WEEK 4

AIM:

To implement character stuffing and bit stuffing programs.

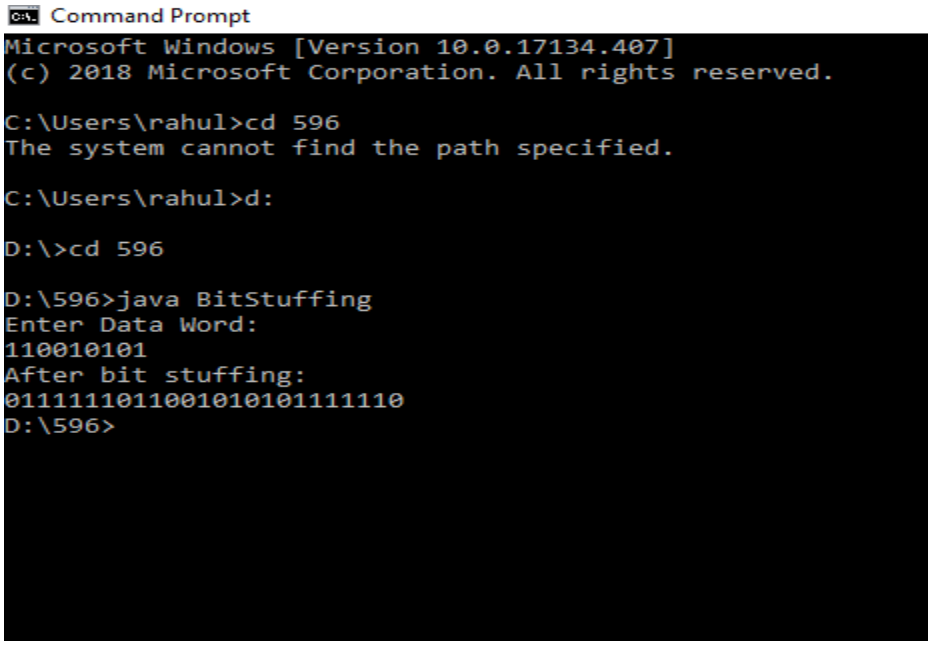
PROGRAM:

1.BIT STUFFING:

```
import java.util.*;
import java.io.*;
class BitStuffing{
    public static void main(String args[]){
        Scanner s=new Scanner(System.in);
        System.out.println("Enter Data Word:");
        String dw=s.next();
        char dwa[]=dw.toCharArray();
        char bdwa[]=new char[dwa.length*dwa.length];
        int i=0,j=0,k=0,count=1;
        bdwa[j++]='0';
        for(i=0;i<6;i++)
          bdwa[j++]='1';
        bdwa[j++]='0';
        i=0;
        while(i<dw.length()){
          if(dwa[i]=='1'){
              bdwa[j]=dwa[i];
              if(i!=dw.length()-1){
              for(k=i+1;dwa[k]=='1'&&k<dwa.length&&count<5;k++){
                 j++;
```

```
                bdwa[j]=dwa[k];
                count++;
                if(count==5){
                  j++;
                  bdwa[j]='0';
                }
               i=k;
              }
            }
          }
          else{
             bdwa[j]=dwa[i];
         }
          i++;j++;
      }
       bdwa[j++]='0';
       for(i=0;i<6;i++)
         bdwa[j++]='1';
       bdwa[j++]='0';
      System.out.println("After bit stuffing:");
      for(i=0;i<j;i++){
        System.out.print(bdwa[i]);
      }
   }
}
```

OUTPUT:



2. CHARACTER STUFFING:

```
import java.util.*;
import java.io.*;
class CharacterStuffing{
    public static void main(String args[]){
        Scanner sc=new Scanner(System.in);
        System.out.println("Enter the character string");
        String dw=sc.next();
        System.out.println("Enter the starting delimiter");
        String sd=sc.next();
        System.out.println("Enter the ending delimiter");
        String ed=sc.next();
        String x,y,d,s;
        char t[]=new char[2];
        String fs="";
```

```
        x=""+sd;
        s=""+sd+sd;
        y=""+ed;
        d=""+ed+ed;
        fs=fs+x;
        for(int i=0;i<dw.length();i++){
           t[0]=dw.charAt(i);
           if(t[0]==sd.charAt(0))
              fs=fs+s;
           else if(t[0]==ed.charAt(0)){
              fs=fs+d;
           }
           else
             fs=fs+""+t[0];
        }
        fs=fs+y;
        System.out.println("After Character Stuffing "+fs);
    }
}
```

OUTPUT:

```
D:\596>java CharacterStuffing
Enter the character string
hello
Enter the starting delimiter
e
Enter the ending delimiter
o
After Character Stuffing eheellooo
```

WEEK 5

AIM:

To implement protocols to transmit data through noisy and noise-less channels in data-link layer.

PROGRAM:

SIMPLEST PROTOCOL

serverSimplest.java

```
import java.util.*;
```

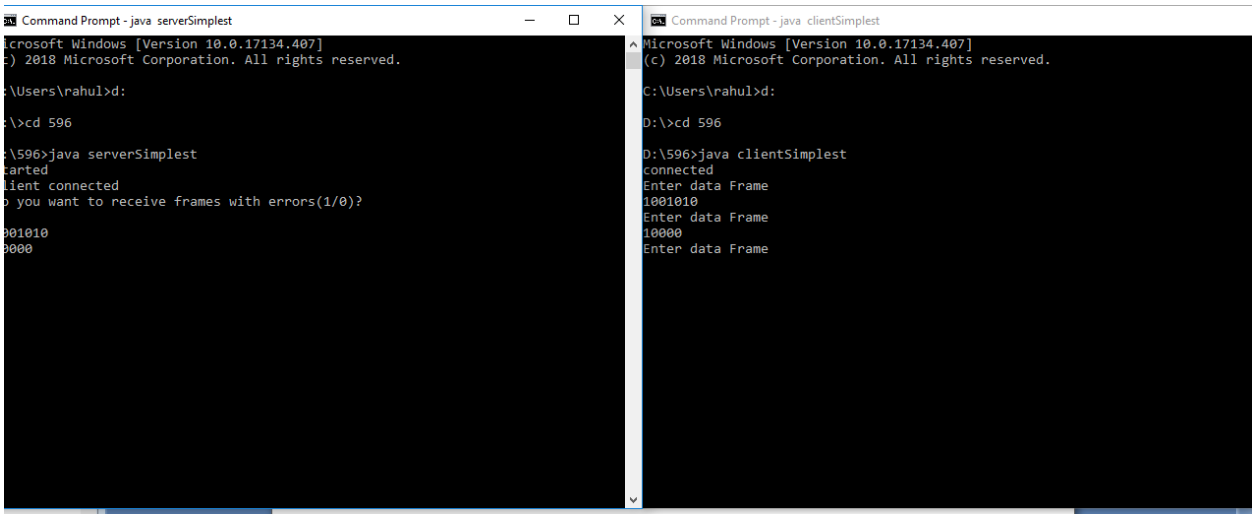
```
import java.io.*;
import java.net.*;
public class serverSimplest{
    public serverSimplest(int port)throws UnknownHostException,IOException{
     ServerSocket ss=new ServerSocket(port);
     System.out.println("started");
     Socket s=ss.accept();
     System.out.println("client connected");
     DataInputStream input=new DataInputStream(s.getInputStream());
     System.out.println("Do you want to receive frames with errors(1/0)?");
     Scanner sc=new Scanner(System.in);
     int d=sc.nextInt();
    if(d==0){
     while(true){
       String str=input.readUTF();
       System.out.println(str);
     }
    }
    else{
       while(true){
         String str1=input.readUTF();
         System.out.println(str1);
         input.readUTF();
        }
    }
   }
```

```
    public static void main(String args[])throws
UnknownHostException,IOException{
        serverSimplest sim=new serverSimplest(5000);
    }
}
```

clientSimplest.java

```
import java.io.*;
import java.util.*;
import java.net.*;
public class clientSimplest{
   public clientSimplest(String ip,int port)throws
UnknownHostException,IOException{
      Socket s=new Socket(ip,port);
      System.out.println("connected");
      DataOutputStream out=new DataOutputStream(s.getOutputStream());
      Scanner sc=new Scanner(System.in);
      while(true){
        System.out.println("Enter data Frame");
        String df=sc.nextLine();
        out.writeUTF(df);
      }
   }
   public static void main(String args[])throws UnknownHostException,IOException{
      clientSimplest cs=new clientSimplest("127.0.0.1",5000);
   }
}
```

OUTPUT



STOP AND WAIT PROTOCOL

serverStopWait.java

```
import java.util.*;
import java.io.*;
import java.net.*;
public class serverStopWait{
    public serverStopWait(int port)throws UnknownHostException,IOException{
     ServerSocket ss=new ServerSocket(port);
     System.out.println("started");
     Socket s=ss.accept();
     System.out.println("client connected");
     DataInputStream input=new DataInputStream(s.getInputStream());
     DataOutputStream out=new DataOutputStream(s.getOutputStream());
     System.out.println("Do you want to receive frames with errors(1/0)?");
     Scanner sc=new Scanner(System.in);
     int d=sc.nextInt();
```

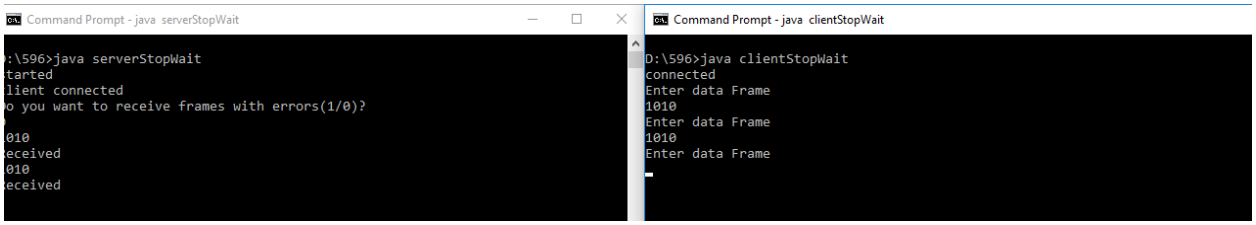
```
     String ack="Received";
     if(d==0){
      while(true){
       String str=input.readUTF();
       System.out.println(str);
       out.writeUTF(ack);
       System.out.println(ack);
     }
    }
    else{
       while(true){
         String str1=input.readUTF();
         System.out.println(str1);
         out.writeUTF(ack);
         System.out.println(ack);
         input.readUTF();
        }
    }
   }
    public static void main(String args[])throws
UnknownHostException,IOException{
        serverStopWait ssw=new serverStopWait(5000);
    }
}
```

clientStopWait.java

```
import java.io.*;
```

```
import java.util.*;
import java.net.*;
public class clientStopWait{
   public clientStopWait(String ip,int port)throws
UnknownHostException,IOException{
      Socket s=new Socket(ip,port);
      System.out.println("connected");
      DataOutputStream out=new DataOutputStream(s.getOutputStream());
      DataInputStream input=new DataInputStream(s.getInputStream());
      Scanner sc=new Scanner(System.in);
      while(true){
        System.out.println("Enter data Frame");
        String df=sc.nextLine();
        out.writeUTF(df);
        String ack=input.readUTF();
        if(!ack.equals("Received")){
           for(int i=0;i<10000000;i++);
        }
      }
   }
   public static void main(String args[])throws UnknownHostException,IOException{
      clientStopWait csw=new clientStopWait("127.0.0.1",5000);
   }
}
```

OUTPUT



STOP AND WAIT WITH ARQ PROTOCOL

serverSW.java

```
import java.util.*;
import java.io.*;
import java.net.*;
public class serverStopWaitARQ{
    public serverStopWaitARQ(int port)throws UnknownHostException,IOException{
     ServerSocket ss=new ServerSocket(port);
     System.out.println("started");
     Socket s=ss.accept();
     System.out.println("client connected");
     DataInputStream input=new DataInputStream(s.getInputStream());
     DataOutputStream out=new DataOutputStream(s.getOutputStream());
     System.out.print("Waiting time is:");
     int t=Integer.parseInt(input.readUTF());
     System.out.println(t);
     System.out.println("Do you want to receive frames with errors(1/0)?");
     Scanner sc=new Scanner(System.in);
     int d=sc.nextInt();
     String ack="Received";
     if(d==0){
```

```
      while(true){
       String str=input.readUTF();
       System.out.println(str);
       out.writeUTF(ack);
       System.out.println(ack);
     }
    }
    else{
      while(true){
        String str1=input.readUTF();
        System.out.println(str1);
        out.writeUTF(ack);
        System.out.println(ack);
        String temp=input.readUTF();
        out.writeUTF("Not");
        System.out.println("Not Received");
        for(int i=0;i<1000*t;i++);
        str1=input.readUTF();
        out.writeUTF(ack);
        System.out.println(str1);
        System.out.println(ack);
       }
    }
   }
    public static void main(String args[])throws
UnknownHostException,IOException{
```

```
        serverStopWaitARQ ssw=new serverStopWaitARQ(5000);
    }
}
```

clientSW.java

```
import java.io.*;
import java.util.*;
import java.net.*;
public class clientStopWaitARQ{
   public clientStopWaitARQ(String ip,int port)throws
UnknownHostException,IOException{
      Socket s=new Socket(ip,port);
      System.out.println("connected");
      DataOutputStream out=new DataOutputStream(s.getOutputStream());
      DataInputStream input=new DataInputStream(s.getInputStream());
      Scanner sc=new Scanner(System.in);
      System.out.println("Enter the waiting time in seconds");
      int t=sc.nextInt();
      out.writeUTF(""+t);
      System.out.println("");
      int fn=0;
      while(true){
        System.out.println("Enter data Frame"+fn);
        String df=sc.next();
        out.writeUTF(df);
        String ack=input.readUTF();
        if(!ack.equals("Received")){
```

```
            for(int i=0;i<1000*t;i++);
            System.out.println("Retransmitting "+df);
            out.writeUTF(df);
            ack=input.readUTF();
          }
          fn++;
        }
    }
    public static void main(String args[])throws UnknownHostException,IOException{
       clientStopWaitARQ csw=new clientStopWaitARQ("127.0.0.1",5000);
    }
}
```

OUTPUT

GO BACK N PROTOCOL

serverGBN.java

```
import java.util.*;
import java.io.*;
import java.net.*;
public class serverGoBackN{
    public serverGoBackN(int port)throws UnknownHostException,IOException{
     ServerSocket ss=new ServerSocket(port);
```

```
System.out.println("started");
Socket s=ss.accept();
System.out.println("client connected");
DataInputStream input=new DataInputStream(s.getInputStream());
DataOutputStream out=new DataOutputStream(s.getOutputStream());
System.out.print("Waiting time is:");
int t=Integer.parseInt(input.readUTF());
System.out.println(t);
System.out.print("Server window size is:");
int ws=Integer.parseInt(input.readUTF());
System.out.println(ws);
int i=0;
System.out.println("Do you want to receive frames with errors(1/0)?");
Scanner sc=new Scanner(System.in);
int d=sc.nextInt();
String ack="Received";
if(d==0){
 while(true){
  String str=input.readUTF();
  System.out.println(str);
  out.writeUTF(ack);
  System.out.println(ack);
 }
}
else{
  while(true){
```

```
 i=0;
 while(i<ws){
   String str=input.readUTF();
   System.out.println(str);
   out.writeUTF(ack);
   System.out.println(ack);
   i++;
}
i=0;
 while(i<ws){
 if(i<ws){
  String str1=input.readUTF();
  System.out.println(str1);
  out.writeUTF(ack);
  }
  i++;
 if(i<ws){
  System.out.println(ack);
  String temp=input.readUTF();
  out.writeUTF("Not");
  System.out.println("Not Received");
  }
  i++;
 }
 i=0;
 while(i<ws){
```

```
        String str=input.readUTF();
        System.out.println(str);
        out.writeUTF(ack);
        System.out.println(ack);
        i++;
      }
      i=0;
     }
    }
   }
    public static void main(String args[])throws
UnknownHostException,IOException{
       serverGoBackN gbn=new serverGoBackN(5000);
    }
}
```

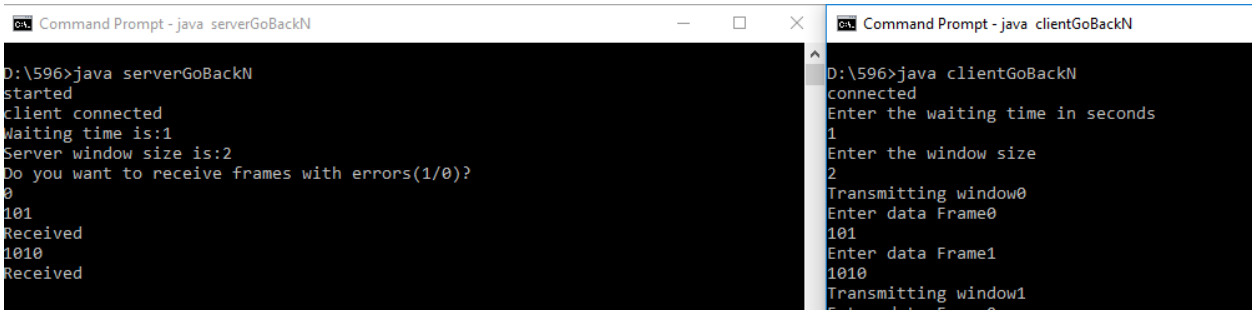
clientGBN.java

```
import java.io.*;
import java.util.*;
import java.net.*;
public class clientGoBackN{
   public clientGoBackN(String ip,int port)throws
UnknownHostException,IOException{
      Socket s=new Socket(ip,port);
      System.out.println("connected");
      DataOutputStream out=new DataOutputStream(s.getOutputStream());
      DataInputStream input=new DataInputStream(s.getInputStream());
      Scanner sc=new Scanner(System.in);
```

```
System.out.println("Enter the waiting time in seconds");
int t=sc.nextInt();
out.writeUTF(""+t);
System.out.println("Enter the window size");
int ws=sc.nextInt();
String dfs[]=new String[ws+1];
int acks[]=new int[ws+1];
int i=0,j=0;
out.writeUTF(""+ws);
while(true){
 System.out.println("Transmitting window"+j);
 while(i<ws){
   System.out.println("Enter data Frame"+i);
   String df=sc.next();
   dfs[i]=df;
   out.writeUTF(df);
   String ack=input.readUTF();
   if(!ack.equals("Received")){
      acks[i]=0;
    }
    else{
      acks[i]=1;
    }
    i++;
}
for(i=0;i<ws;i++){
```

```
        if(acks[i]==0)
         break;
      }
      if(i!=ws){
         System.out.println("Retransmitting window"+j);
         for(i=0;i<ws;i++){
            out.writeUTF(dfs[i]);
            System.out.println(dfs[i]);
            String ack=input.readUTF();
         }
      }
     i=0;
     j++;
    }
   }
   public static void main(String args[])throws UnknownHostException,IOException{
      clientGoBackN gbn=new clientGoBackN("127.0.0.1",5000);
   }
}
```

OUTPUT



SELECTIVE REPEAT PROTOCOL

serverSR.java

```
import java.util.*;
import java.io.*;
import java.net.*;
public class serverSelectiveRepeat{
    public static void printdfs(String arr[]){
       System.out.println("Received window");
        for(int i=0;i<arr.length;i++){
          System.out.println(arr[i]);
        }
    }
    public serverSelectiveRepeat(int port)throws
UnknownHostException,IOException{
     ServerSocket ss=new ServerSocket(port);
     System.out.println("started");
     Socket s=ss.accept();
     System.out.println("client connected");
     DataInputStream input=new DataInputStream(s.getInputStream());
     DataOutputStream out=new DataOutputStream(s.getOutputStream());
     System.out.print("Waiting time is:");
     int t=Integer.parseInt(input.readUTF());
     System.out.println(t);
     System.out.print("Server window size is:");
     int ws=Integer.parseInt(input.readUTF());
     System.out.println(ws);
```

```
String dfs[]=new String[ws];
int acks[]=new int[ws];
int i=0;
System.out.println("Do you want to receive frames with errors(1/0)?");
Scanner sc=new Scanner(System.in);
int d=sc.nextInt();
String ack="Received";
if(d==0){
 while(true){
  while(i<ws){
    String str=input.readUTF();
    out.writeUTF(ack);
    System.out.println(ack);
    dfs[i]=str;
    acks[i]=1;
   i++;
 }
 serverSelectiveRepeat.printdfs(dfs);
 i=0;
 }
}
else{
  while(true){
    while(i<ws){
      String str=input.readUTF();
      out.writeUTF(ack);
```

```
  System.out.println(ack);
  dfs[i]=str;
  acks[i]=1;
  i++;
}
serverSelectiveRepeat.printdfs(dfs);
i=0;
while(i<ws){
 if(i<ws){
  String str1=input.readUTF();
  out.writeUTF(ack);
   dfs[i]=str1;
   acks[i]=1;
  System.out.println(ack);
 }
 i++;
 if(i<ws){
  dfs[i]=input.readUTF();
  out.writeUTF("Not");
  acks[i]=0;
  System.out.println("Not Received");
 }
 i++;
}
i=0;
while(i<ws){
```

```
        if(acks[i]==0){
        out.writeUTF(ack);
        System.out.println(ack);
       }
       i++;
      }
      serverSelectiveRepeat.printdfs(dfs);
      i=0;
     }
    }
   }
    public static void main(String args[])throws
UnknownHostException,IOException{
       serverSelectiveRepeat ssr=new serverSelectiveRepeat(5000);
   }
}
```

clientSR.java

```
import java.io.*;
import java.util.*;
import java.net.*;
public class clientSelectiveRepeat{
   public clientSelectiveRepeat(String ip,int port)throws
UnknownHostException,IOException{
      Socket s=new Socket(ip,port);
      System.out.println("connected");
      DataOutputStream out=new DataOutputStream(s.getOutputStream());
      DataInputStream input=new DataInputStream(s.getInputStream());
```

```
Scanner sc=new Scanner(System.in);
System.out.println("Enter the waiting time in seconds");
int t=sc.nextInt();
out.writeUTF(""+t);
System.out.println("Enter the window size");
int ws=sc.nextInt();
String dfs[]=new String[ws];
int acks[]=new int[ws];
int i=0,j=0;
out.writeUTF(""+ws);
while(true){
 System.out.println("Transmitting window"+j);
 while(i<ws){
   System.out.println("Enter data Frame"+i);
   String df=sc.next();
   dfs[i]=df;
   out.writeUTF(df);
   String ack=input.readUTF();
   if(!ack.equals("Received")){
      acks[i]=0;
    }
    else{
      acks[i]=1;
    }
    i++;
}
```

```
      for(i=0;i<ws;i++){
        if(acks[i]==0)
         break;
      }
      if(i!=ws){
         System.out.println("Retransmitting lost frames in window"+j);
         for(i=0;i<ws;i++){
          if(acks[i]==0){
            System.out.println(dfs[i]);
            String ack=input.readUTF();
          }
        }
      }
     i=0;
     j++;
    }
   }
   public static void main(String args[])throws UnknownHostException,IOException{
      clientSelectiveRepeat csr=new clientSelectiveRepeat("127.0.0.1",5000);
   }
}
```

OUTPUT