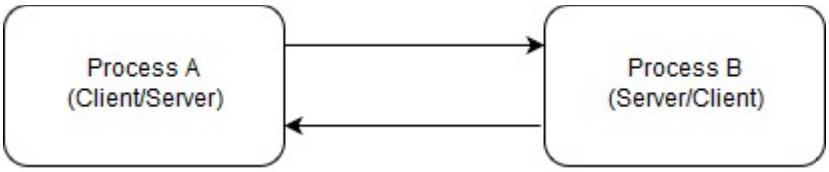
There are two types of programs in an operating system. First one is Application programs, which are built upon the software design provided by an operating system. Application programs are operating system specific and must be developed using that particular operating system and its specifications in mind. Another type of program is called a system program. Systems programs keep the hardware and software running together smoothly.



Various tasks/functions of an operating system

There are many operating systems that have been developed till now and each and every operating system is targeted towards a specific industry. Many a times, a company builds its own customized operating system. For example, GOOGLE uses its own tweak of "UBUNTU Linux" which is a type of Linux operating system. For the general purpose usage of public, the common desktop operating systems are Windows, Mac OS, and Linux.

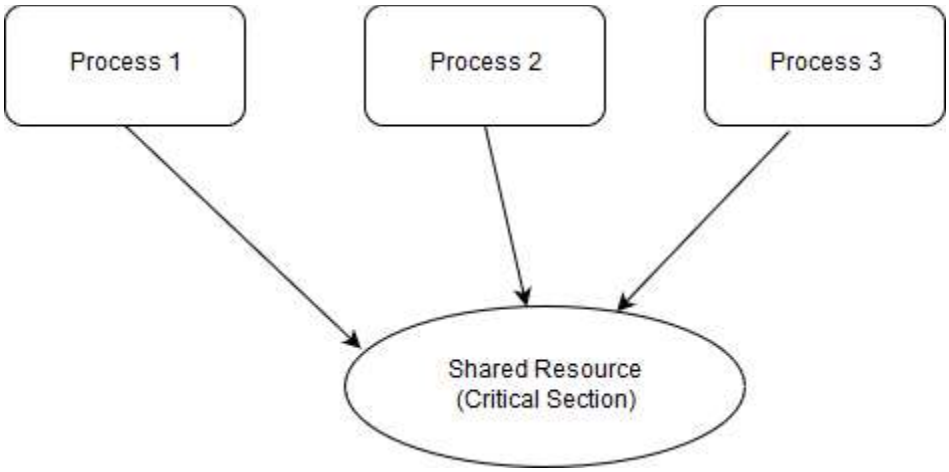
Inter process communication (IPC) refers to the ways in which an operating system allows the processes that it manages to share data. Applications that use IPC are broadly categorized as Clients and Servers. Client's task is to access data from the server and server's task is to serve the required data to the client. These roles can be interchanged by the two processes communicating with each other based on the direction of the data flow.



Inter process communication

Inter process communication (IPC) requires the use of resources, such as memory, which are shared between processes or threads. Synchronization is essential for both single processor and multi processor systems. Thread synchronization is defined as a method which takes care that two or more concurrent processes or threads do not simultaneously execute a particular program segment defined as critical section.

If proper method is not implemented to correctly coordinate or synchronize access to shared resources; numerous problems can arise. It is indispensable to use the synchronization concept in order to avoid the problems produced with inter process communication.



Producer-Consumer is one of the classic examples of multi process synchronization problem. There are two processes in this problem. A producer and a consumer and they share a common buffer. Producer's job is to produce data to the buffer and the consumer's task is to consume that data from the buffer. The problem arises when producer tries to add data after the buffer is full and the consumer consumes the data after the buffer is empty.

This project aims to solve this problem in various methods, which will be discussed one by one.

Tools used to implement the project

1) Eclipse: Eclipse is an integrated development environment used by software engineers to create softwares and even use it for debugging the code. It can used to program in any of the popular programming languages like Java, C++, JavaScript.
   Many of its tools are open source and Its navigation is very easy. It even has inbuilt syntax checking.

2) Java Development Kit: Java development kit is used to install the java platform on the operating system. The advantage of java is that it can be executed on any operating system since it requires only the java platform which is available on many operating systems. Its vast library, open source availability, and proper enrich explorative documentation makes it very easy for programmers to realize any task with ease.

3) Windows operating system: Windows operating system is developed by Microsoft. It is widely used because of its simple interface, when compared to other operating systems.

4) Draw.io: "https://www.draw.io/" is the website I have used to draw the diagrams that appear in this documentation.

5) Microsoft Word: Microsoft word is one of the applications in the proprietary Microsoft Office suite. I have used Microsoft word for documenting this report. It even has features to convert the document to a PDF file which eases the process of publishing.

6) BlueJ IDE: I have used this IDE to generate the UML diagrams for the program

METHOD ONE FOR INTERPROCESS COMMUNICATION

BETWEEN PRODUCER AND CONSUMER

Method

Use a Pipe to transfer 100 data from the producer to the consumer.

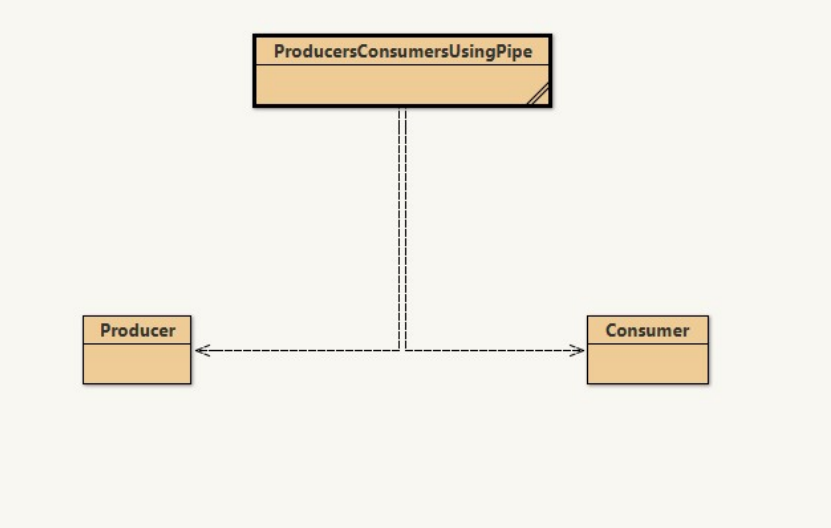
Theory

1) A pipe acts like a connection between an input stream and an output stream.
2) I/O based on pipes is a simple concept. We create two streams, and the producers writes the data to the stream and the consumer reads the data from the stream.

Approach to solve the question

1) Create two classes.
2) The task of the first class is to write the data to the stream and second class is to read the data from the stream
3) In both the classes, data is printed as and when it is written/read.
4) There is a main program that will be calling both the classes. It will be clear in the UML Diagram.
5) For few inbuilt methods, code is commented with explanations taken from the official java documentation for thorough understanding and serving as a good reference in future.

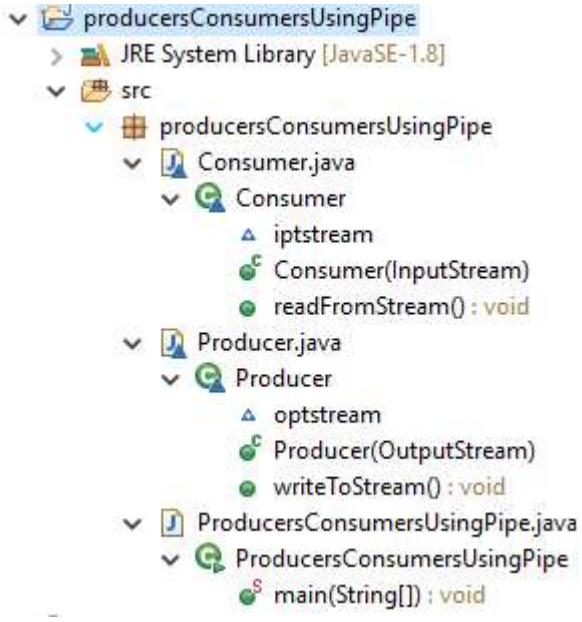
Unified Modeling Language diagram



The Diagram above showing the main

program calling the two classes.

Folder Organization



1) Have separate class files. They will help in maintaining the code very easily.

Source Code

1) Source code for Producer.java :

```
package producersConsumersUsingPipe;
import java.io.OutputStream;

public class Producer {
   /* this class contains producer's code. verification and write data to the stream*/
   OutputStream optstream;
   /*ostream is a variable name*/
   /*This abstract class is the uperclass of all classes representing an output stream of
bytes. An output stream accepts output bytes and sends them to some sink.
Applications that need to define a subclass of OutputStream must always provide at least
a method that writes one byte of output.*/

   public Producer(OutputStream optstream) {
      this.optstream = optstream;
      /*this  is a keyword in Java. It can be used inside the Method or constructor of
Class. It(this) works as a reference to the current Object whose Method or constructor is
being invoked. The this keyword can be used to refer to any member of the current object
from within an instance Method or a constructor*/
```

```
    }
    public void writeToStream() {

        int i = 1;
        for (i = 1; i <= 100; i++) {

            byte b = (byte) i; /*type casting*/
            System.out.println("Producer is now writing integer " + b + " on the stream");

            try {
                /*Write The data to the Stream*/
                optstream.write(b);
                Thread.sleep(500);
                /*
                A thread is a thread of execution in a program. The Java Virtual Machine
allows an application to have multiple threads of execution running concurrently. Causes
the currently executing thread to sleep (temporarily cease execution) for the specified
number of milliseconds, subject to the precision and accuracy of system timers and
schedulers. The thread does not lose ownership of any monitors.
                */
                if (i > 100) {
                    optstream.write((byte) - 1); /*-1 indicates the end of file*/
                }
            } catch (Exception e) {
                System.out.println(e);
            } // to catch an exception in case it is found
        }
    }
}
```

2) Source code for Consumer.java :

```
package producersConsumersUsingPipe;

import java.io.InputStream;

public class Consumer {
    InputStream iptstream;
    /*This abstract class is the superclass of all classes representing an input stream of
bytes.
```

```
    Applications that need to define a subclass of InputStream must always provide a
method that returns the next byte of input.
    */

    public Consumer(InputStream iptstream) {
        this.iptstream = iptstream;
    }


    public void readFromStream() {
        do {
            int filePointer = -1;
            try {
                filePointer = iptstream.read();
                byte b = (byte) filePointer;
                if (b < 0)
                    break; /* break if the stream is empty*/
                System.out.println("Consumer is now reading integer " + b + " from the
stream");
                Thread.sleep(500);
                /*Causes the currently executing thread to sleep (temporarily cease execution)
for the specified number of milliseconds, subject to the precision and accuracy of system
timers and schedulers. The thread does not lose ownership of any monitors.*/
            } catch (Exception e) {
                System.out.println(e);
            }

        } while (true);

    }
}
```

3) Source code for ProducersConsumersUsingPipe.java:

```
package producersConsumersUsingPipe; /* creating the package name*/

import java.io.*; /*this statement means all the classes of io package will be
imported.used when we are using input/output stream*/

public class ProducersConsumersUsingPipe {
```

```
   public static void main(String args[]) {
      try {
         /*creating a Output object.*/
         PipedOutputStream outputObject = new PipedOutputStream();
         /*A piped output stream can be connected to a piped input stream to create a
communications pipe.*/

         /*Creating a PipedInputStream object.*/
         PipedInputStream inputObject = new PipedInputStream(outputObject);
         /*The piped input stream then provides whatever data bytes are written to the
piped output stream.*/

         /* Creating a Producer object.*/
         Producer producerObject = new Producer(outputObject); /*sending that object
into the producer class*/
         producerObject.writeToStream(); /*executing the method which writes data to
stream*/

         /* Creating a Consumer object.*/
         Consumer consumerObject = new Consumer(inputObject); /*sending that object
into the consumer class*/
         consumerObject.readFromStream(); /*executing the method which reads data
from the stream*/
      } catch (IOException e) {
         System.out.println(e);
      } /*Since we are dealing with streams exceptions are in IO.*/
   }

}
```

Explanation for the code

1) Initially pipes are connected.
2) After the pipes are connected, we pass those parameters to the producer and consumer respectively.
3) Producer writes the data to the stream.
4) Then the main program calls consumer class. Consumer class reads the data from the file.
5) Outputs are printed on the console.

Result

```
Producer is now writing integer 1 on the stream
Producer is now writing integer 2 on the stream
Producer is now writing integer 3 on the stream
Producer is now writing integer 4 on the stream
Producer is now writing integer 5 on the stream
Producer is now writing integer 6 on the stream
Producer is now writing integer 7 on the stream
Producer is now writing integer 8 on the stream
Producer is now writing integer 9 on the stream
Producer is now writing integer 10 on the stream
Producer is now writing integer 11 on the stream
Producer is now writing integer 12 on the stream
Producer is now writing integer 13 on the stream
Producer is now writing integer 14 on the stream
Producer is now writing integer 15 on the stream
Producer is now writing integer 16 on the stream
Producer is now writing integer 17 on the stream
Producer is now writing integer 18 on the stream
Producer is now writing integer 19 on the stream
Producer is now writing integer 20 on the stream
Producer is now writing integer 21 on the stream
Producer is now writing integer 22 on the stream
Producer is now writing integer 23 on the stream
Producer is now writing integer 24 on the stream
Producer is now writing integer 25 on the stream
Producer is now writing integer 26 on the stream
Producer is now writing integer 27 on the stream
Producer is now writing integer 28 on the stream
Producer is now writing integer 29 on the stream
Producer is now writing integer 30 on the stream
Producer is now writing integer 31 on the stream
Producer is now writing integer 32 on the stream
Producer is now writing integer 33 on the stream
Producer is now writing integer 34 on the stream
Producer is now writing integer 35 on the stream
Producer is now writing integer 36 on the stream
Producer is now writing integer 37 on the stream
Producer is now writing integer 38 on the stream
```

Producer is now writing integer 39 on the stream
Producer is now writing integer 40 on the stream
Producer is now writing integer 41 on the stream
Producer is now writing integer 42 on the stream
Producer is now writing integer 43 on the stream
Producer is now writing integer 44 on the stream
Producer is now writing integer 45 on the stream
Producer is now writing integer 46 on the stream
Producer is now writing integer 47 on the stream
Producer is now writing integer 48 on the stream
Producer is now writing integer 49 on the stream
Producer is now writing integer 50 on the stream
Producer is now writing integer 51 on the stream
Producer is now writing integer 52 on the stream
Producer is now writing integer 53 on the stream
Producer is now writing integer 54 on the stream
Producer is now writing integer 55 on the stream
Producer is now writing integer 56 on the stream
Producer is now writing integer 57 on the stream
Producer is now writing integer 58 on the stream
Producer is now writing integer 59 on the stream
Producer is now writing integer 60 on the stream
Producer is now writing integer 61 on the stream
Producer is now writing integer 62 on the stream
Producer is now writing integer 63 on the stream
Producer is now writing integer 64 on the stream
Producer is now writing integer 65 on the stream
Producer is now writing integer 66 on the stream
Producer is now writing integer 67 on the stream
Producer is now writing integer 68 on the stream
Producer is now writing integer 69 on the stream
Producer is now writing integer 70 on the stream
Producer is now writing integer 71 on the stream
Producer is now writing integer 72 on the stream
Producer is now writing integer 73 on the stream
Producer is now writing integer 74 on the stream
Producer is now writing integer 75 on the stream
Producer is now writing integer 76 on the stream
Producer is now writing integer 77 on the stream
Producer is now writing integer 78 on the stream

Producer is now writing integer 79 on the stream
Producer is now writing integer 80 on the stream
Producer is now writing integer 81 on the stream
Producer is now writing integer 82 on the stream
Producer is now writing integer 83 on the stream
Producer is now writing integer 84 on the stream
Producer is now writing integer 85 on the stream
Producer is now writing integer 86 on the stream
Producer is now writing integer 87 on the stream
Producer is now writing integer 88 on the stream
Producer is now writing integer 89 on the stream
Producer is now writing integer 90 on the stream
Producer is now writing integer 91 on the stream
Producer is now writing integer 92 on the stream
Producer is now writing integer 93 on the stream
Producer is now writing integer 94 on the stream
Producer is now writing integer 95 on the stream
Producer is now writing integer 96 on the stream
Producer is now writing integer 97 on the stream
Producer is now writing integer 98 on the stream
Producer is now writing integer 99 on the stream
Producer is now writing integer 100 on the stream
Consumer is now reading integer 1 from the stream
Consumer is now reading integer 2 from the stream
Consumer is now reading integer 3 from the stream
Consumer is now reading integer 4 from the stream
Consumer is now reading integer 5 from the stream
Consumer is now reading integer 6 from the stream
Consumer is now reading integer 7 from the stream
Consumer is now reading integer 8 from the stream
Consumer is now reading integer 9 from the stream
Consumer is now reading integer 10 from the stream
Consumer is now reading integer 11 from the stream
Consumer is now reading integer 12 from the stream
Consumer is now reading integer 13 from the stream
Consumer is now reading integer 14 from the stream
Consumer is now reading integer 15 from the stream
Consumer is now reading integer 16 from the stream
Consumer is now reading integer 17 from the stream
Consumer is now reading integer 18 from the stream

Consumer is now reading integer 19 from the stream
Consumer is now reading integer 20 from the stream
Consumer is now reading integer 21 from the stream
Consumer is now reading integer 22 from the stream
Consumer is now reading integer 23 from the stream
Consumer is now reading integer 24 from the stream
Consumer is now reading integer 25 from the stream
Consumer is now reading integer 26 from the stream
Consumer is now reading integer 27 from the stream
Consumer is now reading integer 28 from the stream
Consumer is now reading integer 29 from the stream
Consumer is now reading integer 30 from the stream
Consumer is now reading integer 31 from the stream
Consumer is now reading integer 32 from the stream
Consumer is now reading integer 33 from the stream
Consumer is now reading integer 34 from the stream
Consumer is now reading integer 35 from the stream
Consumer is now reading integer 36 from the stream
Consumer is now reading integer 37 from the stream
Consumer is now reading integer 38 from the stream
Consumer is now reading integer 39 from the stream
Consumer is now reading integer 40 from the stream
Consumer is now reading integer 41 from the stream
Consumer is now reading integer 42 from the stream
Consumer is now reading integer 43 from the stream
Consumer is now reading integer 44 from the stream
Consumer is now reading integer 45 from the stream
Consumer is now reading integer 46 from the stream
Consumer is now reading integer 47 from the stream
Consumer is now reading integer 48 from the stream
Consumer is now reading integer 49 from the stream
Consumer is now reading integer 50 from the stream
Consumer is now reading integer 51 from the stream
Consumer is now reading integer 52 from the stream
Consumer is now reading integer 53 from the stream
Consumer is now reading integer 54 from the stream
Consumer is now reading integer 55 from the stream
Consumer is now reading integer 56 from the stream
Consumer is now reading integer 57 from the stream
Consumer is now reading integer 58 from the stream

Consumer is now reading integer 59 from the stream
Consumer is now reading integer 60 from the stream
Consumer is now reading integer 61 from the stream
Consumer is now reading integer 62 from the stream
Consumer is now reading integer 63 from the stream
Consumer is now reading integer 64 from the stream
Consumer is now reading integer 65 from the stream
Consumer is now reading integer 66 from the stream
Consumer is now reading integer 67 from the stream
Consumer is now reading integer 68 from the stream
Consumer is now reading integer 69 from the stream
Consumer is now reading integer 70 from the stream
Consumer is now reading integer 71 from the stream
Consumer is now reading integer 72 from the stream
Consumer is now reading integer 73 from the stream
Consumer is now reading integer 74 from the stream
Consumer is now reading integer 75 from the stream
Consumer is now reading integer 76 from the stream
Consumer is now reading integer 77 from the stream
Consumer is now reading integer 78 from the stream
Consumer is now reading integer 79 from the stream
Consumer is now reading integer 80 from the stream
Consumer is now reading integer 81 from the stream
Consumer is now reading integer 82 from the stream
Consumer is now reading integer 83 from the stream
Consumer is now reading integer 84 from the stream
Consumer is now reading integer 85 from the stream
Consumer is now reading integer 86 from the stream
Consumer is now reading integer 87 from the stream
Consumer is now reading integer 88 from the stream
Consumer is now reading integer 89 from the stream
Consumer is now reading integer 90 from the stream
Consumer is now reading integer 91 from the stream
Consumer is now reading integer 92 from the stream
Consumer is now reading integer 93 from the stream
Consumer is now reading integer 94 from the stream
Consumer is now reading integer 95 from the stream
Consumer is now reading integer 96 from the stream
Consumer is now reading integer 97 from the stream
Consumer is now reading integer 98 from the stream

Consumer is now reading integer 99 from the stream
Consumer is now reading integer 100 from the stream

METHOD TWO FOR INTERPROCESS COMMUNICATION BETWEEN PRODUCER AND CONSUMER

Method

Use either the direct message passing or indirect message passing (using a mailbox, a message queue, or a similar one) to transfer 100 data from the producer to the consumer.

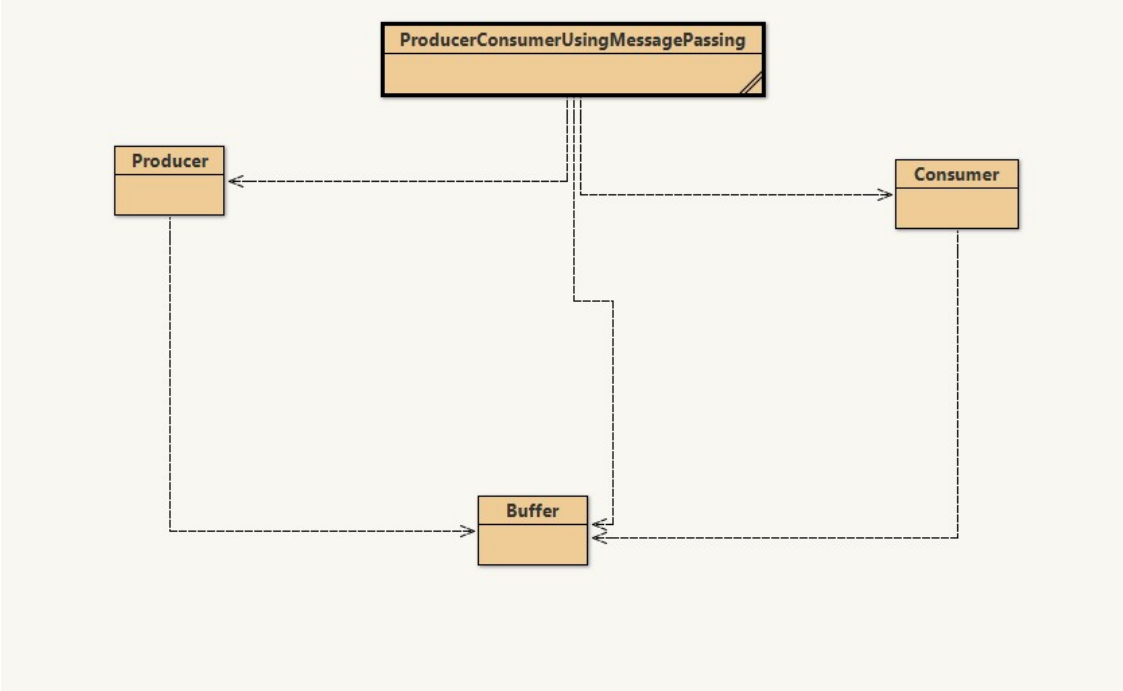
Theory

1) A buffer queue will be used in this method.

Approach to solve the question

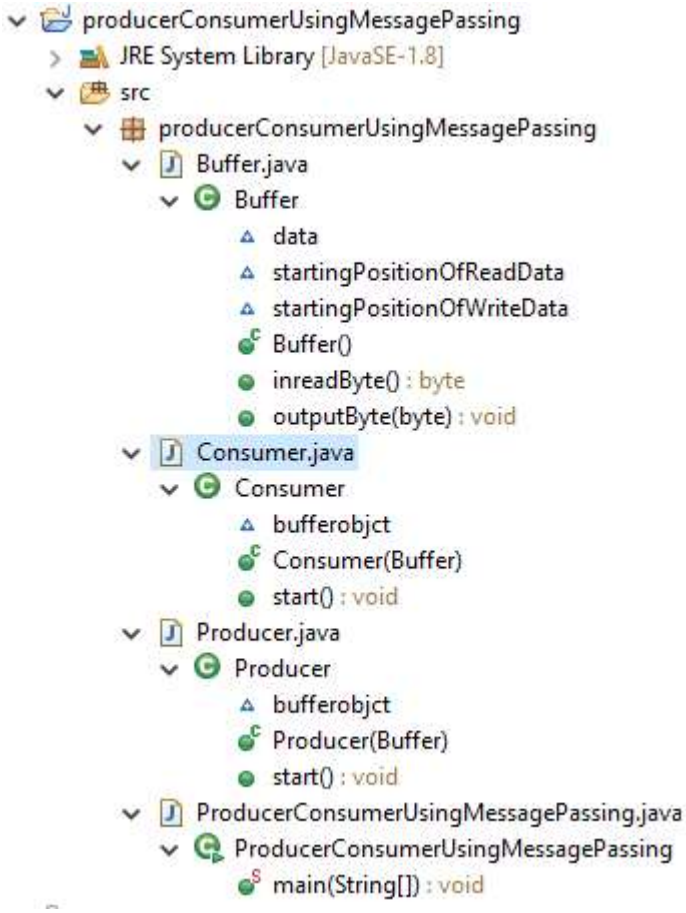
1) There are overall three classes apart from the main program.
2) The producer class communicates with the buffer class and writes the data to the buffer.
3) The consumer class communicates with the buffer class after reads the data from the buffer.
4) The classes intercommunication will be clear in the UML Diagram.

Unified Modeling Language Diagram



Interaction between classes.

Folder Structure



Source Code

1) Source code for Buffer.java:

```
package producerConsumerUsingMessagePassing;


public class Buffer {

   byte[] data;

   int startingPositionOfWriteData = 0;

   int startingPositionOfReadData = 0;
```

```
    public Buffer() {
        // we assigning the space.
        data = new byte[500];
    }

    public void outputByte(byte db) {
        //write the data
        data[startingPositionOfWriteData++] = db;
    }

    public byte inreadByte() {
        // read the data
        byte d = data[startingPositionOfReadData++];
        return d;
    }

}
```

2) Source code for Producer.java:

```
package producerConsumerUsingMessagePassing;

public class Producer {
    Buffer bufferobjct; /* Reference to the Buffer class. creating its object.*/
    public Producer(Buffer bufferobjct) {
        this.bufferobjct = bufferobjct;
```

```
    }
    public void start() {
        /*Writing 100 integers*/


        try {
            int i = 100;
            for (int j = 1; j <= i; j++) {
                byte b = (byte) j;
                System.out.println("Producer is now writing integer " + b + " on the buffer");


                // Write integer to buffer .
                bufferobjct.outputByte(b);
                Thread.sleep(500); /*sleep for the specified time*/
                if (j >= 100)
                    bufferobjct.outputByte((byte) - 1); /*Signalling  End of file if the number crosses
100*/
            }
        } catch (Exception e) {}
    }
}
```

3) Source code for Consumer.java :

```
package producerConsumerUsingMessagePassing;


public class Consumer {
```

```
Buffer bufferobjct;

public Consumer(Buffer bufferobjct) {
    this.bufferobjct = bufferobjct;

}

public void start() {
    // read byte values upto end of the data.

    while (true) {
        // Get the next byte
        byte b = bufferobjct.inreadByte();

        // Check if end-of-data.
        if (b < 0) break;
        System.out.println("Consumer is now reading integer " + b + " from the buffer");
        try {
            Thread.sleep(500);
        } catch (InterruptedException x) { /*Incase of buffers, above is the exceptions we get*/
            System.out.println(x);
        }
    }
}
```

```
}
```

4) Source code for ProducerConsumerUsingMessagePassing.java :

```
package producerConsumerUsingMessagePassing;

public class ProducerConsumerUsingMessagePassing {

   public static void main(String[] argv) {
      /*Create an object from class Buffer*/
      Buffer bufferObject = new Buffer();

      /*Create an object from class producer*/

      Producer producerObject = new Producer(bufferObject);
      producerObject.start();

      /*Create an object from class*/
      Consumer consumerObject = new Consumer(bufferObject);
      consumerObject.start();
   }

}
```

Result:

Producer is now writing integer 1 on the buffer

Producer is now writing integer 2 on the buffer

Producer is now writing integer 3 on the buffer

Producer is now writing integer 4 on the buffer

Producer is now writing integer 5 on the buffer

Producer is now writing integer 6 on the buffer

Producer is now writing integer 7 on the buffer

Producer is now writing integer 8 on the buffer

Producer is now writing integer 9 on the buffer

Producer is now writing integer 10 on the buffer

Producer is now writing integer 11 on the buffer

Producer is now writing integer 12 on the buffer

Producer is now writing integer 13 on the buffer

Producer is now writing integer 14 on the buffer

Producer is now writing integer 15 on the buffer

Producer is now writing integer 16 on the buffer

Producer is now writing integer 17 on the buffer

Producer is now writing integer 18 on the buffer

Producer is now writing integer 19 on the buffer

Producer is now writing integer 20 on the buffer

Producer is now writing integer 21 on the buffer

Producer is now writing integer 22 on the buffer

Producer is now writing integer 23 on the buffer

Producer is now writing integer 24 on the buffer

Producer is now writing integer 25 on the buffer

Producer is now writing integer 26 on the buffer

Producer is now writing integer 27 on the buffer

Producer is now writing integer 28 on the buffer

Producer is now writing integer 29 on the buffer

Producer is now writing integer 30 on the buffer

Producer is now writing integer 31 on the buffer

Producer is now writing integer 32 on the buffer

Producer is now writing integer 33 on the buffer

Producer is now writing integer 34 on the buffer

Producer is now writing integer 35 on the buffer

Producer is now writing integer 36 on the buffer

Producer is now writing integer 37 on the buffer

Producer is now writing integer 38 on the buffer

Producer is now writing integer 39 on the buffer

Producer is now writing integer 40 on the buffer

Producer is now writing integer 41 on the buffer

Producer is now writing integer 42 on the buffer

Producer is now writing integer 43 on the buffer

Producer is now writing integer 44 on the buffer

Producer is now writing integer 45 on the buffer

Producer is now writing integer 46 on the buffer

Producer is now writing integer 47 on the buffer

Producer is now writing integer 48 on the buffer

Producer is now writing integer 49 on the buffer

Producer is now writing integer 50 on the buffer

Producer is now writing integer 51 on the buffer

Producer is now writing integer 52 on the buffer

Producer is now writing integer 53 on the buffer

Producer is now writing integer 54 on the buffer

Producer is now writing integer 55 on the buffer

Producer is now writing integer 56 on the buffer

Producer is now writing integer 57 on the buffer

Producer is now writing integer 58 on the buffer

Producer is now writing integer 59 on the buffer

Producer is now writing integer 60 on the buffer

Producer is now writing integer 61 on the buffer

Producer is now writing integer 62 on the buffer

Producer is now writing integer 63 on the buffer

Producer is now writing integer 64 on the buffer

Producer is now writing integer 65 on the buffer

Producer is now writing integer 66 on the buffer

Producer is now writing integer 67 on the buffer

Producer is now writing integer 68 on the buffer

Producer is now writing integer 69 on the buffer

Producer is now writing integer 70 on the buffer

Producer is now writing integer 71 on the buffer

Producer is now writing integer 72 on the buffer

Producer is now writing integer 73 on the buffer

Producer is now writing integer 74 on the buffer

Producer is now writing integer 75 on the buffer

Producer is now writing integer 76 on the buffer

Producer is now writing integer 77 on the buffer

Producer is now writing integer 78 on the buffer

Producer is now writing integer 79 on the buffer

Producer is now writing integer 80 on the buffer

Producer is now writing integer 81 on the buffer

Producer is now writing integer 82 on the buffer

Producer is now writing integer 83 on the buffer

Producer is now writing integer 84 on the buffer

Producer is now writing integer 85 on the buffer

Producer is now writing integer 86 on the buffer

Producer is now writing integer 87 on the buffer

Producer is now writing integer 88 on the buffer

Producer is now writing integer 89 on the buffer

Producer is now writing integer 90 on the buffer

Producer is now writing integer 91 on the buffer

Producer is now writing integer 92 on the buffer

Producer is now writing integer 93 on the buffer

Producer is now writing integer 94 on the buffer

Producer is now writing integer 95 on the buffer

Producer is now writing integer 96 on the buffer

Producer is now writing integer 97 on the buffer

Producer is now writing integer 98 on the buffer

Producer is now writing integer 99 on the buffer

Producer is now writing integer 100 on the buffer

Consumer is now reading integer 1 from the buffer

Consumer is now reading integer 2 from the buffer

Consumer is now reading integer 3 from the buffer

Consumer is now reading integer 4 from the buffer

Consumer is now reading integer 5 from the buffer

Consumer is now reading integer 6 from the buffer

Consumer is now reading integer 7 from the buffer

Consumer is now reading integer 8 from the buffer

Consumer is now reading integer 9 from the buffer

Consumer is now reading integer 10 from the buffer

Consumer is now reading integer 11 from the buffer

Consumer is now reading integer 12 from the buffer

Consumer is now reading integer 13 from the buffer

Consumer is now reading integer 14 from the buffer

Consumer is now reading integer 15 from the buffer

Consumer is now reading integer 16 from the buffer

Consumer is now reading integer 17 from the buffer

Consumer is now reading integer 18 from the buffer

Consumer is now reading integer 19 from the buffer

Consumer is now reading integer 20 from the buffer

Consumer is now reading integer 21 from the buffer

Consumer is now reading integer 22 from the buffer

Consumer is now reading integer 23 from the buffer

Consumer is now reading integer 24 from the buffer

Consumer is now reading integer 25 from the buffer

Consumer is now reading integer 26 from the buffer

Consumer is now reading integer 27 from the buffer

Consumer is now reading integer 28 from the buffer

Consumer is now reading integer 29 from the buffer

Consumer is now reading integer 30 from the buffer

Consumer is now reading integer 31 from the buffer

Consumer is now reading integer 32 from the buffer

Consumer is now reading integer 33 from the buffer

Consumer is now reading integer 34 from the buffer

Consumer is now reading integer 35 from the buffer

Consumer is now reading integer 36 from the buffer

Consumer is now reading integer 37 from the buffer

Consumer is now reading integer 38 from the buffer

Consumer is now reading integer 39 from the buffer

Consumer is now reading integer 40 from the buffer

Consumer is now reading integer 41 from the buffer

Consumer is now reading integer 42 from the buffer

Consumer is now reading integer 43 from the buffer

Consumer is now reading integer 44 from the buffer

Consumer is now reading integer 45 from the buffer

Consumer is now reading integer 46 from the buffer

Consumer is now reading integer 47 from the buffer

Consumer is now reading integer 48 from the buffer

Consumer is now reading integer 49 from the buffer

Consumer is now reading integer 50 from the buffer

Consumer is now reading integer 51 from the buffer

Consumer is now reading integer 52 from the buffer

Consumer is now reading integer 53 from the buffer

Consumer is now reading integer 54 from the buffer

Consumer is now reading integer 55 from the buffer

Consumer is now reading integer 56 from the buffer

Consumer is now reading integer 57 from the buffer

Consumer is now reading integer 58 from the buffer

Consumer is now reading integer 59 from the buffer

Consumer is now reading integer 60 from the buffer

Consumer is now reading integer 61 from the buffer

Consumer is now reading integer 62 from the buffer

Consumer is now reading integer 63 from the buffer

Consumer is now reading integer 64 from the buffer

Consumer is now reading integer 65 from the buffer

Consumer is now reading integer 66 from the buffer

Consumer is now reading integer 67 from the buffer

Consumer is now reading integer 68 from the buffer

Consumer is now reading integer 69 from the buffer

Consumer is now reading integer 70 from the buffer

Consumer is now reading integer 71 from the buffer

Consumer is now reading integer 72 from the buffer

Consumer is now reading integer 73 from the buffer

Consumer is now reading integer 74 from the buffer

Consumer is now reading integer 75 from the buffer

Consumer is now reading integer 76 from the buffer

Consumer is now reading integer 77 from the buffer

Consumer is now reading integer 78 from the buffer

Consumer is now reading integer 79 from the buffer

Consumer is now reading integer 80 from the buffer

Consumer is now reading integer 81 from the buffer

Consumer is now reading integer 82 from the buffer

Consumer is now reading integer 83 from the buffer

Consumer is now reading integer 84 from the buffer

Consumer is now reading integer 85 from the buffer

Consumer is now reading integer 86 from the buffer

Consumer is now reading integer 87 from the buffer

Consumer is now reading integer 88 from the buffer

Consumer is now reading integer 89 from the buffer

Consumer is now reading integer 90 from the buffer

Consumer is now reading integer 91 from the buffer

Consumer is now reading integer 92 from the buffer

Consumer is now reading integer 93 from the buffer

Consumer is now reading integer 94 from the buffer

Consumer is now reading integer 95 from the buffer

Consumer is now reading integer 96 from the buffer

Consumer is now reading integer 97 from the buffer

Consumer is now reading integer 98 from the buffer

Consumer is now reading integer 99 from the buffer

Consumer is now reading integer 100 from the buffer

METHOD THREE FOR INTERPROCESS COMMUNICATION BETWEEN PRODUCER AND CONSUMER

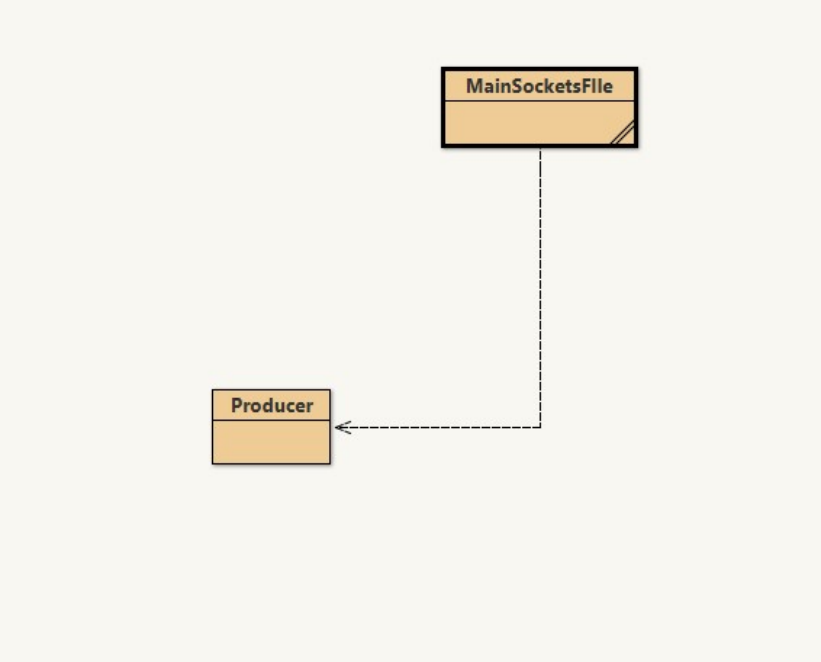
Method

Use the sockets to transfer 100 data from the producer to the consumer.

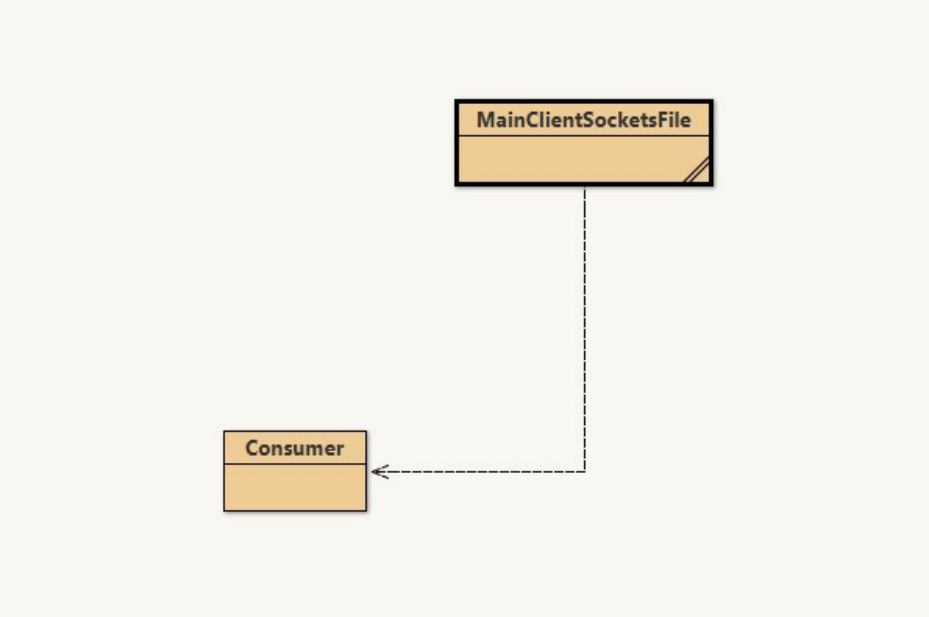
Theory

1) When a client wants to open a TCP/IP connection to a server, it does it using Java Socket.
2) We need to have socket to socket connection.
3) We need to create a socket using java.net.Socket and connect it to server.
4) We will have two files in this method and overall of 4 classes.
5) Producer will write the data as the server, and Consumer will consume the data as the client.

Unified Modeling Language Diagram



ServerSockets class design



ClientSockets class design

Source code for sockets program

1) MainSocketsFIle.java:

```
import java.io.IOException;
import java.io.OutputStream;
import java.net.InetAddress;
import java.net.ServerSocket;
import java.net.Socket;
public class MainSocketsFIle {
   public static void main(String args[]) {
      try {
         /*I am creating the ServerSocket with 8000 port number*/
         ServerSocket serverSocket = new ServerSocket(8000);
         System.out.println("Producer as server: waiting for a connection");
         /*below method accepts the request*/
```

```
            Socket socket = serverSocket.accept();
            // Here the connection is established
            InetAddress Machine = socket.getInetAddress();
            System.out.println("Producer as server has now accepted a connection" +
                " from " + Machine);
            // writing it to the outputstream.
            OutputStream outStream = socket.getOutputStream();
            // We are Creating The Producer instance
            Producer prod = new Producer(outStream);
            Thread prodt = new Thread(prod);
            // initialising the threads.
            prodt.start();
        } catch (IOException e) {
            System.out.println(e);
        } /*exception handling*/
    }

}
```

2) Producer.java:

```
import java.io.IOException;
import java.io.OutputStream;
```

```
public class Producer implements Runnable {
    /*we will now use outputstream here*/
    OutputStream optStream;

    public Producer(OutputStream optStream) {
        /*self referencing with the help of this statement*/
        this.optStream = optStream;
    }

    // Must implement the run() method. As a feature of runnable class

    public void run() {
        int cnt = 1;
        while (cnt <= 100) {
            /*Typecasting to bytes*/
            byte b = (byte) cnt;
            // Display The OutPut In System Console
            System.out.println("Producer is now writing the integer" + b);
            // count=++count;

            try {
                // here integer value stored into outPutStream.
                optStream.write(b);
            } catch (IOException e) {
```

```
            System.out.println(e);
        }

        // Sleep for a while.
        try {
            Thread.sleep(200);
        } catch (InterruptedException e) {
            System.out.println(e);
        }
    }

    // Write EOF and close output stream.
    try {
        optStream.write(-1);
        optStream.close();
    } catch (IOException e) {
        System.out.println(e);
    }

  }

}
```

Result for the ServerSockets

Producer as server: waiting for a connection

Producer as server has now accepted a connection from /127.0.0.1

Producer is now writing the integer 1

Producer is now writing the integer 2

Producer is now writing the integer 3

Producer is now writing the integer 4

Producer is now writing the integer 5

Producer is now writing the integer 6

Producer is now writing the integer 7

Producer is now writing the integer 8

Producer is now writing the integer 9

Producer is now writing the integer 10

Producer is now writing the integer 11

Producer is now writing the integer 12

Producer is now writing the integer 13

Producer is now writing the integer 14

Producer is now writing the integer 15

Producer is now writing the integer 16

Producer is now writing the integer 17

Producer is now writing the integer 18

Producer is now writing the integer 19

Producer is now writing the integer 20

Producer is now writing the integer 21

Producer is now writing the integer 22

Producer is now writing the integer 23

Producer is now writing the integer 24

Producer is now writing the integer 25

Producer is now writing the integer 26

Producer is now writing the integer 27

Producer is now writing the integer 28

Producer is now writing the integer 29

Producer is now writing the integer 30

Producer is now writing the integer 31

Producer is now writing the integer 32

Producer is now writing the integer 33

Producer is now writing the integer 34

Producer is now writing the integer 35

Producer is now writing the integer 36

Producer is now writing the integer 37

Producer is now writing the integer 38

Producer is now writing the integer 39

Producer is now writing the integer 40

Producer is now writing the integer 41

Producer is now writing the integer 42

Producer is now writing the integer 43

Producer is now writing the integer 44

Producer is now writing the integer 45

Producer is now writing the integer 46

Producer is now writing the integer 47

Producer is now writing the integer 48

Producer is now writing the integer 49

Producer is now writing the integer 50

Producer is now writing the integer 51

Producer is now writing the integer 52

Producer is now writing the integer 53

Producer is now writing the integer 54

Producer is now writing the integer 55

Producer is now writing the integer 56

Producer is now writing the integer 57

Producer is now writing the integer 58

Producer is now writing the integer 59

Producer is now writing the integer 60

Producer is now writing the integer 61

Producer is now writing the integer 62

Producer is now writing the integer 63

Producer is now writing the integer 64

Producer is now writing the integer 65

Producer is now writing the integer 66

Producer is now writing the integer 67

Producer is now writing the integer 68

Producer is now writing the integer 69

Producer is now writing the integer 70

Producer is now writing the integer 71

Producer is now writing the integer 72

Producer is now writing the integer 73

Producer is now writing the integer 74

Producer is now writing the integer 75

Producer is now writing the integer 76

Producer is now writing the integer 77

Producer is now writing the integer 78

Producer is now writing the integer 79

Producer is now writing the integer 80

Producer is now writing the integer 81

Producer is now writing the integer 82

Producer is now writing the integer 83

Producer is now writing the integer 84

Producer is now writing the integer 85

Producer is now writing the integer 86

Producer is now writing the integer 87

Producer is now writing the integer 88

Producer is now writing the integer 89

Producer is now writing the integer 90

Producer is now writing the integer 91

Producer is now writing the integer 92

Producer is now writing the integer 93

Producer is now writing the integer 94

Producer is now writing the integer 95

Producer is now writing the integer 96

Producer is now writing the integer 97

Producer is now writing the integer 98

Producer is now writing the integer 99

Producer is now writing the integer 100

Source code for ClientSocket

1) MainClientSocketsFile.java:

```
import java.io.IOException;
import java.io.InputStream;
import java.net.InetAddress;
import java.net.Socket;
public class MainClientSocketsFile {
   public static void main(String[] argv) {
      try {
         /* Here We are Creating The Socket Instance with 8000 port number*/
         Socket socket = new Socket("localhost", 8000);
         InetAddress Machine = socket.getInetAddress();
         System.out.println("Consumer as client: attempting connection" + " to the " +
            Machine);
         /* We are Creating The Instance For InputStream*/
         InputStream inptStream = socket.getInputStream();
         /*Here We Are Creating The ConsumerTestApp instance.*/
         Consumer consumerobj = new Consumer(inptStream);
         /* Here We Are Creating The Thread Instance.*/
         Thread consthread = new Thread(consumerobj);
         /*Here We are Start the thread.*/
         consthread.start();
      } catch (IOException e) {
```

```
            System.out.println(e);
        }
    }
}
```

2) Consumer.java :

```
import java.io.IOException;
import java.io.InputStream;

class Consumer implements Runnable {
            /*we will reference input stream*/
              InputStream inptStream;

  public Consumer(InputStream inptStream)
  {
    this.inptStream = inptStream;
  }
  public void run ()
  {
        while (true) {
      /* Get the next integer*/
      int j = -1;
      try {
            j = inptStream.read();
      }
```

```
    catch (IOException e) { System.out.println (e); }

    /*read the data.*/
    byte b = (byte) j;

    /*See if its end of data*/
    if ( (j < 0) || (b < 0) ) break;

    System.out.println ("Consumer is now reading the integer "+b);

    /* Sleep for a while.*/
    try {
        Thread.sleep (200);
    }
    catch (InterruptedException e) { System.out.println(e); }
  }

  try {
        inptStream.close();
  }
  catch (IOException e) { System.out.println (e); }

}
```

}

Result for the ClientSockets

Consumer as client: attempting connection to the localhost/127.0.0.1

Consumer is now reading the integer 1

Consumer is now reading the integer 2

Consumer is now reading the integer 3

Consumer is now reading the integer 4

Consumer is now reading the integer 5

Consumer is now reading the integer 6

Consumer is now reading the integer 7

Consumer is now reading the integer 8

Consumer is now reading the integer 9

Consumer is now reading the integer 10

Consumer is now reading the integer 11

Consumer is now reading the integer 12

Consumer is now reading the integer 13

Consumer is now reading the integer 14

Consumer is now reading the integer 15

Consumer is now reading the integer 16

Consumer is now reading the integer 17

Consumer is now reading the integer 18

Consumer is now reading the integer 19

Consumer is now reading the integer 20

Consumer is now reading the integer 21

Consumer is now reading the integer 22

Consumer is now reading the integer 23

Consumer is now reading the integer 24

Consumer is now reading the integer 25

Consumer is now reading the integer 26

Consumer is now reading the integer 27

Consumer is now reading the integer 28

Consumer is now reading the integer 29

Consumer is now reading the integer 30

Consumer is now reading the integer 31

Consumer is now reading the integer 32

Consumer is now reading the integer 33

Consumer is now reading the integer 34

Consumer is now reading the integer 35

Consumer is now reading the integer 36

Consumer is now reading the integer 37

Consumer is now reading the integer 38

Consumer is now reading the integer 39

Consumer is now reading the integer 40

Consumer is now reading the integer 41

Consumer is now reading the integer 42

Consumer is now reading the integer 43

Consumer is now reading the integer 44

Consumer is now reading the integer 45

Consumer is now reading the integer 46

Consumer is now reading the integer 47

Consumer is now reading the integer 48

Consumer is now reading the integer 49

Consumer is now reading the integer 50

Consumer is now reading the integer 51

Consumer is now reading the integer 52

Consumer is now reading the integer 53

Consumer is now reading the integer 54

Consumer is now reading the integer 55

Consumer is now reading the integer 56

Consumer is now reading the integer 57

Consumer is now reading the integer 58

Consumer is now reading the integer 59

Consumer is now reading the integer 60

Consumer is now reading the integer 61

Consumer is now reading the integer 62

Consumer is now reading the integer 63

Consumer is now reading the integer 64

Consumer is now reading the integer 65

Consumer is now reading the integer 66

Consumer is now reading the integer 67

Consumer is now reading the integer 68

Consumer is now reading the integer 69

Consumer is now reading the integer 70

Consumer is now reading the integer 71

Consumer is now reading the integer 72

Consumer is now reading the integer 73

Consumer is now reading the integer 74

Consumer is now reading the integer 75

Consumer is now reading the integer 76

Consumer is now reading the integer 77

Consumer is now reading the integer 78

Consumer is now reading the integer 79

Consumer is now reading the integer 80

Consumer is now reading the integer 81

Consumer is now reading the integer 82

Consumer is now reading the integer 83

Consumer is now reading the integer 84

Consumer is now reading the integer 85

Consumer is now reading the integer 86

Consumer is now reading the integer 87

Consumer is now reading the integer 88

Consumer is now reading the integer 89

Consumer is now reading the integer 90

Consumer is now reading the integer 91

Consumer is now reading the integer 92

Consumer is now reading the integer 93

Consumer is now reading the integer 94

Consumer is now reading the integer 95

Consumer is now reading the integer 96

Consumer is now reading the integer 97

Consumer is now reading the integer 98

Consumer is now reading the integer 99

Consumer is now reading the integer 100

DISCUSSION

Having done this project there are a few things I have learned in the course of completion of this project. They are as follows:

1) Java is a very useful language and with a lot of inbuilt libraries.
2) When BlueJ and Eclipse are considered as IDEs, I found BlueJ to be good for analysis part. Since BlueJ helps us with the UML diagrams. UML diagrams are useful to architect and think about the solution. It is like the blueprint schematics of the whole program. Eclipse on the other hand has a very powerful debugger that helps us spot the bug very easily.
3) Apart from the textbooks that were mentioned in the project description, I found the Java Documentation available online to be really helpful in understanding the internals of few classes and their methods.
4) The client and socket program can be implemented using different systems in which one acts as client and other acts as server.
5) Data encryption will ensure the security of the data during its transmission in the pipe from producer to consumer.

CONCLUSION

Interprocess communication between a producer and a consumer is implemented by 3 methods. The complete project was designed and implemented using leading edge software and the whole process was well documented. This project can be used as a reference for solving sections of bigger problems that arise in future in this field.

APPENDIX A

[A1]

The following are the links of the software that I have used for the implementation of this project, to help you get started with the programming of it.

1) Java SE Development Kit 8 - http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html

2) Eclipse IDE for Java Developers – http://www.eclipse.org/downloads/packages/eclipse-ide-java-developers/oxygen3a

3) BlueJ- For UML class designs – https://www.bluej.org/
4) draw.io – for flow charts design - https://www.draw.io/

The rest of the softwares that have been used by me in the completion of this project are not open sourced, and one needs to buy them from the vendors themselves.