Cycle-1-Shell Scripting

0:UNIX commands   Date : 27/01/2020

Problem 0: UNIX COMMANDS

AIM:

Try out the following unix commands(use manual and help features for support).

1. echo, read
2. more, less
3. man
4. chmod, chown
5. cd, mkdir, pwd, ls, find
6. cat, mv, cp, rm
7. wc, cut, paste
8. head, tail, grep, expr
9. Redirections & Piping
10. useradd, usermod, userdel, passwd
11. tar

PROGRAM CODE:

echo read	Echo displays line of text or strings that are passes as arguments. Syntax : Echo [string] Read is used to get input from keyboard. Syntax : Read text
more less	More is used to view the text file in command prompt displaying one screen at a time in case file is large. Syntax : more -d sample.txt Less is used to read contents of text file in page per time. Syntax : sudo | less
man	man displays man pages for commands manmeors manual for command. Syntax : man chown
Chmod chown	Chmod set the permission flag . Syntax : chmod 766 ex.txt Chown allow change owner and group owner of file. Syntax : sudo chown clave : mary example.txt

cd mkdir pwd ls find	Cd changes current directory. Syntax : cd directory_name Mkdir creates new directory . Syntax : mkdir invoice Pwd prints workind directory from root directory. Syntax : pwd Ls lists file and folder in directory . Syntax : ls –l Find tracks down file. Syntax : find . name * ones
cat mv cp rm	Cat concatenates file and print to stdout. Syntax : cat [OPTION]..[FILE] Mv moves file or rename files. Syntax : mv[OPTION]source Cp copy files. Syntax : cp [OPTION] source Rm removes files and directories. Syntax:rm [OPTION]…[FILE]
wc cut paste	Wc is used for printing newline , word and byte counts for files. Cut is used for cutting out sections for each line of files and writes result to standard output. Paste is used to join files horizontally by outputting lines.
head and tail grep expr	Head command prints lines from beginning of a file and tail prints lines from end of file. Grep searches through a set of files for arbitrary text pattern through regular expression. Expr evaluates a given expression and displays corresponding output.
Redirection	Redirection is a feature when executes a command, we can change input or output devices.

piping	Piping is a form of redirection ie used in linux to send output of one command for further processing.
Useradd Usermod userdel Passwd	Useradd is used to create new accounts in linux. Usermod used to modify existing accounts in linux. Userdel is used to delete account in linux. Passwd is used to assign password to local accounts of users.
tar	Tar stands for tap to achieve which is used to tape drive back up command used by linux. Syntax : tar [OPTIONS] [ARCHIEVE-FILE] [FILE OD DIRECTORY TO BE ACHIEVED]

1:welcome message Date : 02/02/2020

Problem 1 : WELCOME MESSAGE

AIM:

Print a customized welcome message. Get the name of the user as input and attach the name to the welcome message. Eg. "Welcome Rahul".

PROGRAM CODE:

welcome	read -p "Enter your name " name echo Welcome $name

RESULT:

```
Enter your name Revathy
Welcome Revathy
```

2: Greatest of 2 numbers Date : 06/02/2020

Problem 2 : GREATEST OF 2 NUMBERS

AIM:

Take 2 numbers as input and print the greater of the two.

PROGRAM CODE:

greater	read -p “Enter first number “ num1 read –p “Enter second number “ num2 if [$num1 –gt $num2] then echo “ Largest number is :” $num1 else echo “Largest number is :” $num2 fi

RESULT:

```
Enter first number 45
Enter second number 67
Largest number is : 67
```

3: Odd numbers Date : 09/01/2020

Problem 3: ODD NUMBERS

AIM:

Print the first 20 odd numbers.

PROGRAM CODE:

odd	i=1
	count=0
	echo “Odd numbers are :”
	while [$i –lt 100]
	do
	echo $i
	count=` expr  $count + 1 `
	if [$count –eq 20]
	then
	break
	fi
	i=` expr $i + 2 ’
	done

RESULT:

```
Odd numbers are :
1
3
….
39
```

4: Sum of 20 numbers Date : 16/02/2020

Problem 4 : SUM OF 20 NUMBERS

AIM:

Store 20 numbers in an array and print their sum.

PROGRAM CODE:

arraysum	read –p “Enter 20 numbers” input
	sum=0
	for i in ${input[@]}
	do
	sum=` expr  $sum + $i `
	done
	echo “The sum is :” $sum

RESULT:

```
Enter 20 numbers 1 2 3 4 5 6 1 2 3 4 1 2 3 4 5 6 7 8 9 1
The sum is  : 77
```

5: Creating a text file Date : 16/02/2020

Problem 5 : CREATING A TEXT FILE

AIM:

Create a text file with 20 lines of text.

PROGRAM CODE:

text	cat > que5.txt Welcome to Linux. Linux is the best-known and most-used open source operating system. As an operating system, Linux is software that sits underneath all of the other software on a computer, receiving requests from those programs and relaying these requests to the computer's hardware. Cntrl + D

RESULT:

File created que5.txt.

6: Replacing strings Date : 20/02/2020

Problem 6: REPLACING STRING

AIM:

Open the file created in question 5 and replace any string with another without using stream editor.

PROGRAM CODE:

replace	While read a; do
	echo ${a//linux/shell scripting}
	done<que5.txt>que5.txt.t
	mv que5.txt{.t,}

RESULT:

Welcome to shell scripting. Linux is the best-known and most-used open source operating system. As an operating system, Linux is software that sits underneath all of the other software on a computer, receiving requests from those programs and relaying these requests to the computer's hardware.

7:Protocols and description Date : 23/02/2020

PROGRAM 7 : PROTOCOLS AND DESCRIPTION

AIM:

Open the /etc/protocols file and copy the protocol number of the following protocols into another file named "favorite protocols" and format it in the same way as the original /etc/protocol file.

1. udp
2. idrp
3. skip
4. ipip

PROGRAM CODE:

protocol	grep "udp\\|idrp\\|skip\\|ipip" /etc/protocols> favoriteprotocols.txt cat favoriteprotocols.txt

RESULT:

```
idrp    17      UDP    #User Datagram Protocol
```

idrp	45	IDRP	#Inter- Domain Routing Protocol
skip	57	SKIP	#SKIP
ipip	94	IPIP	#IP- within- IP Encapsulation Protocol

8: Using AT and BATCH Date : 27/02/2020

Problem 8: USING AT AND BATCH

AIM:

Use "at" and "batch" to schedule tasks.

PROGRAM CODE:

at	echo "sh execute.sh" |at now+1 minute
execute.sh	echo " Hello world" >create.txt
Batch (In Terminal)	>batch Warning: commands will be executed using /bin/sh at> echo " welcome.... >out.txt at>cntrl + D

RESULT:

job 1 at Mon Feb 27 21:55:00 2020

The file named execute.sh will be executed and create.txt will be created after one minute.dd

job 2 at Mon Feb 27 22:05:00 2020

The file named out.txt will be created .

9:CRON command   Date : 02/03/2020

Problem 9: CRON COMMAND

AIM:

Use cron to schedule tasks.

PROGRAM CODE:

In terminal	Crontab –e
In nano editor	49 * * * * cd/home/mca50/Downloads/sd && sh execute.sh
execute.sh	echo “Hello world “>create.txt

RESULT:

When time (minutes) becomes 49 execute.sh will be execute. In execute.sh , we wants to create a file named as create.txt with text “ Hello world “.

11: UNIX Mail   Date : 05/03/2020

Problem 10: UNIX MAIL

AIM:

Set up unix mail and use mail to send and receive mails to and from users using shell scripting.

PROGRAM CODE:

ques10.sh	echo “hi revathy, welcome “|mail –s “This is the subject” revathychandran47@gmail.com

RESULT:

In gmail
19mca50
This is the subject
Hi revathy, welcome

Cycle 2- Version Control using git

1:Git repository   Date : 09/03/2020

PROGRAM 1 : GIT REPOSITORY

AIM:

0. Install and initialize git and perform the following operations

a. Create a text file in your git directory.

b. Configure your git with your credentials.

c. Configure the default editor to your favorite editor

d. Stage your files

e. Create your first commit

f. Push to remote repository

The following exercises must be done by a team of four students.

1. Create team account.

2. Create empty repository in any git remote repository service and
add collaborators.

3. Leader must create the first commit.

4. All members must clone the remote repository.

5. Each member must create a feature branch each and add features to them(any mod)

6. Commit changes to branches.

7. Push the branches.

8. View Graph.

9. Leader must make changes to the master.

10. All member must rebase their branches to the position of latest commit in master.

11.Merge all branches to master.

12.Cherry pick commits from each branch created earlier.

13.View Status.

14.View History.

15.Delete all branches.

PROGRAM CODE :

Setting the git configurations in terminal.

- git config --global user.name

  Revathychandran47

- git config --global user.email

  revathychandran47@gmail.com

- git config --global core.edit nano

Adding a file in your git initialized folder to staging area ,Use the

command :

- git add filename.txt

We can use following command to see both tracked and untracked

files.

- git status

To commit the added file to local repository, Use the

command :

- git commit -m "a meaninful comment".

To clone the entire github repository , use command :

- git clone https://github.com/Anjali-941/first-

  repo.git.

To push the committed file to our git repository , use the below command:

- git pull https://github.com/Anjali-941/first-repo.git
- git push https://github.com/Anjali-941/first-repo.git master

To create a new branch and branch operations. Use the following commands

- git checkout -b revs   /*to create a branch named revs*/
- git branch   /*to show all branches*/
- git pull https://github.com/Anjali-941/first-repo.git
- git push https://github.com/Anjali-941/first-repo.git

  revs

To merge the branch to our master branch ,use following

commands.

- git checkout master   /*switching to master branch*/
- git merge revs   /*merging the branch repos*/

To cherry pick a commit done in revs branch to master branch, use the

command :

- git cherry-pick commitid

To view the operations in a graph format , use the

command:

- git log –-graph

To rebase the branch to master branch , use the

command :

- git checkout revs
- git rebase master   / * rebasing branch to master

branch*/

To view the history of git commits, use the command :

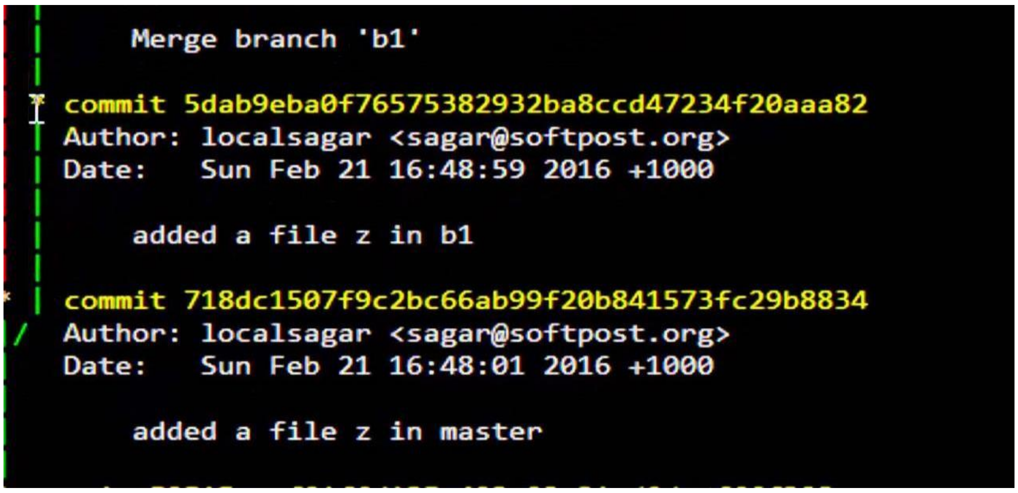
- git log --oneline

To delete a feature branch , use the command :

- git branch -d revs /*deletes repos branch*/

RESULT :

Output of git log –graph command



```
| |     Merge branch 'b1'
| |
| * commit 5dab9eba0f76575382932ba8ccd47234f20aaa82
| | Author: localsagar <sagar@softpost.org>
| | Date:   Sun Feb 21 16:48:59 2016 +1000
| |
| |     added a file z in b1
| |
* | commit 718dc1507f9c2bc66ab99f20b841573fc29b8834
|/  Author: localsagar <sagar@softpost.org>
|   Date:   Sun Feb 21 16:48:01 2016 +1000
|
|       added a file z in master
|
```

Figure 1 :Git graph

Cycle-3-Network Programming In Java

1:TCP client-server   Date : 12/3/2020

Problem 1:TCP CLIENT - SERVER

AIM:

Implement Bidirectional Client-Server communication using TCP.

PROGRAM CODE:

Server2.java

```
import java.io.*;
import java.net.*;
class Server2 {

      public static void main(String args[])
            throws Exception
      {
    ServerSocket ss = new ServerSocket(888);
    Socket s = ss.accept();
    System.out.println("Connection
    established");

    PrintStream ps = new
    PrintStream(s.getOutputStream());

            BufferedReader br = new
            BufferedReader(new InputStreamReader
        (s.getInputStream()));

            BufferedReader kb = new
            BufferedReader(new InputStreamReader
        (System.in));
```

	```
	        while (true) {
	          String str, str1;
	          while ((str = br.readLine()) != null)
	          {
	                                    System.out.println(str);
	                                    str1 = kb.readLine();

	                              ps.println(str1);
	                                }

	          ps.close();
	                                br.close();
	                                kb.close();
	                                ss.close();
	                                s.close();
	           System.exit(0);

	                    }
	          }
	}
	```

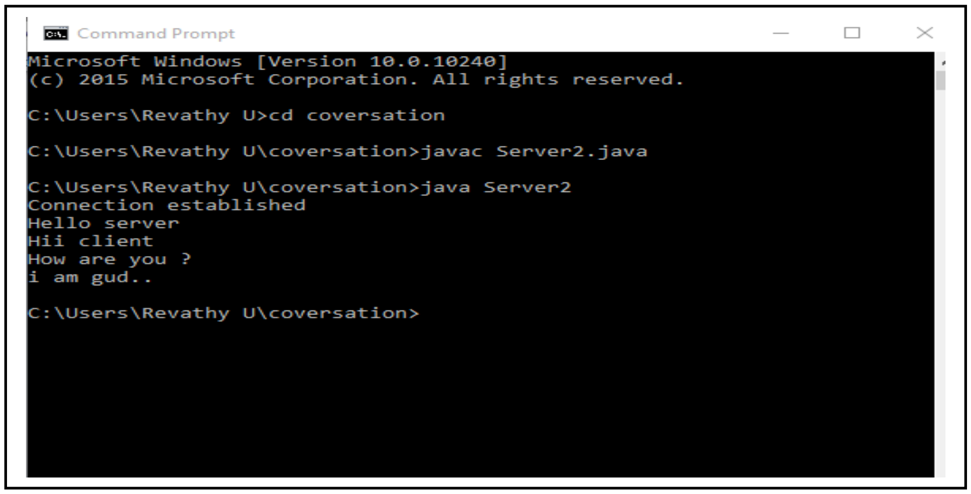
Client2.java	
	```
	import java.io.*;
	import java.net.*;

	class Client2 {

	            public static void main(String args[])
	                        throws Exception
	            {

	                Socket s = new Socket("localhost", 888);
	        DataOutputStream dos = new
	                DataOutputStream(s.getOutputStream());

	         BufferedReader br = new
	                        BufferedReader(new InputStreamReader(
	         s.getInputStream()));
	```

	```
	BufferedReader kb = new
	BufferedReader(new InputStreamReader
	(System.in));

	String str, str1;
	while (!(str = kb.readLine()).
	equals("exit")) {

	    dos.writeBytes(str + "\n");

	    str1 = br.readLine();

	    System.out.println(str1);
	}


	dos.close();
	br.close();
	kb.close();
	s.close();
	}
	}
	```

RESULT:

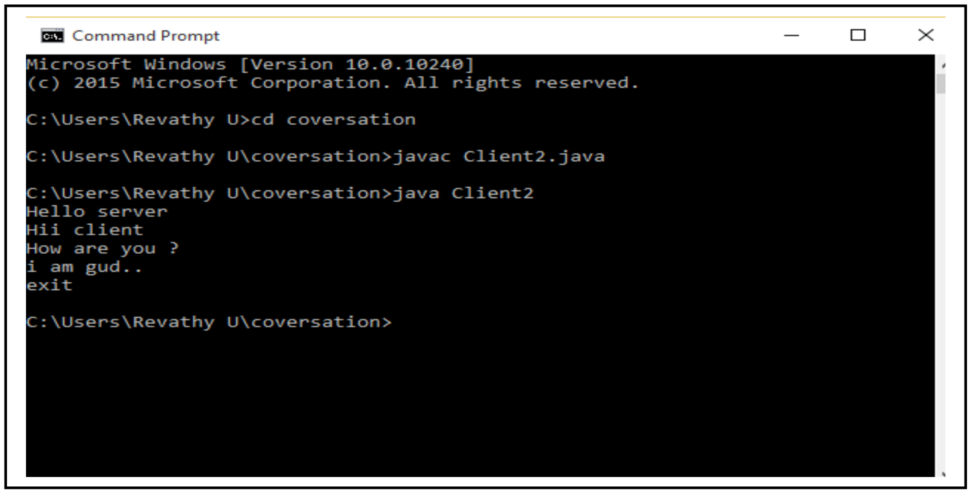
We are creating a local client and server communication. First running server program , if it is free of bugs it will wait for a client to connect. Then we run client program , if it is correct then a connection between client and server will be established.

SCREENSHOTS:



```
Microsoft Windows [Version 10.0.10240]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Users\Revathy U>cd coversation

C:\Users\Revathy U\coversation>javac Server2.java

C:\Users\Revathy U\coversation>java Server2
Connection established
Hello server
Hii client
How are you ?
i am gud..

C:\Users\Revathy U\coversation>
```

Figure 1: This is the local server which communicates with client machine after the establishment of connection between client and server.



```
Microsoft Windows [Version 10.0.10240]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Users\Revathy U>cd coversation

C:\Users\Revathy U\coversation>javac Client2.java

C:\Users\Revathy U\coversation>java Client2
Hello server
Hii client
How are you ?
i am gud..
exit

C:\Users\Revathy U\coversation>
```

Figure 2: This is the local client which communicates with server machine after the establishment of connection between client and server.

2:TCP echo server Date : 16/3/2020

Problem 2:TCP ECHO SERVER

AIM:

Implement Echo Server using TCP.

PROGRAM CODE:

EchoServer.java	import java.io.*;
	import java.net.*;
	public class EchoServer
	{
	public static void main(String args[]) throws Exception

	```
	{
	  try
	  {
	    int Port;
	    BufferedReader Buf =new BufferedReader(
	    new InputStreamReader(System.in));

	    System.out.print(" Enter the Port
	    Address : " );

	    Port=Integer.parseInt(Buf.readLine());
	    ServerSocket sok =new ServerSocket
	    (Port);

	    System.out.println(" Server is Ready To
	    Receive a Message. ");
	    System.out.println(" Waiting ..... ");
	    Socket so=sok.accept();

	   if(so.isConnected()==true)
	    System.out.println(" Client Socket is
	    Connected Succecfully. ");

	    InputStream in=so.getInputStream();
	    OutputStream ou=so.getOutputStream();
	    PrintWriter pr=new PrintWriter(ou);
	    BufferedReader buf=new BufferedReader

	   (new InputStreamReader(in));

	   String str=buf.readLine();
	   System.out.println(" Message Received
	   From Client : " + str);
	```

	```
	System.out.println(" This Message is
	Forwarded To Client. ");

	pr.println(str);
	pr.flush();
	}
	catch(Exception e)
	{
	System.out.println(" Error : " +
	e.getMessage());
	}
	}
	}
	```

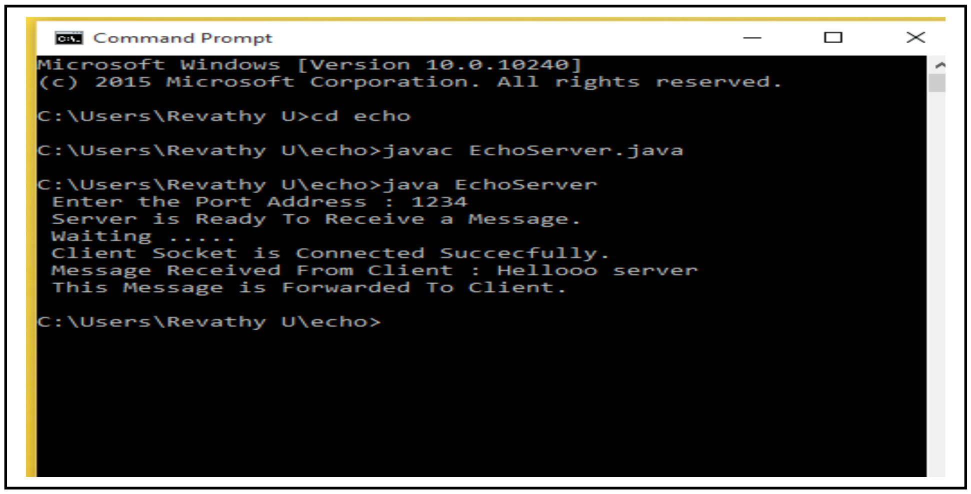
EchoClient.java	
	```
	import java.io.*;
	import java.net.*;

	public class EchoClient
	  {
	    public static void main(String args[]) throws
	    Exception
	    {
	      try
	      {

	        int Port;
	        BufferedReader Buf =new BufferedReader
	        (new InputStreamReader(System.in));

	        System.out.print(" Enter the Port
	        Address : " );

	        Port=Integer.parseInt(Buf.readLine());
	        Socket sok=new Socket("localhost",
	        Port);



	        if(sok.isConnected()==true)
	        System.out.println(" Server Socket is
	        Connected Succecfully. ");
	```

	```
	        InputStream in=sok.getInputStream();
	        OutputStream ou=sok.getOutputStream();

	        PrintWriter pr=new PrintWriter(ou);

	        BufferedReader buf1=new BufferedReader
	        (new InputStreamReader (System.in));
	        BufferedReader buf2= new
	        BufferedReader(new InputStreamReader
	        (in));

	        String str1,str2;
	        System.out.println(" Enter the Message :
	        ");

	        str1=buf1.readLine();

	        pr.println(str1);
	        pr.flush();

	        System.out.println(" Message Send
	        Successfully. ");

	        str2=buf2.readLine();

	        System.out.println(" Message From
	        Server : " + str2);
	      }
	      catch(Exception e)
	      {
	        System.out.println(" Error : " +
	        e.getMessage());
	      }
	    }
	```

RESULT:

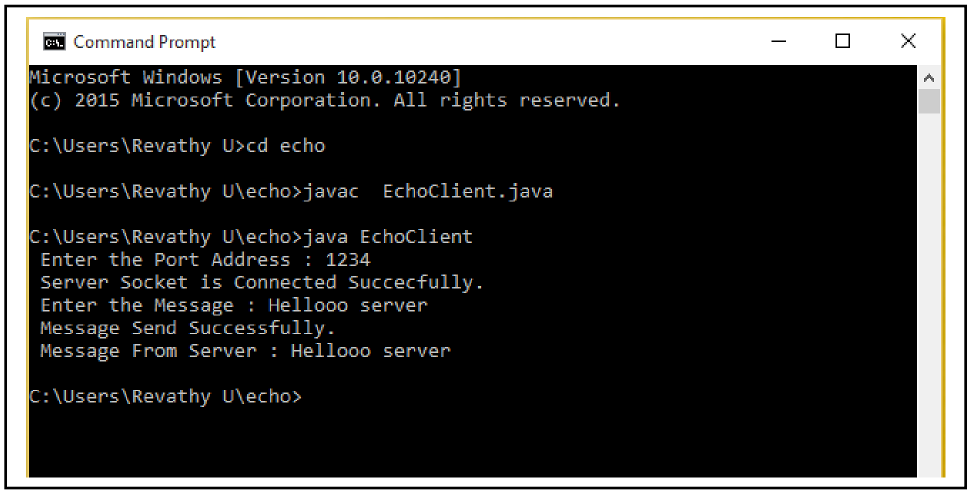
We are creating a local client and echo server communication. First running echo server program , if it is free of bugs it will wait for a client to connect. Then we run client program , if it is correct then a connection between client and echo server will be established. In our program , the message received from client is forwarded to client itself by the echo server.

SCREENSHOTS:



```
Microsoft Windows [Version 10.0.10240]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Users\Revathy U>cd echo

C:\Users\Revathy U\echo>javac EchoServer.java

C:\Users\Revathy U\echo>java EchoServer
 Enter the Port Address : 1234
 Server is Ready To Receive a Message.
 Waiting .....
 Client Socket is Connected Succecfully.
 Message Received From Client : Hellooo server
 This Message is Forwarded To Client.

C:\Users\Revathy U\echo>
```

Figure 1: This is the echo server which communicates with Client machine after the establishment of connection between client and server. The message received is forwarded to client.



```
Microsoft Windows [Version 10.0.10240]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Users\Revathy U>cd echo

C:\Users\Revathy U\echo>javac  EchoClient.java

C:\Users\Revathy U\echo>java EchoClient
 Enter the Port Address : 1234
 Server Socket is Connected Succecfully.
 Enter the Message : Hellooo server
 Message Send Successfully.
 Message From Server : Hellooo server

C:\Users\Revathy U\echo>
```

Figure 2: This is the client which communicates with echo server after the establishment of connection between client and server.

Problem 3: CHAT SERVER USING UDP

AIM:

Implement Chat Server using UDP.

PROGRAM CODE:

UDPServer.java

```
import java.io.*;
import java.net.*;
class UDPServer
{
  public static DatagramSocket serversocket;
  public static DatagramPacket dp;
  public static BufferedReader dis;
  public static InetAddress ia;
  public static byte buf[] = new byte[1024];
  public static int cport = 789,sport=790;
  public static void main(String[] a) throws
  IOException
  {
    serversocket = new DatagramSocket(sport);
    dp = new DatagramPacket(buf,buf.length);
    dis = new BufferedReader
    (new InputStreamReader(System.in));
    ia = InetAddress.getLocalHost();
    System.out.println("Server is Running...");
    while(true)
    {
      serversocket.receive(dp);
      String str = new String(dp.getData(), 0,
      dp.getLength());
      if(str.equals("STOP"))
```

	```
	        {
	          System.out.println("Terminated...");
	          break;
	        }
	        System.out.println("Client: " + str);
	        String str1 = new String(dis.readLine());
	        buf = str1.getBytes();
	       serversocket.send(new
	       DatagramPacket(buf,str1.length(), ia,
	       cport));
	     }
	   }
	 }
	```

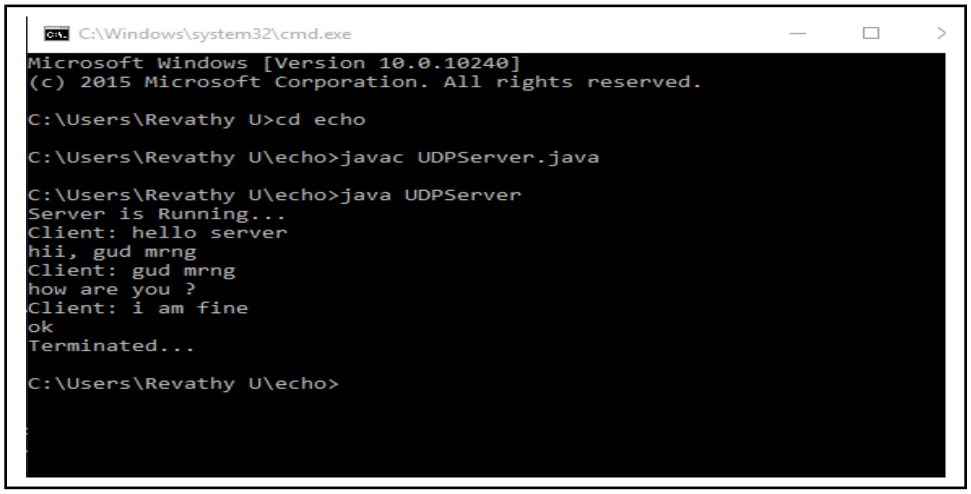
UDPClient.java	
	```
	import java.io.*;
	import java.net.*;

	class UDPClient
	{
	  public static DatagramSocket clientsocket;
	  public static DatagramPacket dp;
	  public static BufferedReader dis;
	  public static InetAddress ia;
	  public static byte buf[] = new byte[1024];
	  public static int cport = 789, sport = 790;
	  public static void main(String[] a) throws
	  IOException
	  {
	    clientsocket = new DatagramSocket(cport);
	    dp = new DatagramPacket(buf, buf.length);

	    dis = new BufferedReader(new
	    InputStreamReader(System.in));
	    ia = InetAddress.getLocalHost();

	    System.out.println("Client is Running... Type
	    'STOP'to Quit");

	    while(true)

	{
	```

```
        String str = new String(dis.readLine());
        buf = str.getBytes();
        if(str.equals("STOP"))
        {
          System.out.println("Terminated...");



          clientsocket.send(new
          DatagramPacket(buf,str.length(), ia,
          sport));
          break;
        }

        clientsocket.send(new DatagramPacket(buf,
        str.length(), ia, sport));

        clientsocket.receive(dp);

        String str2 = new String(dp.getData(), 0,
        dp.getLength());

        System.out.println("Server: " + str2);
      }

    }

  }
```

RESULT:

We are creating a local client and echo server communication using UDP. First running server program , if it is free of bugs it will wait for a client to connect. Then we run client program , if it is correct then a connection between client and server will be established. Thus after the successful establishment of connection between them , they can send and receive messages as in a chat.

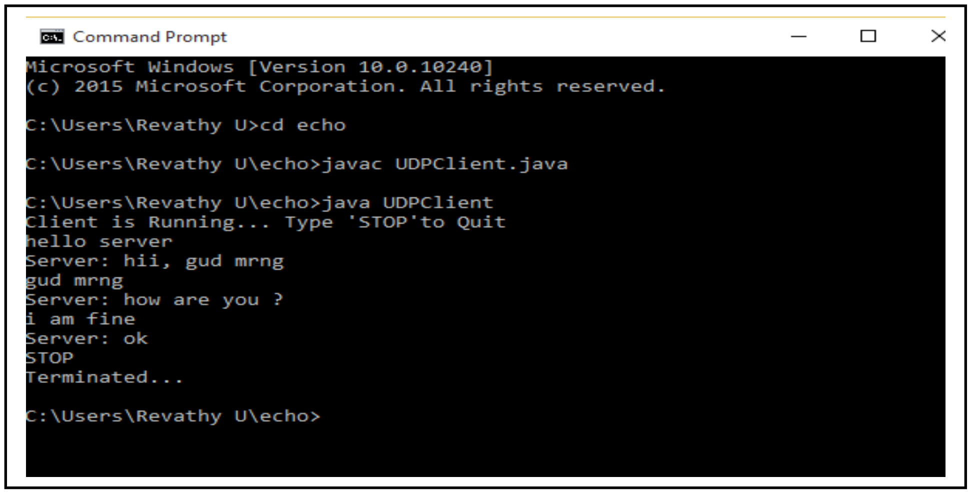
SCREENSHOTS:



C:\Windows\system32\cmd.exe

```
Microsoft Windows [Version 10.0.10240]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Users\Revathy U>cd echo

C:\Users\Revathy U\echo>javac UDPServer.java

C:\Users\Revathy U\echo>java UDPServer
Server is Running...
Client: hello server
hii, gud mrng
Client: gud mrng
how are you ?
Client: i am fine
ok
Terminated...

C:\Users\Revathy U\echo>
```

Figure 1:The figure shows the local server. Server will wait for the client to connect. After the client has connected , client can send a message to server. It will be shown as a chat .



Command Prompt

```
Microsoft Windows [Version 10.0.10240]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Users\Revathy U>cd echo

C:\Users\Revathy U\echo>javac UDPClient.java

C:\Users\Revathy U\echo>java UDPClient
Client is Running... Type 'STOP'to Quit
hello server
Server: hii, gud mrng
gud mrng
Server: how are you ?
i am fine
Server: ok
STOP
Terminated...

C:\Users\Revathy U\echo>
```

Figure 2: The figure shows the local client. After the client has connected to the server , client can send a message to server. It will be shown as a chat . The chat can be stopped if client can send a message to server as STOP.