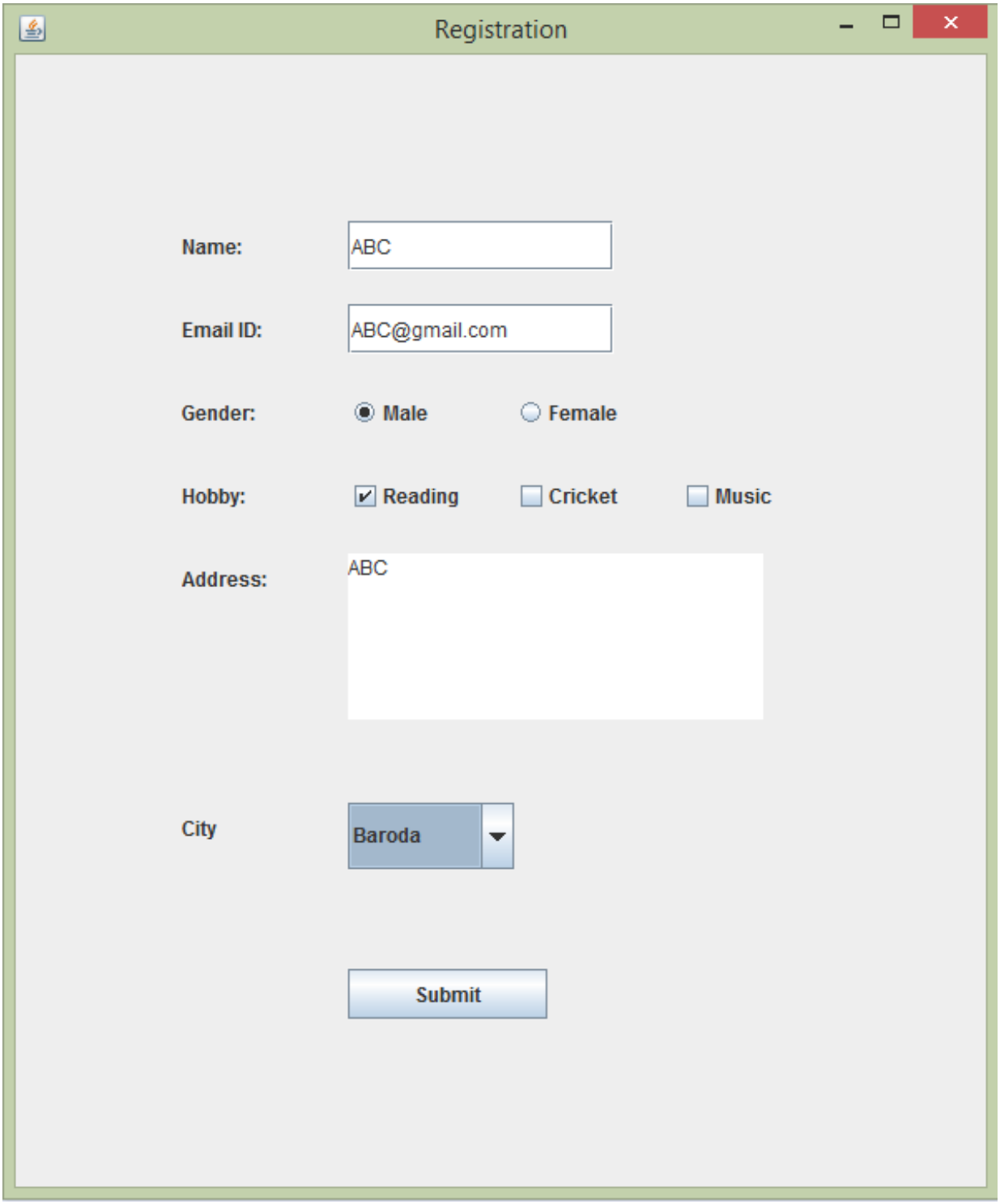
2 Applet GUI

2.2 Write an applet program to create following GUI structure

2.2.1 Source of `1a.java`

```
1  package swing;
2  import javax.swing.*;
3  import java.awt.*;
4  public class info extends JFrame {
5      JLabel l1;
6      JLabel l2;
7      JLabel l3;
8      JLabel l4;
9      JLabel l5;
10     JLabel l6;
11     JTextField name;
12     JTextField email;
13     JTextArea address;
14     JRadioButton Male, Female;
15     JCheckBox Cricket, Music, Reading;
16     JComboBox jcb;
17     JButton Submit;
18     Container c;
19     ButtonGroup bg;
20     public info() {
21         c = this.getContentPane();
22         c.setLayout(null);
23         l1 = new JLabel();
24         l2 = new JLabel();
25         l3 = new JLabel();
26         l4 = new JLabel();
27         l5 = new JLabel();
28         l6 = new JLabel();
29         l1.setText("Name:");
30         l2.setText("Email ID:");
31         l3.setText("Gender:");
32         l4.setText("Hobby:");
33         l5.setText("Address:");
34         l6.setText("City");
35         name = new JTextField(15);
36         email = new JTextField(15);
37         address =  new JTextArea(3, 8);
38         String[] data = {"Surat", "Baroda", "Ahemdabad"};
39         jcb = new JComboBox(data);
40         bg = new ButtonGroup();
41         Male = new JRadioButton("Male");
42         Female = new JRadioButton("Female");
43         Cricket  = new JCheckBox("Cricket");
44         Music = new JCheckBox("Music");
45         Reading = new JCheckBox("Reading");
46         Submit = new JButton("Submit");
47         l1.setBounds(100, 100, 60, 30);
```

```
48          name.setBounds(200, 100, 160, 30);
49          l2.setBounds(100, 150, 80, 30);
50          email.setBounds(200, 150, 160, 30);
51          l3.setBounds(100, 200, 80, 30);
52          Male.setBounds(200, 200, 60, 30);
53          Female.setBounds(300, 200, 100, 30);
54          l4.setBounds(100, 250, 60, 30);
55          Reading.setBounds(200, 250, 80, 30);
56          Cricket.setBounds(300, 250, 80, 30);
57          Music.setBounds(400, 250, 80, 30);
58          l5.setBounds(100, 300, 60, 30);
59          address.setBounds(200, 300, 250, 100);
60          l6.setBounds(100, 450, 40, 30);
61          jcb.setBounds(200, 450, 100, 40);
62          Submit.setBounds(200, 550, 120, 30);
63          c.add(l1);
64          c.add(name);
65          c.add(l2);
66          c.add(email);
67          c.add(l3);
68          bg.add(Male);
69          bg.add(Female);
70          c.add(Male);
71          c.add(Female);
72          c.add(l4);
73          c.add(Reading);
74          c.add(Cricket);
75          c.add(Music);
76          c.add(l5);
77          c.add(address);
78          c.add(l6);
79          c.add(jcb);
80          c.add(Submit);
81      }
82      public static void main(String atgs[]) {
83          info pr = new info();
84          pr.setVisible(true);
85          pr.setEnabled(true);
86          pr.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
87          pr.setSize(600, 720);
88          pr.setTitle("Registration");
89      }
90  }
```


Registration
Name:
ABC
Email ID:
ABC@gmail.com
Gender:
Male
Female
Hobby:
Reading
Cricket
Music
Address:
ABC
City
Baroda
Submit

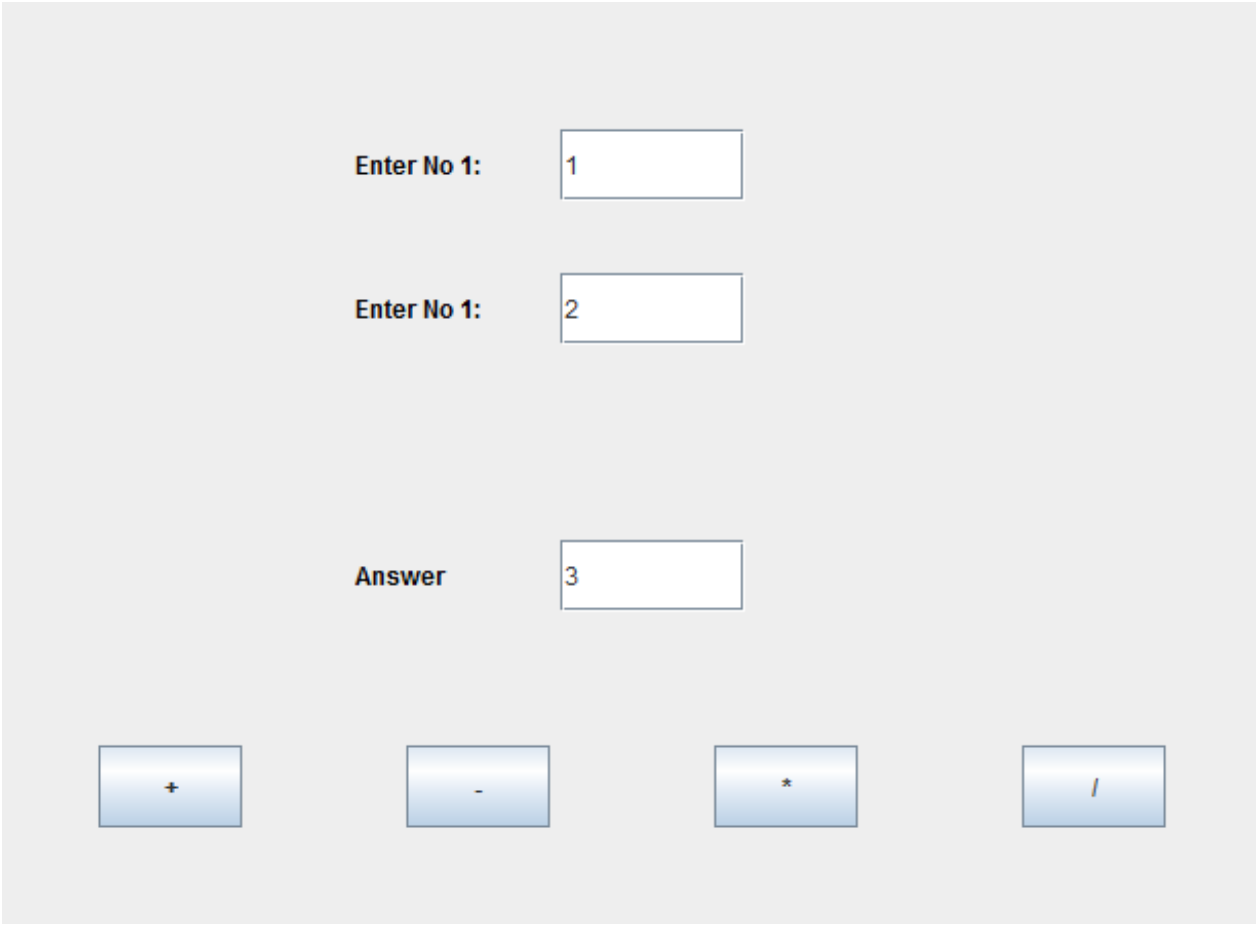
Output 2.2.1.1

2.3 Write an applet program to create following GUI structure

2.3.1 Source of `1b.java`

```
1  package swing;
2  import javax.swing.*;
3  import java.awt.event.*;
4  import java.awt.*;
5
6  public class cal extends JApplet implements
7      ActionListener {
8      Container c;
9      JLabel l1 = new JLabel();
10     JLabel l2 = new JLabel();
11     JLabel l3 = new JLabel();
12     JTextField no1 = new JTextField(4);
13     JTextField no2 = new JTextField(4);
14     JTextField no3 = new JTextField(4);
15     JButton addd = new JButton("+");
16     JButton subb = new JButton("-");
17     JButton mull = new JButton("*");
18     JButton divv = new JButton("/");
19
20     public void init() {
21         c = this.getContentPane();
22         c.setLayout(null);
23         l1.setText("Enter No 1:");
24         l2.setText("Enter No 1:");
25         l3.setText("Answer");
26         l1.setForeground(Color.BLACK);
27         l2.setForeground(Color.BLACK);
28         l3.setForeground(Color.BLACK);
29         l1.setBounds(525, 200, 90, 35);
30         no1.setBounds(625, 200, 90, 35);
31         l2.setBounds(525, 270, 90, 35);
32         no2.setBounds(625, 270, 90, 35);
33         l3.setBounds(525, 400, 90, 35);
34         no3.setBounds(625, 400, 90, 35);
35         addd.setBounds(400, 500, 70, 40);
36         subb.setBounds(550, 500, 70, 40);
37         mull.setBounds(700, 500, 70, 40);
38         divv.setBounds(850, 500, 70, 40);
39         c.add(l1);
40         c.add(no1);
41         c.add(l2);
42         c.add(no2);
43         c.add(l3);
44         c.add(no3);
45         c.add(addd);
46         c.add(subb);
47         c.add(mull);
48         c.add(divv);
49         no1.addActionListener(this);
```

```
50          no2.addActionListener(this);
51          no3.addActionListener(this);
52          addd.addActionListener(this);
53          subb.addActionListener(this);
54          mull.addActionListener(this);
55          divv.addActionListener(this);
56      }
57
58      @Override
59      public void actionPerformed(ActionEvent ae) {
60          String str = ae.getActionCommand();
61          int ans = 1;
62
63          if(no1.getText() != "" && no2.getText() != "") {
64              if(str.equals("+")) {
65                  ans = Integer.parseInt(no1.getText()) +
66                        Integer.parseInt(no2.getText());
67
68              } else if(str.equals("-")) {
69                  ans = Integer.parseInt(no1.getText()) -
70                        Integer.parseInt(no2.getText());
71
72              } else if(str.equals("*")) {
73                  ans =
74                      Integer.parseInt(no1.getText()) *
75                      Integer.parseInt(no2.getText());
76
77              } else if(str.equals("/")) {
78                  ans = Integer.parseInt(no1.getText()) /
79                        Integer.parseInt(no2.getText());
80              }
81
82              System.out.println(ans);
83              no3.setText(Integer.toString(ans));
84          }
85      }
86  }
```



Output 2.3.1.1

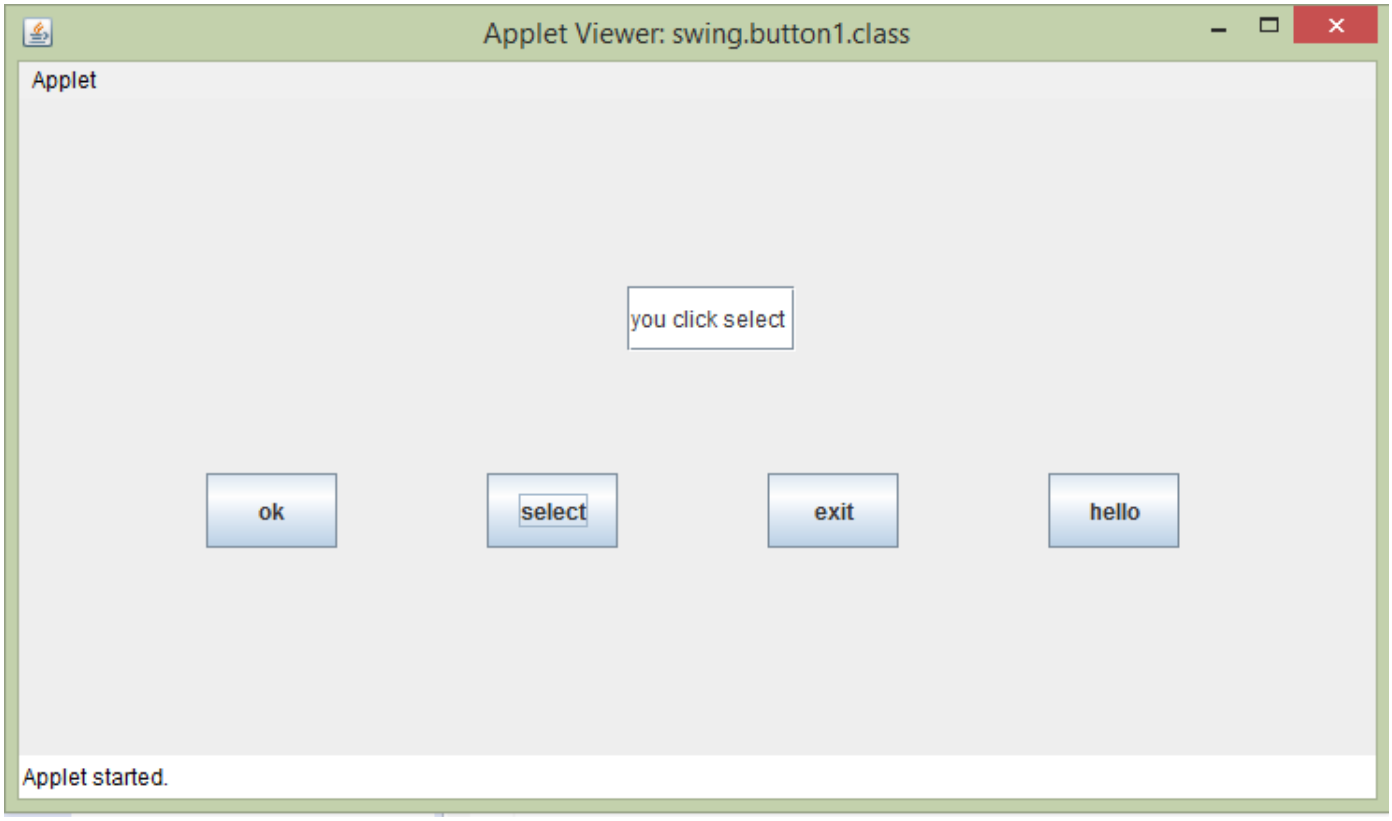
3 Swing application

3.1 Develop a swing application using event handling

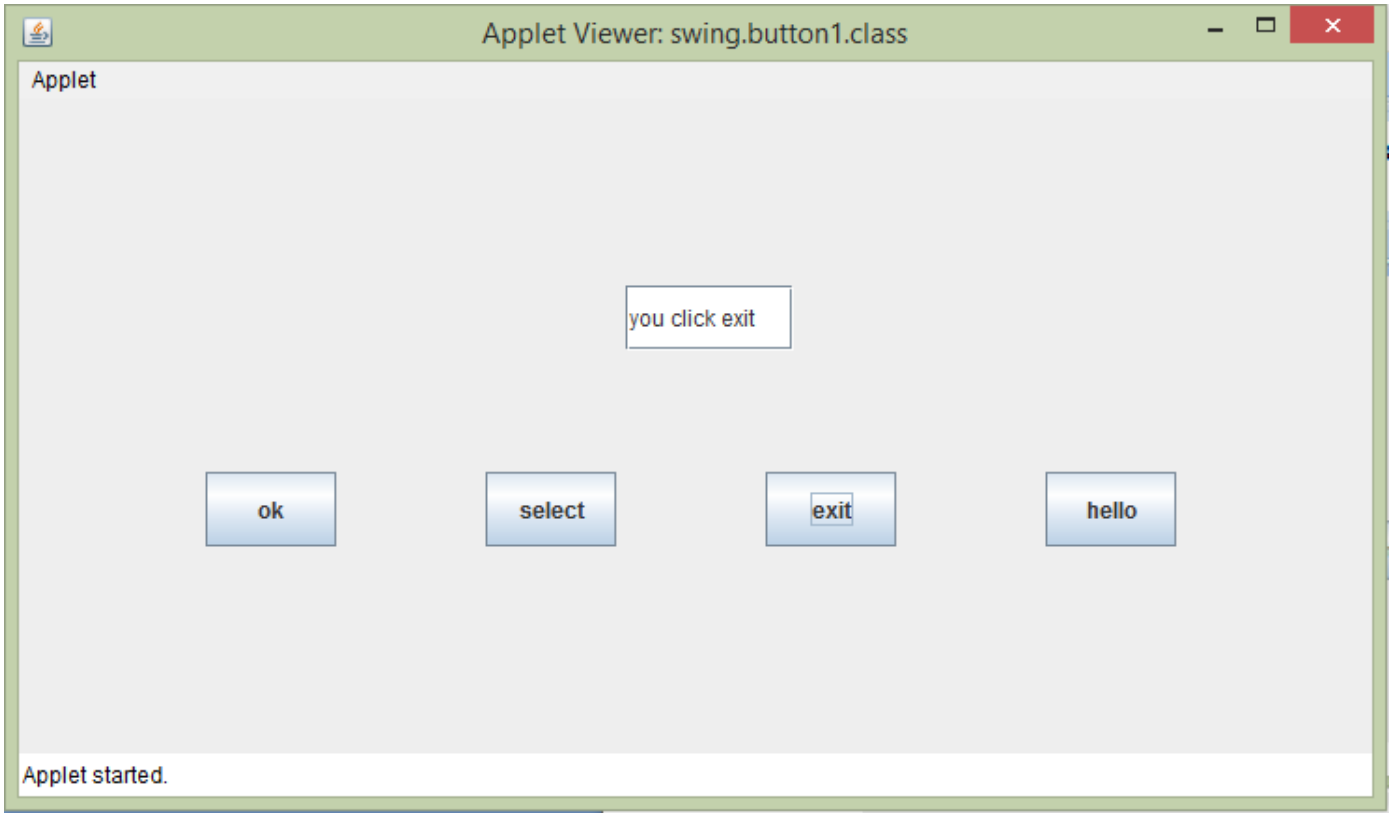
3.1.1 Source of 2a.java

```
1  package swing;
2  package swing;
3
4  import java.awt.Container;
5  import java.awt.event.ActionEvent;
6  import java.awt.event.ActionListener;
7
8  import javax.swing.JApplet;
9  import javax.swing.JButton;
10 import javax.swing.JLabel;
11 import javax.swing.JTextField;
12
13 public class button1 extends JApplet implements
14     ActionListener {
15     Container c;
16
17     JButton ok = new JButton("ok");
18     JButton select = new JButton("select");
19     JButton exit = new JButton("exit");
20     JButton hello = new JButton("hello");
21     JTextField ans = new JTextField(15);
22
23     public button1() {
24         c = this.getContentPane();
25         c.setLayout(null);
26         ans.setBounds(625, 400, 90, 35);
27         ok.setBounds(400, 500, 70, 40);
28         select.setBounds(550, 500, 70, 40);
29         exit.setBounds(700, 500, 70, 40);
30         hello.setBounds(850, 500, 70, 40);
31         c.add(ans);
32         c.add(ok);
33         c.add(select);
34         c.add(exit);
35         c.add(hello);
36         ok.addActionListener(this);
37         select.addActionListener(this);
38         exit.addActionListener(this);
39         hello.addActionListener(this);
40     }
41     @Override
42     public void actionPerformed(ActionEvent e) {
43         String str = e.getActionCommand();
44
45         if(str.equals("ok")) {
46             ans.setText("you click ok");
47
```

```
48          } else if(str.equals("select")) {

49              ans.setText("you click select");
50
51          } else if(str.equals("exit")) {
52              ans.setText("you click exit");
53
54          } else {
55              ans.setText("you click hello");
56          }
57      }
58  }
```



Output 3.1.1.1


Applet Viewer: swing.button1.class
Applet
you click exit
ok
select
exit
hello
Applet started.

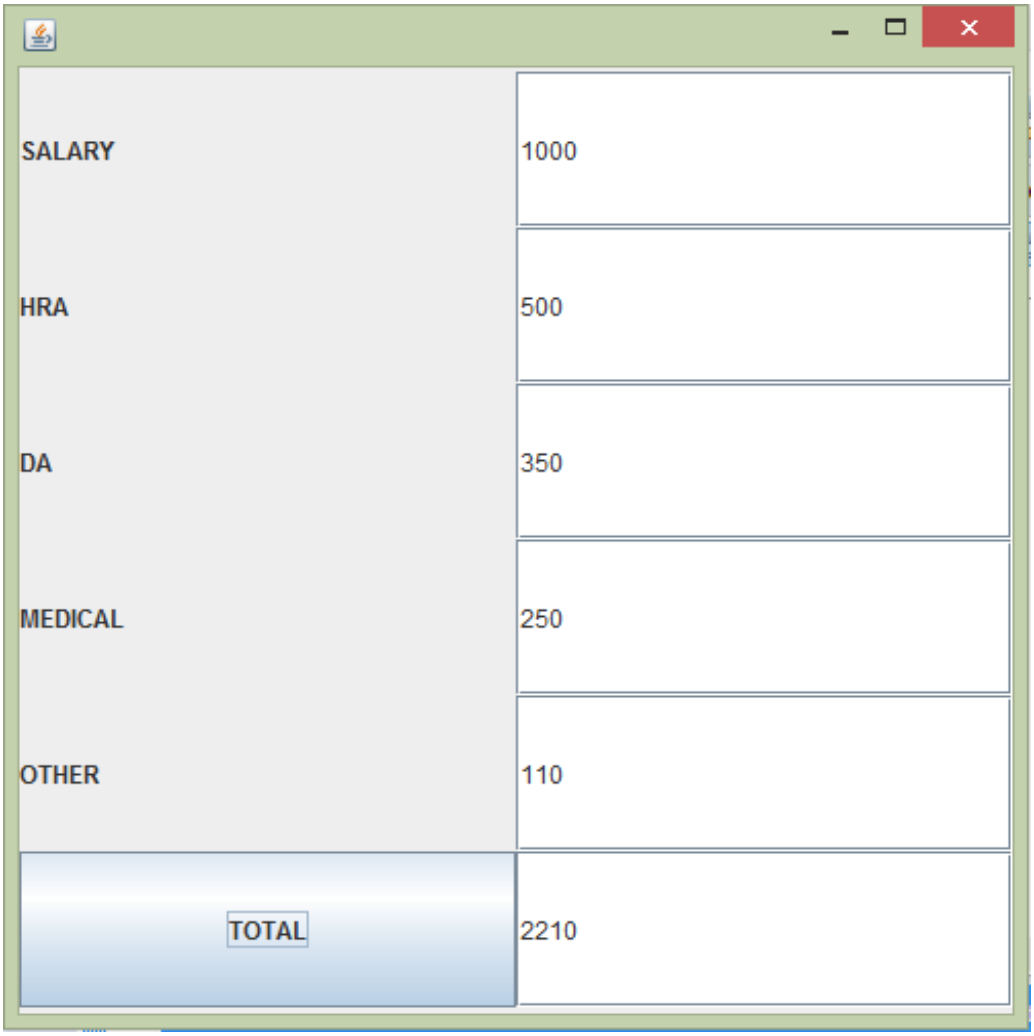
Output 3.1.1.2

3.2 Develop a swing application with different layouts

3.2.1 Source of 2b.java

```
1  package swing;
2
3  import java.awt.Container;
4  import java.awt.GridLayout;
5  import java.awt.event.ActionEvent;
6  import java.awt.event.ActionListener;
7  import javax.swing.JButton;
8  import javax.swing.JFrame;
9  import javax.swing.JLabel;
10 import javax.swing.JPanel;
11 import javax.swing.JTextField;
12
13 public class salary {
14     public static void main(String s[]) {
15         count c1 = new count();
16         c1.setVisible(true);
17     }
18 }
19
20 class count extends JFrame implements
21     ActionListener {
22     JLabel l1, l2, l3, l4, l5;
23     JTextField t1, t2, t3, t4, t5, total1;
24     JButton b1;
25     Container c;
26     JPanel p1;
27     count() {
28         c = new Container();
29         setSize(500, 500);
30         p1 = new JPanel();
31         b1 = new JButton("TOTAL");
32         l1 = new JLabel("SALARY");
33         l2 = new JLabel("HRA");
34         l3 = new JLabel("DA");
35         l4 = new JLabel("MEDICAL");
36         l5 = new JLabel("OTHER");
37         t1 = new JTextField(10);
38         t2 = new JTextField(10);
39         t3 = new JTextField(10);
40         t4 = new JTextField(10);
41         t5 = new JTextField(10);
42         total1 = new JTextField(10);
43         setLayout(new GridLayout(6, 2));
44         add(l1);
45         add(t1);
46         add(l2);
47         add(t2);
48         add(l3);
49         add(t3);
```

```
50          add(l4);

51          add(t4);
52          add(l5);
53          add(t5);
54          add(b1);
55          add(total1);
56          c.add(p1);
57          b1.addActionListener(this);
58      }
59
60      @Override
61      public void actionPerformed(ActionEvent arg0) {
62          // TODO Auto-generated method stub
63          String str = arg0.getActionCommand();
64          double ans = 0;
65          double a, b, c, d, e;
66
67          if(t1.getText() != "" && t2.getText() != "" &&
68                  t3.getText() != "" && t4.getText() != "" &&
69                  t5.getText() != "") {
70              a = Integer.parseInt(t1.getText());
71              b = a * (.50);
72              System.out.println(b);
73              t2.setText(Integer.toString((int) b));
74              c = a * (.35);
75              System.out.println(c);
76              t3.setText(Integer.toString((int) c));
77              d = a * (.25);
78              System.out.println(d);
79              t4.setText(Integer.toString((int) d));
80              e = a * (.11);
81              System.out.println(e);
82              t5.setText(Integer.toString((int) e));
83              ans = a + b + c + d + e;
84              System.out.println(ans);
85              total1.setText(Integer.toString((int) ans));
86          }
87      }
88  }
```



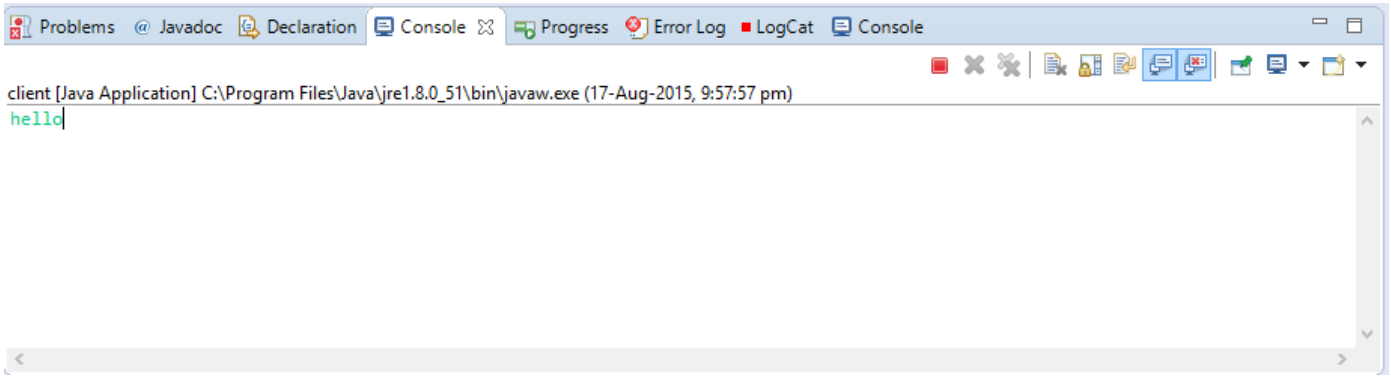
Output 3.2.1.1

4 Write a program to implement the connection oriented echo client server application

4.1 Source of 6_1.java

```
1  package connection;
2  import java.io.BufferedReader;
3  import java.io.IOException;
4  import java.io.InputStream;
5  import java.io.InputStreamReader;
6  import java.io.OutputStream;
7  import java.net.*;
8
9  public class client {
10
11     Socket s;
12     InputStream is;
13     OutputStream os;
14     BufferedReader br = new BufferedReader(
15     new InputStreamReader(System.in) {
16         @Override
17         public int read() throws IOException {
18             return 0;
19         }
20     });
21     client() {
22         // TODO Auto-generated constructor stub
23         try {
24             s = new Socket("127.0.0.1", 1000);
25
26         } catch(IOException e) {
27             // TODO Auto-generated catch block
28             e.printStackTrace();
29         }
30     }
31
32     void doEcho() throws IOException {
33         int ch = 0;
34
35         try {
36             is = s.getInputStream();
37
38         } catch(IOException e) {
39             // TODO Auto-generated catch block
40             e.printStackTrace();
41         }
42
43         try {
44             os = s.getOutputStream();
45
46         } catch(IOException
47                     e) { // TODO Auto-generated catch block
48             e.printStackTrace();
49         }
```

```
50
51          boolean flag = true;
52
53          while(flag) {
54              try {
55                  ch = br.read(); //To read users input}
56                  catch(IOException
57                              e) { // TODO Auto-generated catch block
58                      e.printStackTrace();
59                  }
60
61                  if(ch == 'x') {
62                      break;
63                  }
64
65                  try {
66                      os.write(ch);//TO write to server's console }
67                      catch(IOException
68                                  e) { // TODO Auto-generated catch block
69                          e.printStackTrace();
70                      }
71
72                      try {
73                          is.read(); //To read server's response  }
74                          catch(IOException
75                                      e) { // TODO Auto-generated catch block
76                              e.printStackTrace();
77                          }
78                      }
79                  }
80
81                  public static void main(String args[]) throws
82                      IOException {
83                      client sc = new client();
84                      sc.doEcho();
85                  }
86              }
```

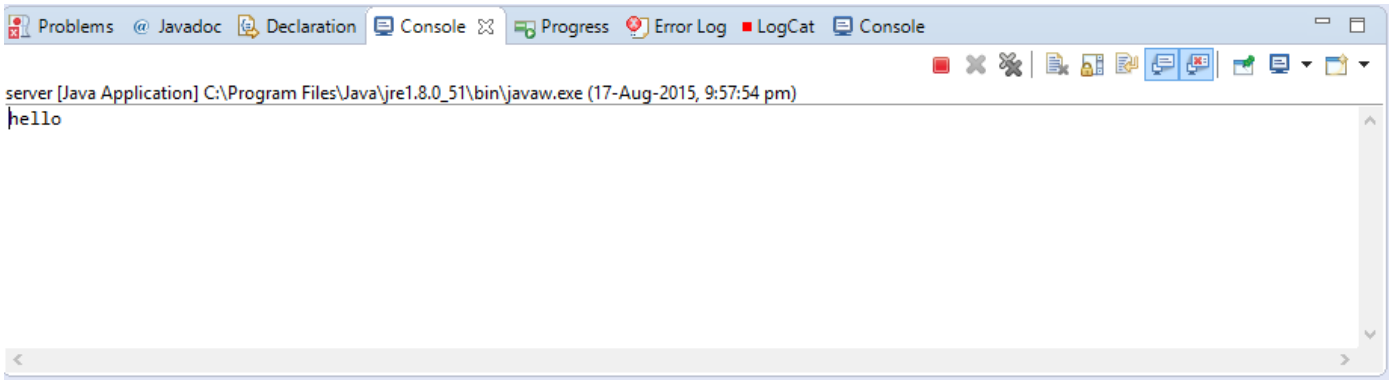


Output 4.1.1

4.2 Source of 6_2.java

```java
package connection;

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.OutputStream;
import java.net.*;
import java.lang.*;
public class server {

    Socket s;
    ServerSocket ss;
    InputStream in;
    OutputStream out;

    BufferedReader br = new BufferedReader(
    new InputStreamReader(System.in) {
        @Override
        public int read() throws
            IOException {// TODO Auto-generated method stub
            return 0;
        }
    });
    public server() {   // TODO Auto-generated constructor stub
        try {
            ss = new ServerSocket(1000);

        } catch(IOException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }
    void doEcho() throws IOException {
        int ch = 0;
        int t = 0;
        s = ss.accept();
        in = s.getInputStream();
        out = s.getOutputStream();

        while((ch = in.read()) != (-1)) {
            System.out.print((char)ch);
            out.write(ch);
            //System.out.println("");
        }

        boolean flag = true;

        while(flag) {
            try {
                t = br.read(); //To read users input}
                catch(IOException
                            e) { // TODO Auto-generated catch block
                    e.printStackTrace();
```

```
54                  }
55
56                  if(t == 'x')  {
57                      break;
58                  }
59
60                  try {
61                      out.write(t);//TO write to server's console }
62                      catch(IOException e) {
63                          // TODO Auto-generated catch block
64                          e.printStackTrace();
65                      }
66
67                      try {
68                          in.read(); //To read server's response  }
69                          catch(IOException
70                                      e) { // TODO Auto-generated catch block
71                              e.printStackTrace();
72                          }
73                      }
74                  }
75
76                  public static void main(String args[]) throws
77                      IOException {
78                      server soc = new server();
79                      soc.doEcho();
80                  }
81              }
```



Output 4.2.1

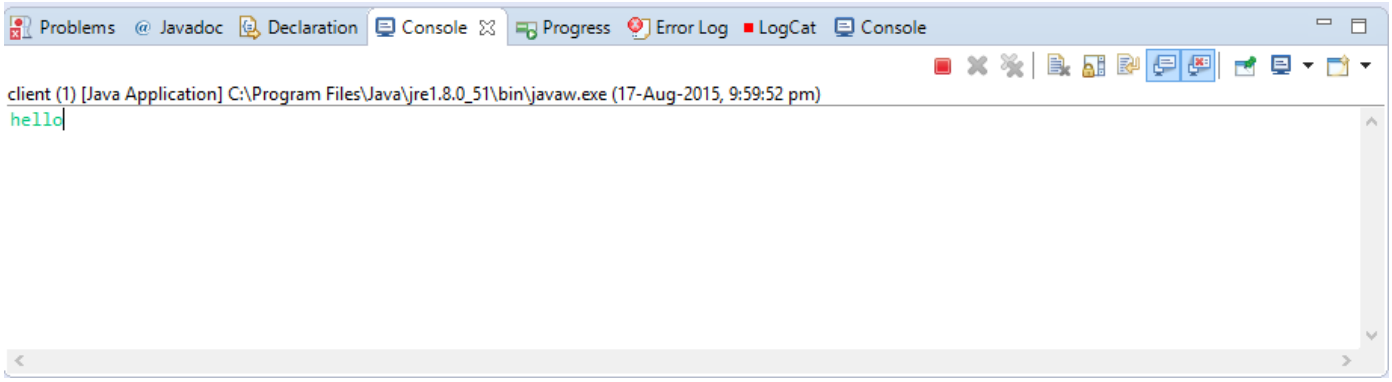
5 Write a program to implement the connectionless echo client server application

5.1 Source of 7_1.java

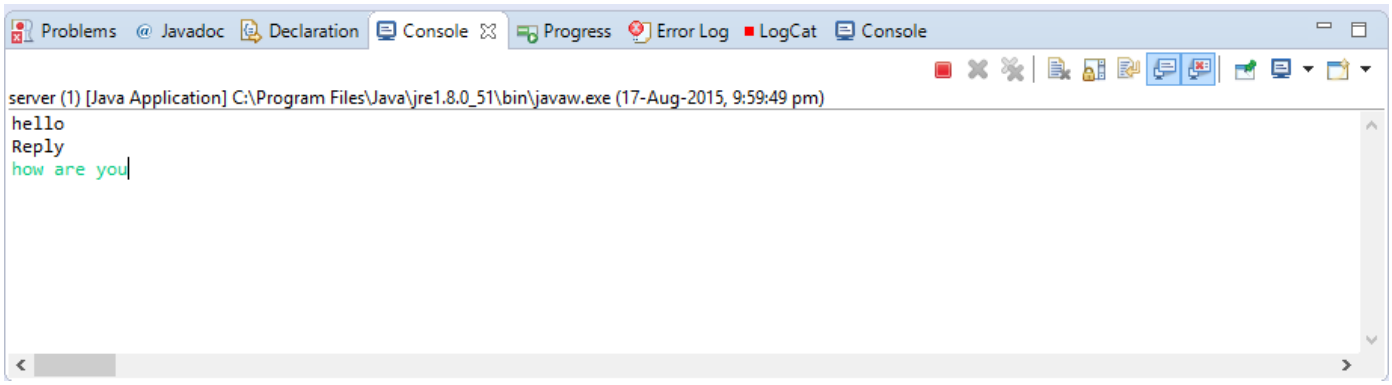
```
1  package connectionless;
2  import java.io.BufferedReader;
3  import java.io.IOException;
4  import java.io.InputStreamReader;
5  import java.net.*;
6
7  public class client {
8      public static void main(String a[]) throws
9          IOException {
10         DatagramSocket ds = new DatagramSocket();
11         byte b[] = null;
12         byte b1[] = null;
13
14         //DatagramSocket ds1=new DatagramSocket();
15         while(true) {
16             b = new byte[1024 * 2];
17             b1 = new byte[1024 * 2];
18             InetAddress ip = InetAddress.getLocalHost();
19             BufferedReader br = new BufferedReader(
20                 new InputStreamReader(System.in));
21             String str = br.readLine();
22             b = str.getBytes();
23             DatagramPacket dp = new DatagramPacket(b,
24                                                    b.length, ip, 8001);
25             ds.send(dp);
26
27             if(str.toLowerCase().contains("exit")) {
28                 System.out.println("bye....!!!!!!!");
29                 break;
30             }
31
32             DatagramPacket dp1 = new DatagramPacket(b1,
33                                                     b1.length);
34             ds.receive(dp1);
35             String str1 = new String(dp1.getData());
36             System.out.println(str1);
37             //System.out.println(str1.length());
38
39             if(str1.toLowerCase().contains("exit")) {
40                 System.out.println("bye....!!!!!!!");
41                 break;
42             }
43         }
44
45         ds.close();
46     }
47 }
```

5.2 Source of 7_1.java

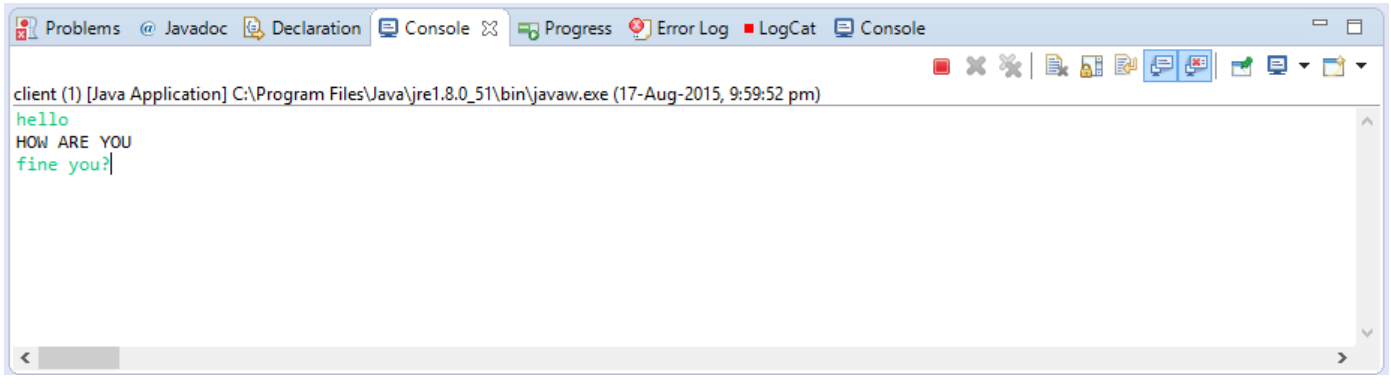
```java
package connectionless;
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.net.*;

public class client {
    public static void main(String a[]) throws
        IOException {
        DatagramSocket ds = new DatagramSocket();
        byte b[] = null;
        byte b1[] = null;

        //DatagramSocket ds1=new DatagramSocket();
        while(true) {
            b = new byte[1024 * 2];
            b1 = new byte[1024 * 2];
            InetAddress ip = InetAddress.getLocalHost();
            BufferedReader br = new BufferedReader(
                new InputStreamReader(System.in));
            String str = br.readLine();
            b = str.getBytes();
            DatagramPacket dp = new DatagramPacket(b,
                                                   b.length, ip, 8001);
            ds.send(dp);

            if(str.toLowerCase().contains("exit")) {
                System.out.println("bye....!!!!!!!");
                break;
            }

            DatagramPacket dp1 = new DatagramPacket(b1,
                                                    b1.length);
            ds.receive(dp1);
            String str1 = new String(dp1.getData());
            System.out.println(str1);
            //System.out.println(str1.length());

            if(str1.toLowerCase().contains("exit")) {
                System.out.println("bye....!!!!!!!");
                break;
            }
        }

        ds.close();
    }
}
```



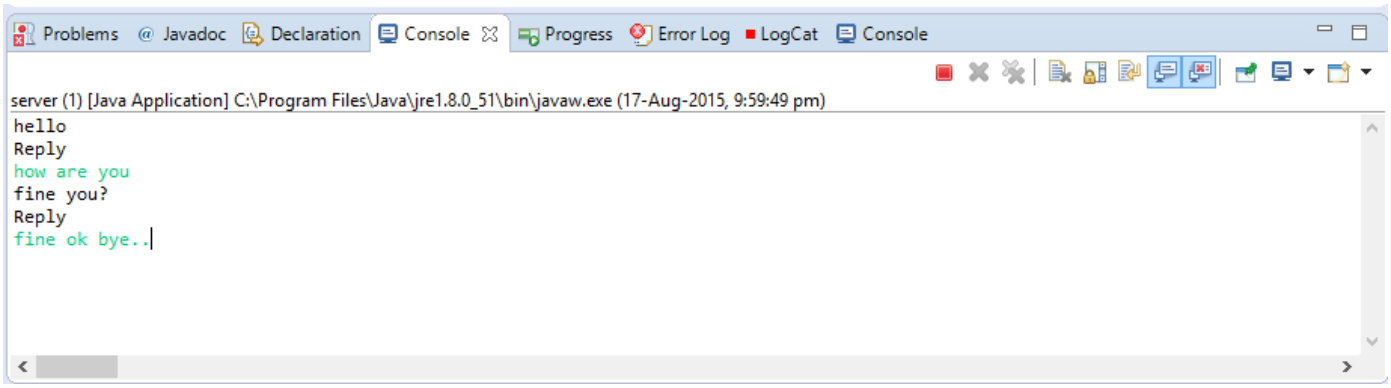
Output 5.2.1



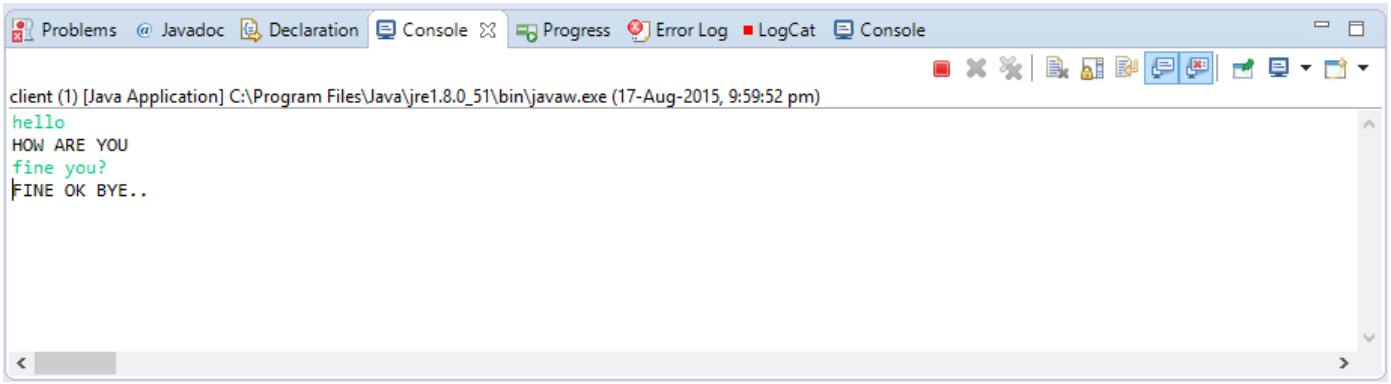
Output 5.2.2



Output 5.2.3



Output 5.2.4



Output 5.2.5

6 Remote Method Invocation Programs

6.1 Write a java RMI application which implements the simple arithmetic operations like add, multiply, subtract and divide. The client program sends two number and the arithmetic operation that wants to perform. The server performs the operation and sends back the result to the client

6.1.1 Source of 8a1.java

```
1  package rmi1;
2
3  import java.rmi.Remote;
4  import java.rmi.RemoteException;
5  public interface Calculator extends Remote {
6      public long add(long a,
7                      long b) throws RemoteException;
8      public long sub(long a,
9                      long b) throws RemoteException;
10     public long mul(long a,
11                     long b) throws RemoteException;
12     public long div(long a,
13                     long b) throws RemoteException;
14 }
```

6.1.2 Source of 8a2.java

```
1  package rmi1;
2  import java.rmi.RemoteException;
3  import java.rmi.server.UnicastRemoteObject;
4  public class CalculatorImpl extends
5      UnicastRemoteObject implements Calculator {
6
7      public CalculatorImpl() throws RemoteException {
8          super();
9      }
10     public long add(long a,
11                     long b) throws RemoteException {
12         return a + b;
13     }
14     public long sub(long a,
15                     long b) throws RemoteException {
16         return a - b;
17     }
18     public long mul(long a,
19                     long b) throws RemoteException {
20         return a * b;
21     }
22     public long div(long a,
23                     long b) throws RemoteException {
24         try {
25             return a / b;
26
27         } catch(Exception e) {
```

```
28              System.out.println(e);
29              return (-1);
30          }
31      }
32  }
```

6.1.3 Source of 8a3.java

```
1   package rmi1;
2   import java.rmi.Naming;
3   public class CalculatorServer {
4       CalculatorServer() {
5           try {
6               Calculator c = new CalculatorImpl();
7               Naming.rebind("rmi://127.0.0.1:1099/CalculatorService",
8                             c);
9
10          } catch(Exception e) {
11              e.printStackTrace();
12          }
13      }
14      public static void main(String[] args) {
15          new CalculatorServer();
16      }
17  }
```

6.1.4 Source of 8a4.java

```
1   package rmi1;
2
3   import java.rmi.Naming;
4   public class CalculatorClient {
5       public static void main(String[] args) {
6           try {
7               Calculator c = (Calculator)
8                             Naming.lookup("//127.0.0.1:1099/CalculatorService");
9               System.out.println("addition : " + c.add(10, 15));
10              System.out.println("subtraction : " + c.sub(10,
11                                 15));
12              System.out.println("multiplication : " + c.mul(10,
13                                 15));
14              System.out.println("division : " + c.div(25, 5));
15
16          } catch(Exception e) {
17              System.out.println(e);
18          }
19      }
20  }
```

```
addition : 25
subtraction : -5
multiplication : 150
division : 5
```

Output 6.1.4.1

6.2 Write an RMI application where client supplies the string arguments and server responds the number of character, number of words and number of digits in given string.

6.2.1 Source of 8b1.java

```
1  public interface WordsInterface extends
2      java.rmi.Remote {
3      int countChar(String s) throws
4          java.rmi.RemoteException;
5      int countWords(String s) throws
6          java.rmi.RemoteException;
7      int countDigits(String s) throws
8          java.rmi.RemoteException;
9  }
```

6.2.2 Source of 8b2.java

```
1  import java.rmi.Naming;
2  import java.rmi.RemoteException;
3  import java.rmi.server.UnicastRemoteObject;
4  import java.util.StringTokenizer;
5
6  public class WordsImpl extends UnicastRemoteObject
7      implements WordsInterface {
8      int totalCharacters = 0;
9      int count = 0;
10     int totalWords = 0;
11     public WordsImpl(String name) throws
12         RemoteException {
13         super();
14
15         try {
16             Naming.rebind(name, this);
17
18         } catch(Exception e) { }
19     }
20     @Override
21     public int countChar(String s) throws
22         RemoteException {
23         StringTokenizer tokens = new StringTokenizer(s,
24                 " ");
25
26         while(tokens.hasMoreTokens()) {
27             String token = tokens.nextToken();
28
29             for(int i = 0; i < s.length(); i++) {
30                 if(Character.isDigit(s.charAt(i))) {
31                     count++;
32
33                 } else {
34                     totalCharacters++;
35                 }
```

```
36              }
37
38              totalWords++;
39          }
40
41          return totalCharacters;
42      }
43      @Override
44      public int countWords(String s) throws
45          RemoteException {
46          return totalWords;
47      }
48      @Override
49      public int countDigits(String s) throws
50          RemoteException {
51          return count;
52      }
53  }
```

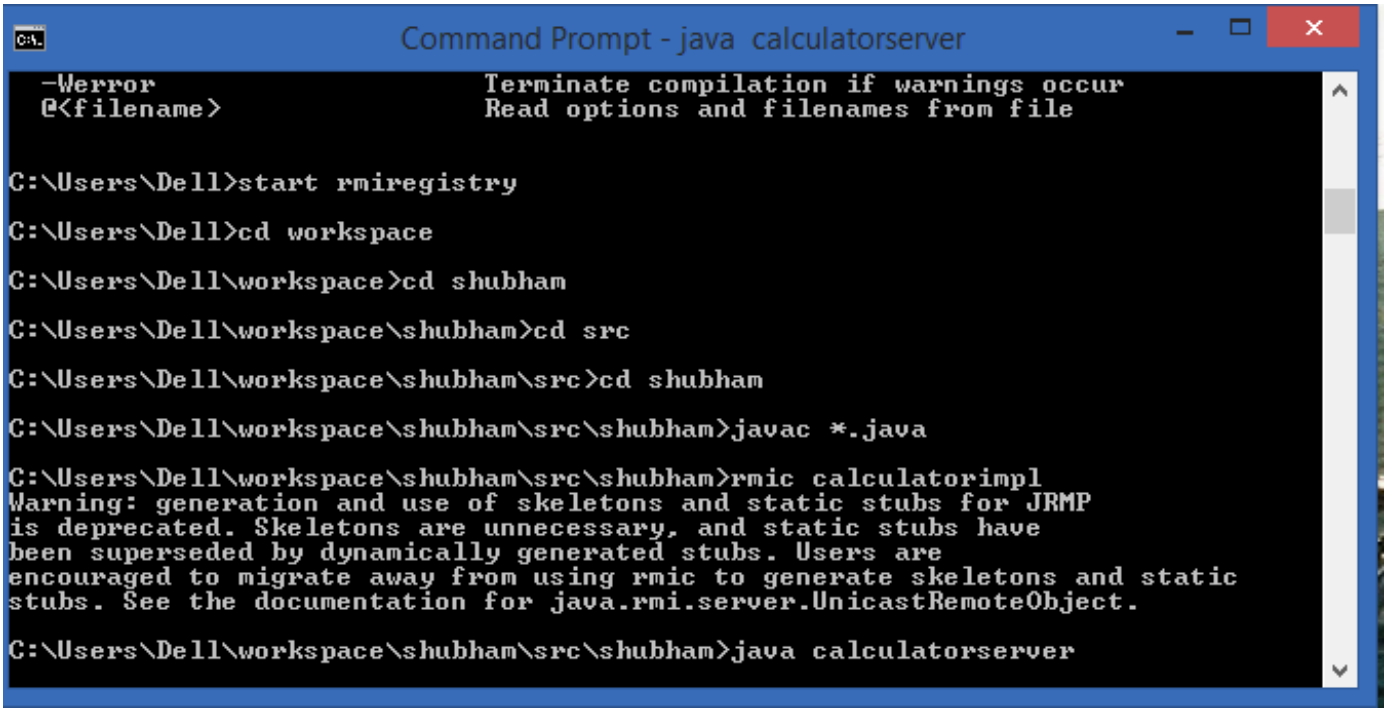
6.2.3 Source of 8b3.java

```
1   import java.rmi.*;
2   import java.rmi.registry.*;
3   import java.rmi.server.*;
4   public class WordsServer {
5       public static void main(String args[]) {
6           System.setSecurityManager(new
7                                       RMISecurityManager());
8
9           try {
10              WordsImpl pim = new WordsImpl("//localhost/pno");
11              System.out.println("\nServer is ready...");
12
13          } catch(Exception e) { }
14      }
15  }
```

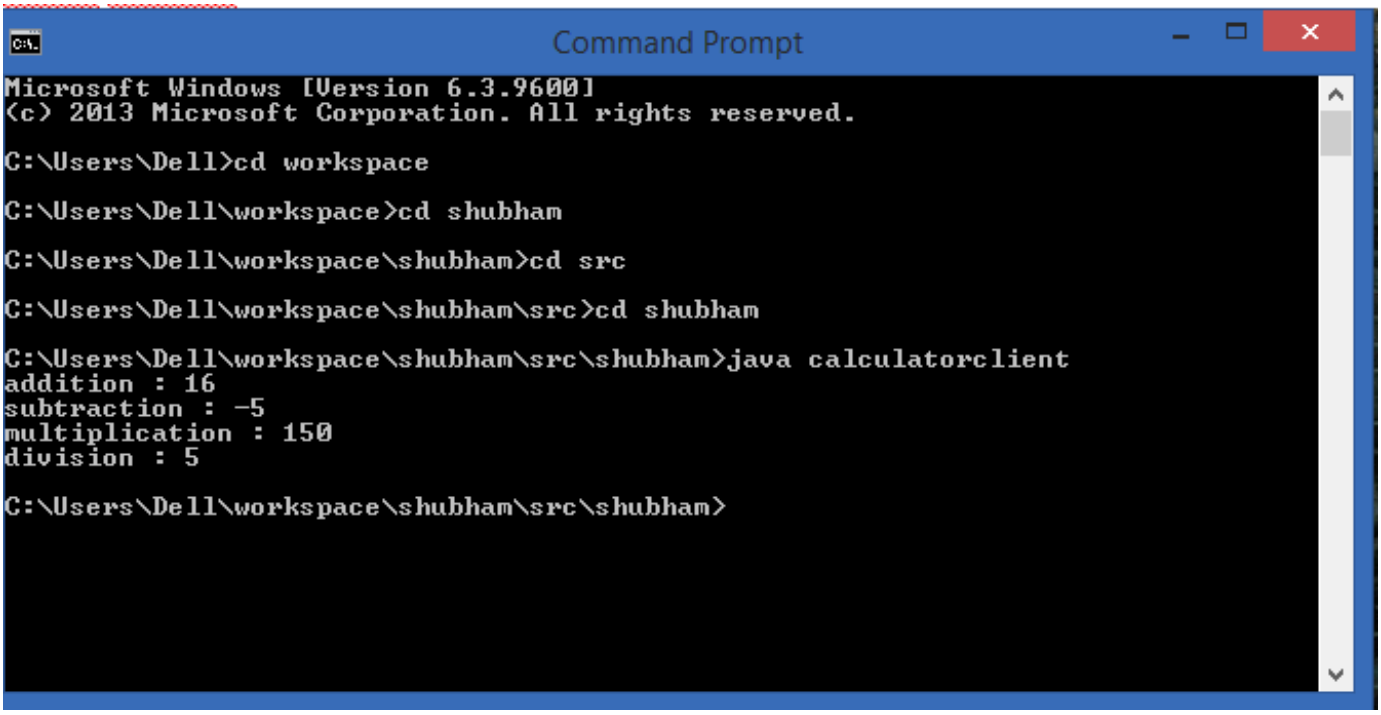
6.2.4 Source of 8b4.java

```
1   import java.rmi.*;
2   import java.rmi.registry.*;
3   import java.rmi.server.*;
4   import java.io.DataInputStream;
5   public class WordsClient {
6       public static void main(String args[]) {
7           System.setSecurityManager(new
8                                       RMISecurityManager());
9
10          try {
11              int chars, words, digits;
12              String s;
13              WordsInterface pi = (WordsInterface)
```

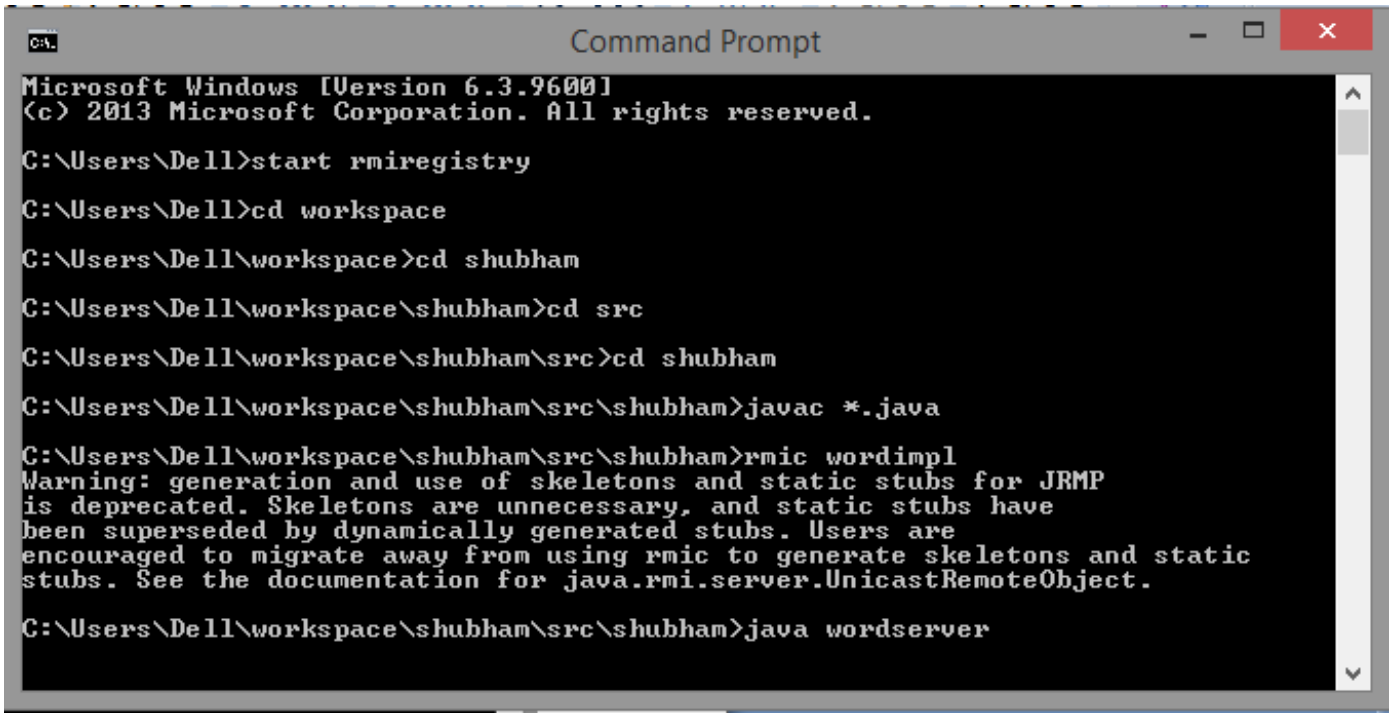
```
14                                  Naming.lookup("//localhost/pno");

15              DataInputStream in = new DataInputStream(
16                  System.in);;
17              System.out.print("Enter String and press Y to stop ");
18              char ch;
19              int i = 0;
20              s = in.readLine();
21              chars = pi.countChar(s);
22              System.out.println("characters are :" + chars);
23              digits = pi.countDigits(s);
24              System.out.println("digits are :" + digits);
25              words = pi.countWords(s);
26              System.out.println("characters are :" + words);
27
28          } catch(Exception e) { }
29      }
30  }
```



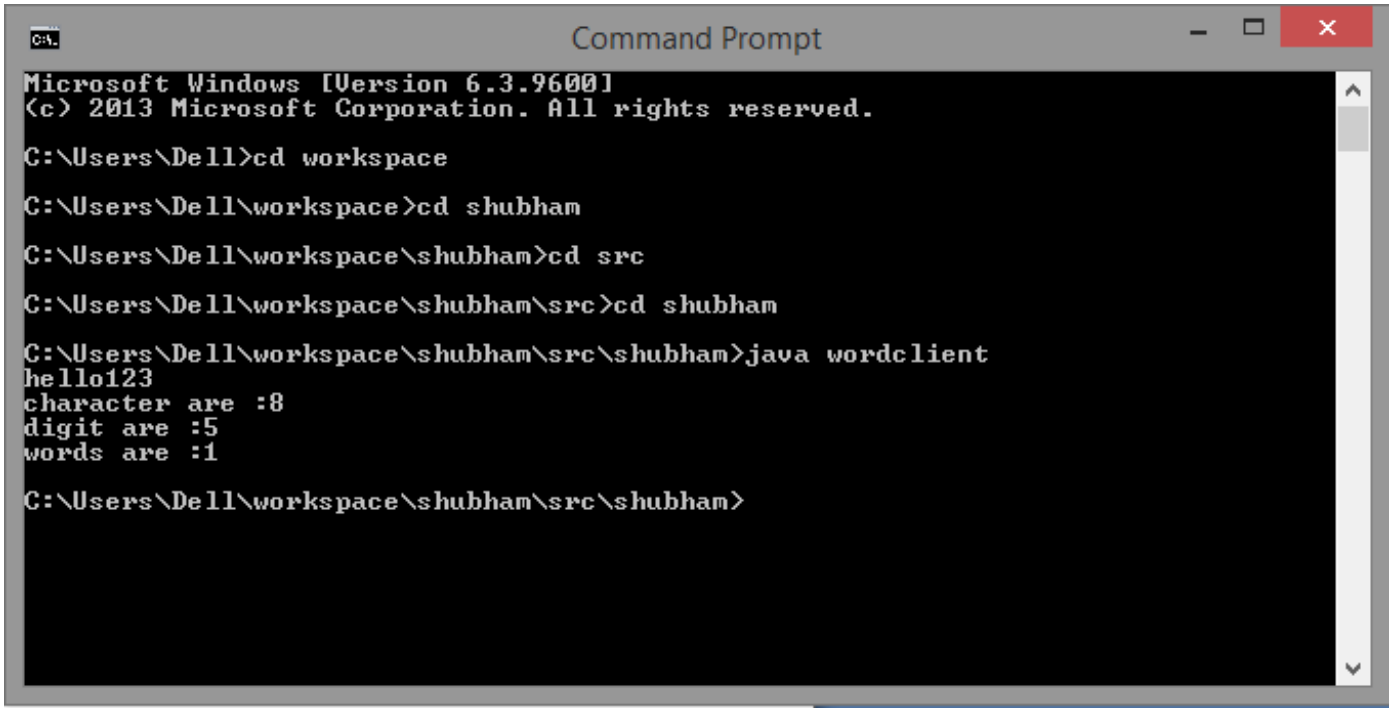
Output 6.2.4.1



Output 6.2.4.2



Output 6.2.4.3


Command Prompt
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.
C:\Users\Dell>cd workspace
C:\Users\Dell\workspace>cd shubham
C:\Users\Dell\workspace\shubham>cd src
C:\Users\Dell\workspace\shubham\src>cd shubham
C:\Users\Dell\workspace\shubham\src\shubham>java wordclient
hello123
character are :8
digit are :5
words are :1
C:\Users\Dell\workspace\shubham\src\shubham>

Output 6.2.4.4