NETWORK PROGRAMMING
LAB MANUAL

INDEX

Hardware requirements:

Processor	: P-IV (2.4 GHZ)
RAM	: 256MB
HARDDISK	: 40GB
Other	: router, switch 16 port

Software requirement:

Operating system	: windows 2003, windows XP
Language	; java 2.0

Rationale Behind the Network Programming lab

This course presents a systematic introduction to the principles and practices of configuring and maintaining computer systems and networks. Today, computer technology is having a major impact as a result of linking them into networks, and perhaps more dramatically as the biggest network of all, referred as the INTERNET has changed the face of business, industry, government, education and almost every
aspect of human activity. But, how does all this actually work? What are the protocols? How can we develop programs/systems that actually execute in separate computers? What does the client/server model mean? The course offers a top-down approach to investigating the layers and components of network technology and provides an understanding of networked systems. This course advances the introduction to networks provided in 1007ICT and integrates this with some practical programming examples.

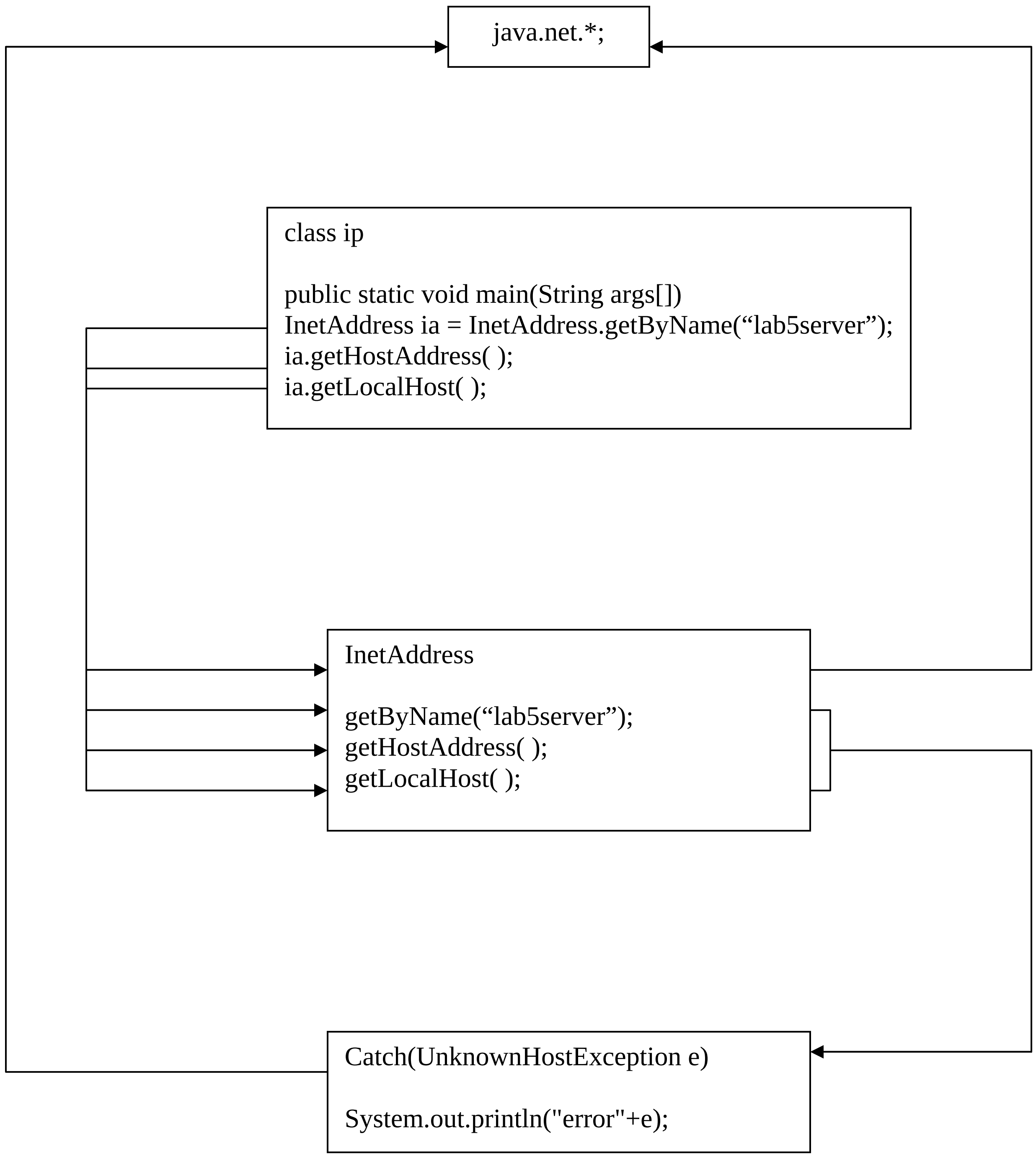
LEARNING OUTCOMES
The students should achieve a deep understanding of the protocol stack in widely available computer networks. It should enable the students to consider programming client/server systems over transport layer protocols. The students should understand the functionality of network layers in detail, with the potential to eventually develop or implement simple versions of tasks like packetization, control flow, error correction, network flow control and security. Students should have a detail understanding of headers for some protocols in some of the network layers, and also a solid understanding of the functionality of each component. Students should understand what is involved in networking a computer on a wired as well as wireless access point. This course will offer content-based outcomes in terms of understanding the
concepts of the network layer stack. However, it will include cognitive outcomes (e.g.: understanding, analysis, evaluation), as students will be required to contrast concept like switching networks vs. packaging networks.

Students will solve some implementation problems in their assessment items and as such application outcomes (i.e.: skills-based outcomes eg: problem solving) will be achieved.
Therefore, specific outcomes are as follows:

1. - Achieve an understanding of the functionality of each layer of the network stack.
2. - Achieve an understanding of the algorithms that achieve the functionality of the network stack.
3. - Recognize the packet system and protocol of layers of a network stack
4. - Recognize the features of client/server systems and programs, with a view to be able to implement
simple systems in this model.
5, - Analyse, develop and implement the client and server of a simple program over a transport layer.
6. - Evaluate the advantages and disadvantages of the alternative technologies for networking as they
have evolved in the recent years.

Experiment no 1.

Objective: Program to manipulate the IP address of a system.


java.net.*;
class ip
public static void main(String args[])
InetAddress ia = InetAddress.getByName("lab5server");
ia.getHostAddress( );
ia.getLocalHost( );
InetAddress
getByName("lab5server");
getHostAddress( );
getLocalHost( );
Catch(UnknownHostException e)
System.out.println("error"+e);

Source code

```
import java.net.*;
class InetDemo
{ public static void main(String args[])
{try
{InetAddress ia = InetAddress.getLocalHost();
System.out.println("The IP address of local host is "+ia);
ia=InetAddress.getByName(args[0]);
System.out.println("the IP address of "+args[0]+"is"+ia);
}
catch(UnknownHostException ue)
{
System.out.println("There is an error "+ue);
}
}
}
```

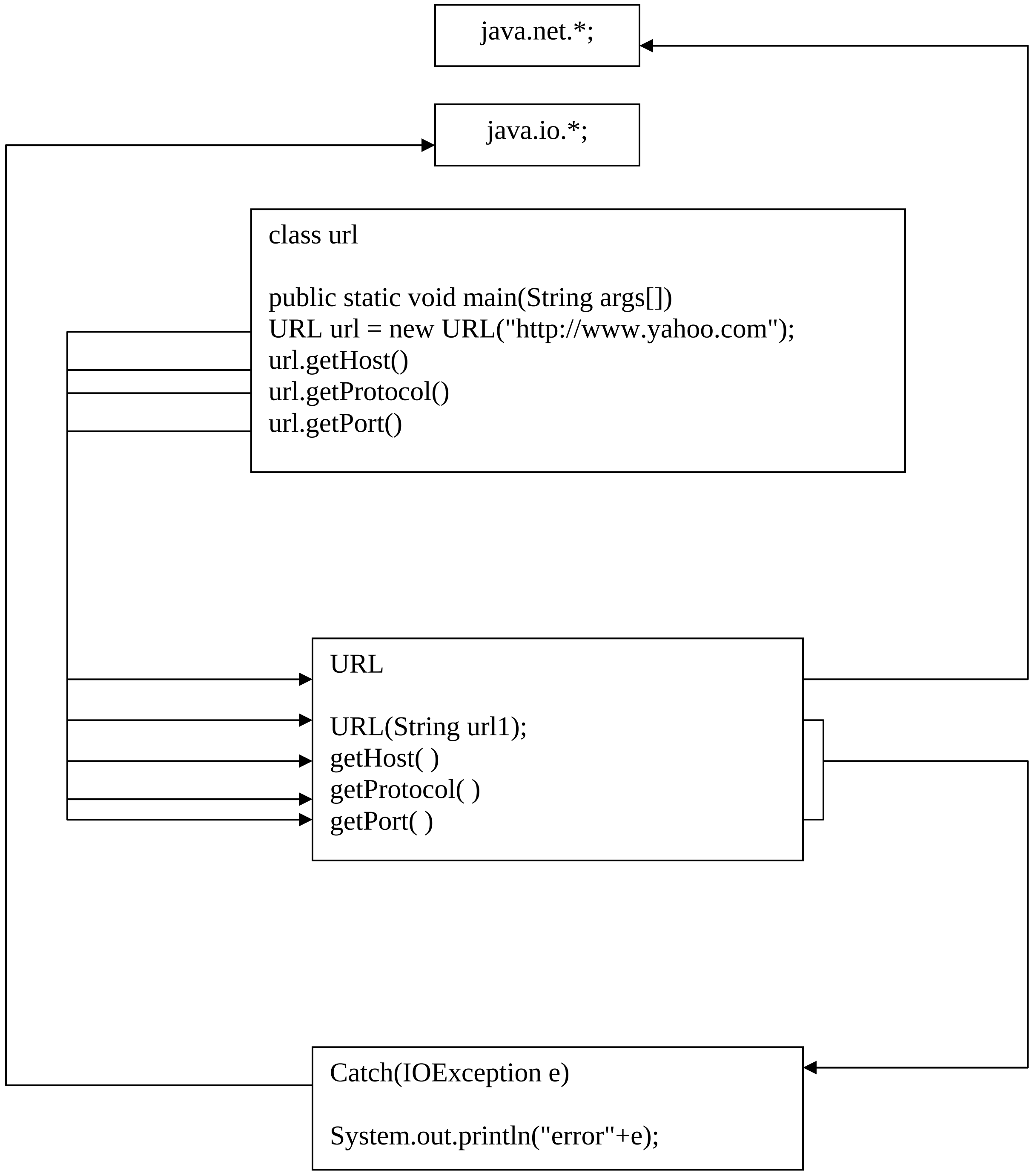
Output

```
C:\JAVA\BIN>javac InetDemo.java

C:\JAVA\BIN>java InetDemo
The IP address of local host is ececom5/192.168.1.175
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 0
 at InetDemo.main(InetDemo.java:7)

C:\JAVA\BIN>
```

Experiment no. 2
Objective:Program to obtain the information about the (a)Host (b)Port(c)



Source code

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

class ud1
        {
        public static void main(String args []) throws
        MalformedURLException
                { URL url = new URL("http://www.yahoo.com");
                        try
                        {
                                System.out.println("host name is " + url.getHost());
                                System.out.println("port no. is " + url.getPort());
                                System.out.println("protocol used is " + url.getProtocol());
                        }
                                catch (Exception e)
                                {       System.out.println("error"+e);
                                }
                }
        }
```

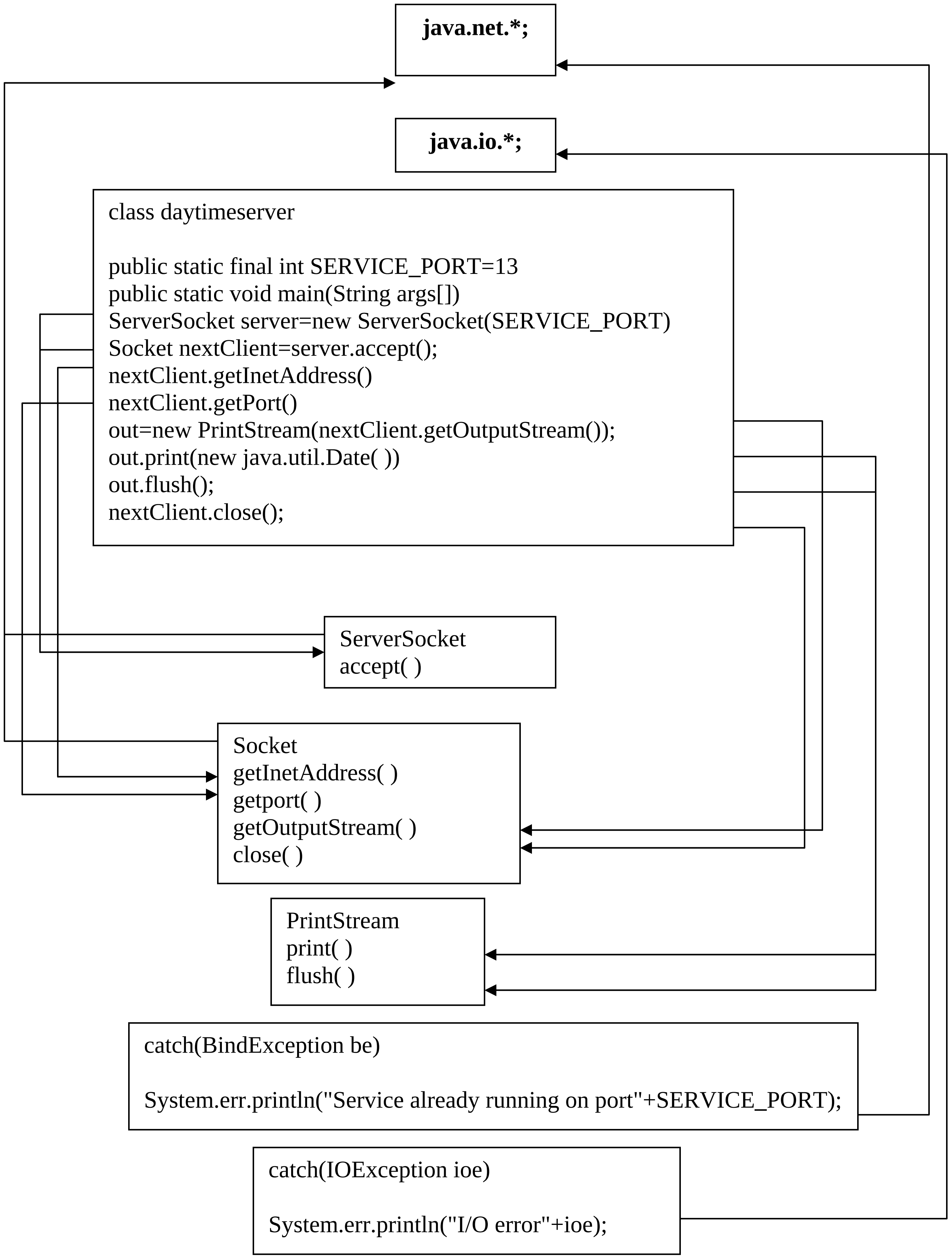
OUTPUT

```
C:\JAVA\BIN>javac ud1.java
C:\JAVA\BIN>java ud1 yahoo.com
host name is www.yahoo.com
port no. is -1
protocol used is http

C:\JAVA\BIN
```

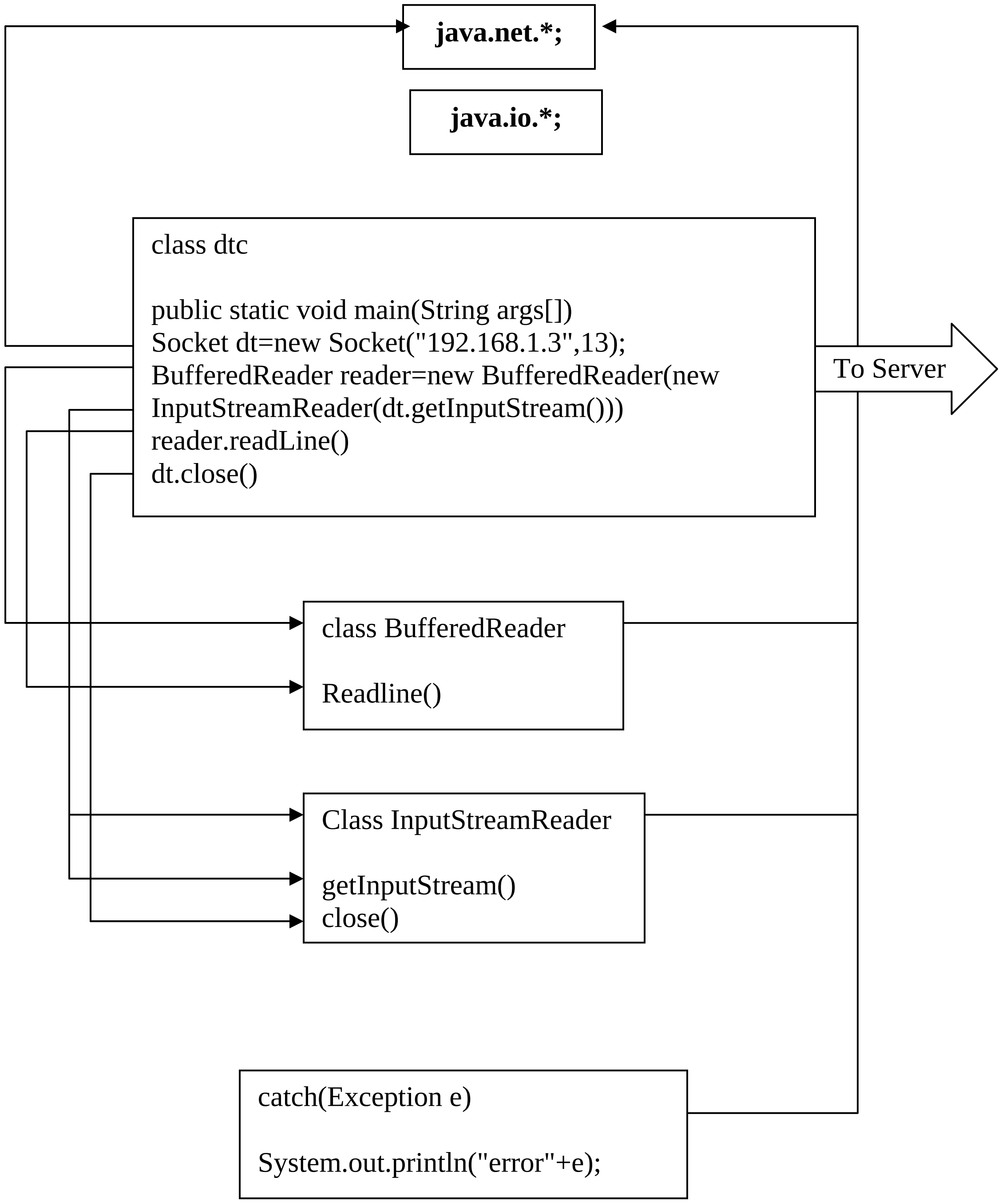
Experiment no 3.

Objective: Program to access daytime service from server using socket



java.net.*;
java.io.*;
class daytimeserver
public static final int SERVICE_PORT=13
public static void main(String args[])
ServerSocket server=new ServerSocket(SERVICE_PORT)
Socket nextClient=server.accept();
nextClient.getInetAddress()
nextClient.getPort()
out=new PrintStream(nextClient.getOutputStream());
out.print(new java.util.Date( ))
out.flush();
nextClient.close();
ServerSocket
accept( )
Socket
getInetAddress( )
getport( )
getOutputStream( )
close( )
PrintStream
print( )
flush( )
catch(BindException be)
System.err.println("Service already running on port"+SERVICE_PORT);
catch(IOException ioe)
System.err.println("I/O error"+ioe);

DAYTIME CLIENT



java.net.*;
java.io.*;
class dtc
public static void main(String args[])
Socket dt=new Socket("192.168.1.3",13);
BufferedReader reader=new BufferedReader(new
InputStreamReader(dt.getInputStream()))
reader.readLine()
dt.close()
To Server
class BufferedReader
Readline()
Class InputStreamReader
getInputStream()
close()
catch(Exception e)
System.out.println("error"+e);

Source code

```
import  java.net.*;
import  java.io.*;
class daytime

{
public static void main(String args[])
{
try
{
Socket daytime=new Socket("192.168.1.7",13);
System.out.println("Connection established");
daytime.setSoTimeout(2000);
BufferedReader reader=new BufferedReader(new
InputStreamReader(daytime.getInputStream()));
System.out.println("result:"+reader.readLine());
daytime.close();
}
catch(Exception ioe)
{
System.err.println("Error" +ioe);
}
}
}
```

OUTPUT

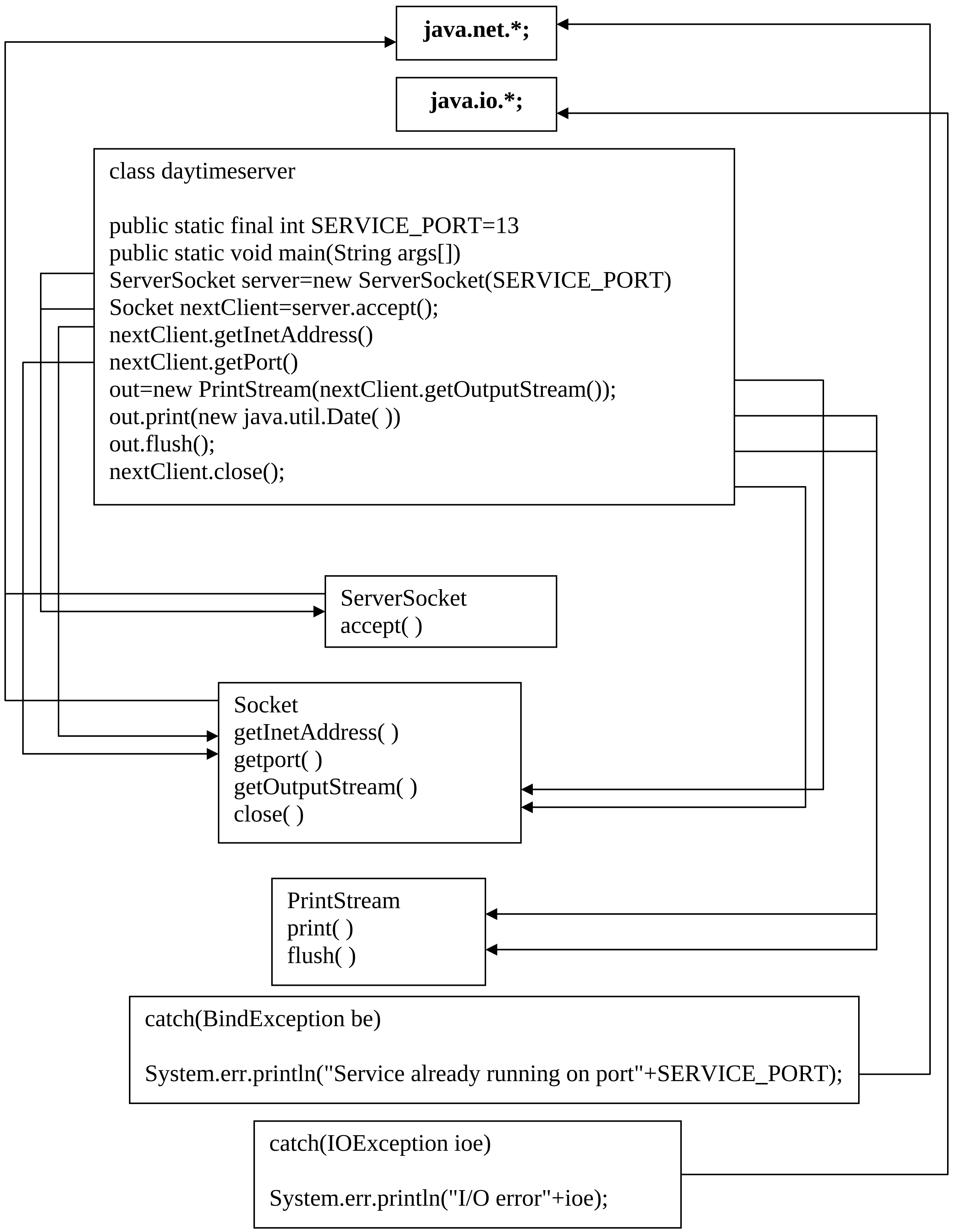
ON CLIENT

```
C:\JAVA\BIN>javac daytime1.java
C:\JAVA\BIN>java daytime1
Connection established
result:Wed Mar 29 11:01:48 GMT+05:30 2006
C:\JAVA\BIN>
```

ON SERVER

```
C:\java\bin>
C:\java\bin>
C:\java\bin>java daytimeserver
Daytime service started
Received request from /192.168.1.175:3043
```

Experiment no 4.
Objective: Program to get remote and local socket address.



```
java.net.*;
```

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

```
class daytimeserver

public static final int SERVICE_PORT=13
public static void main(String args[])
ServerSocket server=new ServerSocket(SERVICE_PORT)
Socket nextClient=server.accept();
nextClient.getInetAddress()
nextClient.getPort()
out=new PrintStream(nextClient.getOutputStream());
out.print(new java.util.Date( ))
out.flush();
nextClient.close();
```

```
ServerSocket
accept( )
```

```
Socket
getInetAddress( )
getport( )
getOutputStream( )
close( )
```

```
PrintStream
print( )
flush( )
```

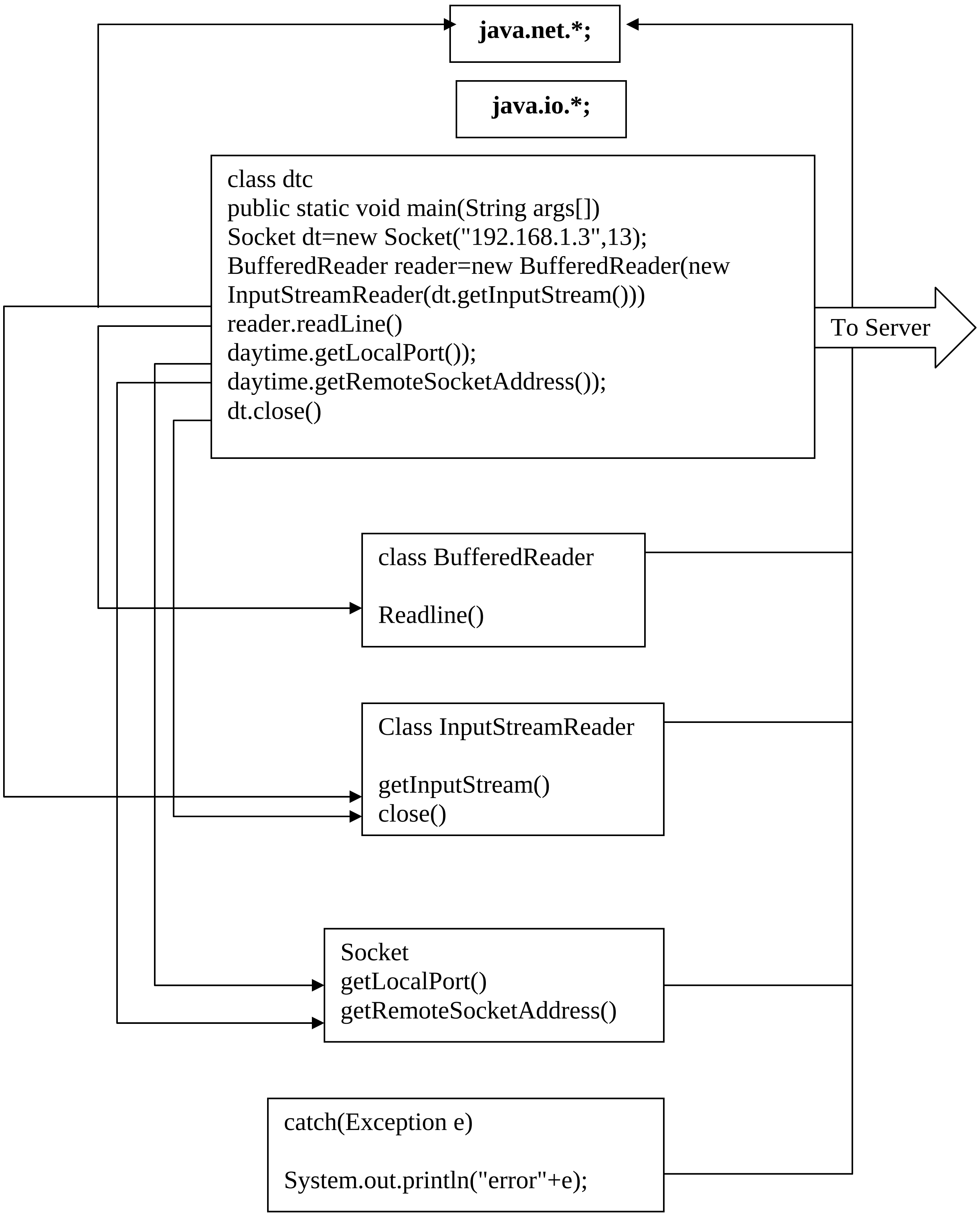
```
catch(BindException be)

System.err.println("Service already running on port"+SERVICE_PORT);
```

```
catch(IOException ioe)

System.err.println("I/O error"+ioe);
```

DAYTIME CLIENT



Source code

```
import  java.net.*;
import  java.io.*;
class daytime

{
public static void main(String args[])
{
try
{
Socket daytime=new Socket("192.168.1.7",13);
System.out.println("Connection established");
daytime.setSoTimeout(2000);
BufferedReader reader=new BufferedReader(new
InputStreamReader(daytime.getInputStream()));
System.out.println("result:"+reader.readLine());
System.out.println("local socket address"+daytime.getLocalPort());
System.out.println("remote socket address"+daytime.getRemoteSocketAddress());
daytime.close();
}
catch(Exception ioe)
{
System.err.println("Error" +ioe);
}
}
}
```

OUTPUT

ENTON CLI

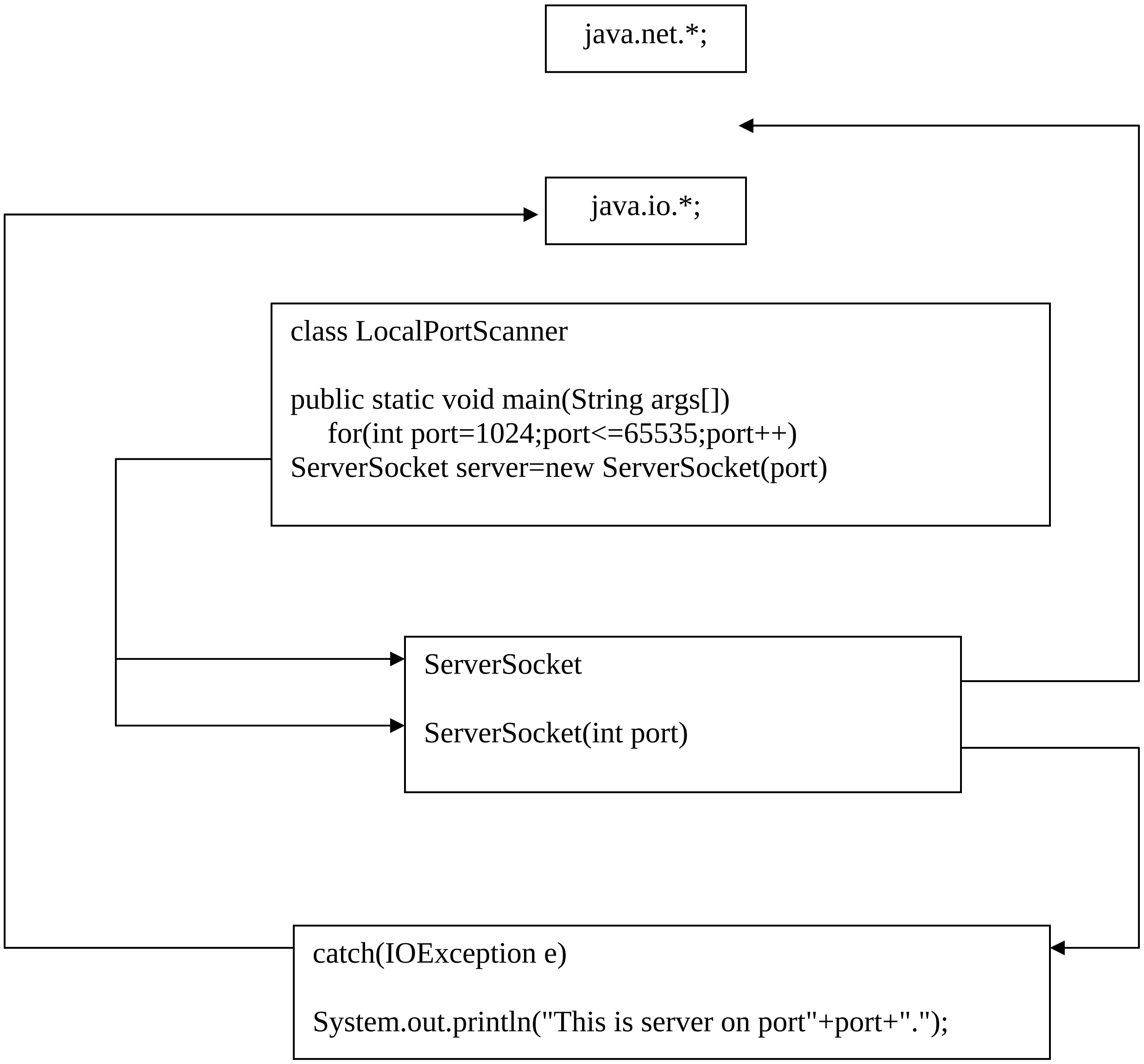
```
C:\JAVA\BIN>javac daytime.java

C:\JAVA\BIN>java daytime
Connection established
result:Wed Mar 29 10:32:43 GMT+05:30 2006
local socket address4200
remote socket address/192.168.1.7:13
C:\JAVA\BIN>
```

ON SERVER

```
C:\java\bin>java daytimeserver
Daytime service started
Received request from /192.168.1.175:4200
```

Experiment no. 5
Objective: Program to find port no running on server.


java.net.*;
java.io.*;
class LocalPortScanner
public static void main(String args[])
for(int port=1024;port<=65535;port++)
ServerSocket server=new ServerSocket(port)
ServerSocket
ServerSocket(int port)
catch(IOException e)
System.out.println("This is server on port"+port+".");

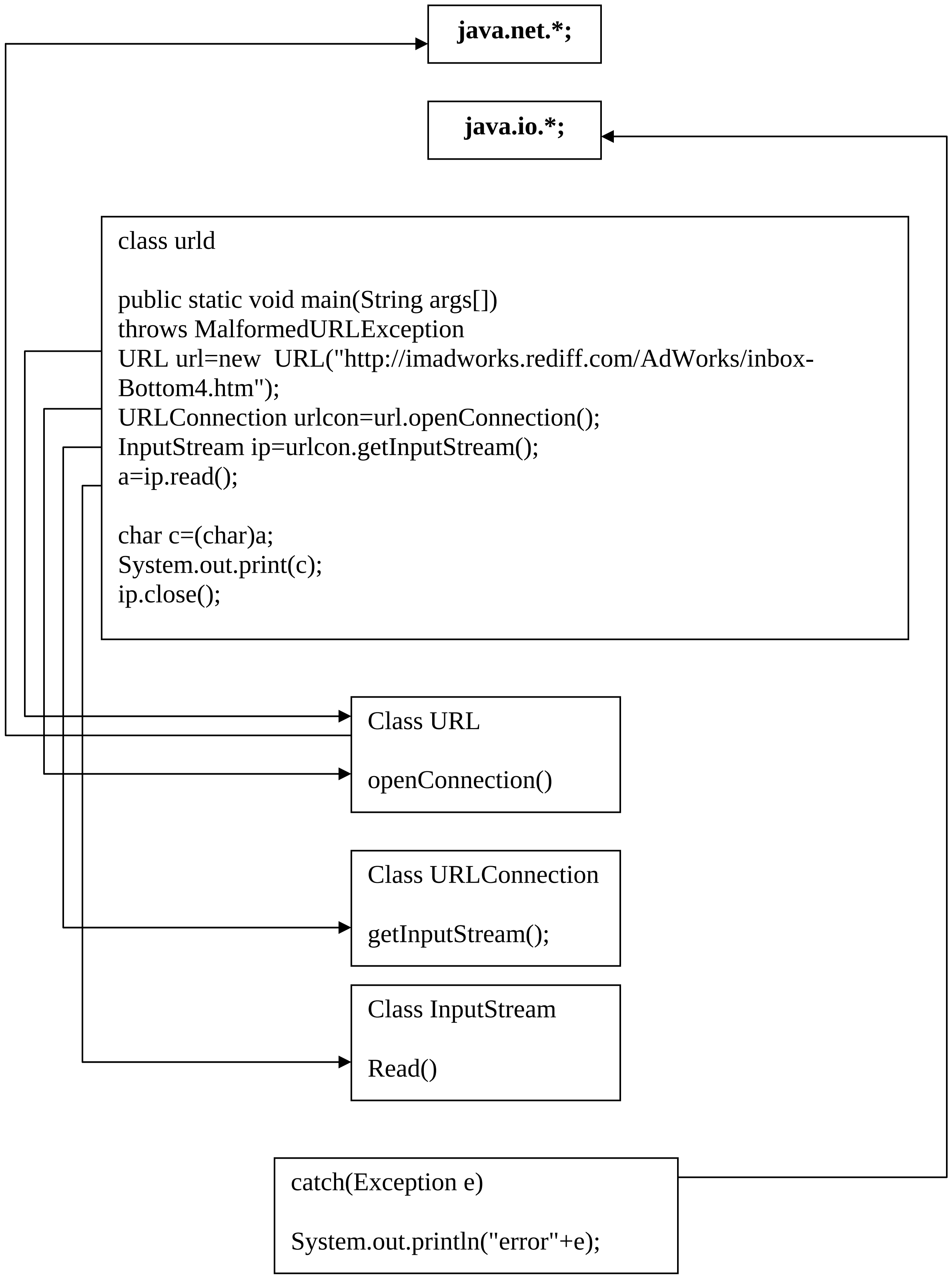
Source code

```
import java.net.*;
import java.io.*;
public class LocalPortScanner
{
public static void main(String args[])
{
for(int port=1024;port<=65535;port++)
{
try
{
ServerSocket server=new ServerSocket(port);
}
catch(IOException e)
{
System.out.println("There is a server on port"+port);
}
}
}
}
```

OUTPUT

```
C:\JAVA\BIN>javac LocalPortScanner.java
C:\JAVA\BIN>java LocalPortScanner
There is a server on port4314
There is a server on port4315
There is a server on port4316
There is a server on port4317
There is a server on port4318
C:\JAVA\BIN>
```

Experiment no 6
Objective: Program to read the source code of the web page


java.net.*;
java.io.*;
class urld
public static void main(String args[])
throws MalformedURLException
URL url=new URL("http://imadworks.rediff.com/AdWorks/inbox-Bottom4.htm");
URLConnection urlcon=url.openConnection();
InputStream ip=urlcon.getInputStream();
a=ip.read();
char c=(char)a;
System.out.print(c);
ip.close();
Class URL
openConnection()
Class URLConnection
getInputStream();
Class InputStream
Read()
catch(Exception e)
System.out.println("error"+e);

Source code

```
import java.lang.*;
import java.io.*;
import java.net.*;
class urld
{
public static void main(String args[]) throws MalformedURLException
{
try
{URL url=new URL("http://www.google.com");
URLConnection urlcon=url.openConnection();
InputStream ip=urlcon.getInputStream();
boolean flag=true;
while(flag)
{int a=ip.read();
if(a==-1)
{flag=false;
}
else
{
char c=(char)a;
System.out.print(c);
}
}
ip.close();
}
catch(Exception e)
{
System.out.println("error"+e);
}
}
                                                }
```

OUTPUT

```
C:\JAVA\BIN>javac urld.java

C:\JAVA\BIN>java urld
<html><head><meta http-equiv="content-type" content="text/html; charset=ISO-8859
-1"><title>Google</title><style><!--
body,td,a,p,.h{font-family:arial,sans-serif;}
.h{font-size: 20px;}
.q{color:#0000cc;}
//-->
</style>
<script>
<!--
function sf(){document.f.q.focus();}
// -->
</script>
</head><body bgcolor=#ffffff text=#000000 link=#0000cc vlink=#551a8b alink=#ff00
00 onLoad=sf() topmargin=3 marginheight=3><center><table border=0 cellspacing=0
cellpadding=0 width=100%><tr><td align=right nowrap><font size=-1><a href="/url?
</tr><tr><td class=h align=right valign=top><b></b></td><td valign=top><img src=
images/hp3.gif width=50 height=32 alt=""></td><td valign=top class=h><font color
=#6f6f6f style=font-size:16px><b>India</b></font></td></tr></table><br>
<form action=/search name=f><table border=0 cellspacing=0 cellpadding=4><tr><td
nowrap><font size=-1><b>Web</b>&nbsp;&nbsp;&nbsp;&nbsp;<a id=1a class=q
href="/i
=q>more&nbsp;&raquo;</a></b></font></td></tr></table><table cellspacing=0 cellpa
dding=0><tr><td width=25%>&nbsp;</td><td align=center><input type=hidden
name=hl
C:\JAVA\BIN>
```

Experiment no 7.
Objective: Program to create socket for sending and receiving data

Source code

Server

```
import java.net.*;
import java.io.*;
public class server {
public static void main(String args[]) {
 int port = 4917; // just a random port. make sure you enter something between 1025 and
65535.
 try {
    ServerSocket ss = new ServerSocket(port); /* create a server socket and bind it to the
above port number.*/
    System.out.println("Waiting for a client...");
 Socket socket = ss.accept(); // make the server listen for a connection, and let you know
when it gets one.
 System.out.println("Got a client :) ... ");
    System.out.println();
 // Get the input and output streams of the socket, so that you can receive and send data to
the client.
    InputStream sin = socket.getInputStream();
    OutputStream sout = socket.getOutputStream();
BufferedReader keyboard = new BufferedReader(new InputStreamReader(System.in));
// Just converting them to different streams, so that string handling becomes easier.
    DataInputStream in = new DataInputStream(sin);
    DataOutputStream out = new DataOutputStream(sout);
 String line = null;
    while(true) {
 line = in.readUTF(); // wait for the client to send a line of text.
 System.out.println(" client just sending the  line : " + line);
 line=keyboard.readLine();
```

```
    System.out.println("I'm sending it ..."+line);
 out.writeUTF(line); // send the same line back to the client.
   System.out.println("Waiting for the next line...");
    System.out.println();
ss.close();
}


} catch(Exception x) {

System.out.println("Exception caught"+x);
}
}
}
```

Client

```
import java.net.*;
import java.io.*;
public class Client {
public static void main(String[] ar) {


try {

 Socket socket = new Socket("192.168.1.7",4917); // create a socket with the server's IP address and server's port.
System.out.println("Yes! I just got hold of the program.");
 // Get the input and output streams of the socket, so that you can receive and send data to the client.
InputStream sin = socket.getInputStream();
OutputStream sout = socket.getOutputStream();
 // Just converting them to different streams, so that string handling becomes easier.
DataInputStream in = new DataInputStream(sin);
DataOutputStream out = new DataOutputStream(sout);
 // Create a stream to read from the keyboard.
BufferedReader keyboard = new BufferedReader(new InputStreamReader(System.in));
 String line = null;
System.out.println("Type in something and press enter. Will send it to the server and tell ya what it thinks.");
System.out.println();
 while(true) {
line = keyboard.readLine(); // wait for the user to type in something and press enter.
 System.out.println("Sending this line to the server...");
out.writeUTF(line); // send the above line to the server.
out.flush(); // flush the stream to ensure that the data reaches the other end.
line = in.readUTF(); // wait for the server to send a line of text.
 System.out.println("The server was very polite. It sent me this : " + line);
System.out.println("Looks like the server is pleased with us. Go ahead and enter more
lines.");
System.out.println();
}


} catch(Exception x) {
System.out.println("exception caught"+x);;
}
}
}
```

OUTPUT

Server

```
C:\java\bin>java server
Waiting for a client...
Got a client :) ...

 client just sending the  line : hi
hi
I'm sending it ...hi
Waiting for the next line...
```

Client

```
C:\java\bin>java client
Type something in and press enter
Will send it to the server

Hi
Sending line to server…
Text date back from the sever hi
If you want to enter more data then enter
```

New ideas beside university syllabus:-

1. PC to PC/peripherals communication

- a. Establish RS232 communication
- b. Establish Parallel port communication

2. MAC Layer LAN Protocols

Observe the behavior and measure the throughput, compare the performance with other MAC Layer protocols.

- a. CSMA/CD at MAC Layer
- b. Token Bus at MAC Layer
- c. Token Ring at MAC Layer
- d. CSMA/CA at MAC Layer

3. LLC (Logical Link Control) Layer LAN Protocols

Observe the behaviour and measure the throughput of reliable data transfer protocols. Compare the performance with other LLC Layer protocols.

- a. Stop & Wait at LLC Layer
- b. Sliding Window – Go-Back-N at LLC Layer
- c. Sliding Window – Selective Repeat at LLC Layer

4. Routing Algorithm

Performance Study of Routing Algorithms through simulation

- a. Distance Vector Routing
- b. Link State Routing

5. Introduction to Socket Communication in Linux & Windows

- a. Socket programming concept in Windows & Linux platforms
- b. File Transfer between PC's through sockets

FAQS for network programming

References

Java **Network** Programming, Third Edition:: Elliote Rusty Harold
Java Network programming-by Elliotte Rusty Harold
Java network-by David Reilly,Michael Reilly