```
class Muster extends Panel{
  // hier Referenzen fuer Komponenten
  // (Buttons, Textfields, Panels) vereinbaren
  Button OK;
  public Muster() {
    // Komponenten erzeugen und zu Oberflaeche zusammenbauen,
    // Listener verbinden
    OK=new Button("OK");
    this.add(OK);
    //addActionListener(...);}
  public static void main(String args[]){
    Frame F=new Frame();
      F.addWindowListener(new WindowAdapter(){public void
windowClosing(WindowEvent we){System.exit(0);}});
    Muster P=new Muster();
    F.add(P);
    F.pack();
    F.setVisible(true);}}
```

```
class Figur{ double getCircumference(){return 0;}
  double getArea(){ return 0;}
  int getHeight(){return 0;}
  int getWidth(){return 0;}
  public void paint(Graphics g){};}
```

```
class AWTFigur extends Panel{
  Figur f;
  AWTFigur (Figur f)
  { this.f=f; }
  public void paint(Graphics g)
  {  f.paint(g);  }
  public Dimension getPreferredSize()
  {return new Dimension (f.getWidth()+200, f.getHeight()+200);
}
  public static void main(String args[]) {
    Figur P11= new Rectangle(10,20);
    System.out.println("\nFlaeche des Rechtecks = "+P11.getArea()
+" \n");
    Frame F= new Frame();
    F.setLayout(new FlowLayout());
    F.addWindowListener(new WindowAdapter()
      {public void windowClosing(WindowEvent we)
{System.exit(0);}});
    AWTFigur P1=new AWTFigur(new Quadrat(30));
    F.add(P1);
    AWTFigur P2=new AWTFigur (new Circle(50));
    F.add(P2);
    AWTFigur P3 = new AWTFigur(new Rectangle(70,90));
    F.add(P3);
    AWTFigur P4 = new AWTFigur(new Polygon());
    F.add(P4);
    AWTFigur P5 = new AWTFigur(new String1(10,35));
    F.add(P5);
    F.pack();
    F.setVisible(true); }}
```

```
public class Rectangle extends Figur{
  public int  w; //Breite
  public int h;  //Höhe
  Rectangle (int  breite, int  hoehe){
    this.w=breite;
    this.h=hoehe; }
  public double getCircumference(){
    return 2*w + 2*h; }
  public double getArea(){
    return w*h; }
  public int getHeight(){
    return h;}
  public int getWidth(){
    // int b = (int)w;
    return w; }
  public void paint (Graphics g){
    g.setColor(Color.RED);
    g.drawRect(0,0, getWidth(), getHeight()); } }
```

```
public class String1 extends Figur{
  public int x;
  public int y;
   String1(int m, int n){
    this.x = m;
    this.y = n;  }
  double getCircumference() {return 1;}
  double getArea(){return 1;}
  int    getHeight(){return x;}
  int    getWidth() {return y;}
  public void paint (Graphics g){
    Font font = new Font("Serif", Font.ITALIC, 50);
    g.setFont(font);
    g.drawString("Hello",x,y); } }
```

```
public class Calculator1 extends Panel{
  private double result=0;
  private double m=0;
  private String memorystring;
  private double memory;
  private String lastCommand = "=";
  private boolean start = true;
  ActionListener nl = new ActionListener(){
    public void actionPerformed(ActionEvent e){
      if(start)
  tf.setText("");
      tf.setText(tf.getText()+e.getActionCommand());
      start=false; } };
  ActionListener clr = new ActionListener(){
    public void actionPerformed(ActionEvent e){
      tf.setText("");
      start=true; } };
  ActionListener mpl = new ActionListener(){
    public void actionPerformed(ActionEvent e){
      Double d = Double.parseDouble(tf.getText());
      memory=memory+d; }};
  ActionListener mr = new ActionListener(){
    public void actionPerformed(ActionEvent e){
      memorystring = String.valueOf(memory);
      tf.setText(""+memorystring);}};
  ActionListener command = new CommandAction();
  private class CommandAction implements ActionListener{
    public void actionPerformed(ActionEvent event){
      String command = event.getActionCommand();{
        calculate(Double.parseDouble(tf.getText()));
        lastCommand = command;
        start = true;}}}
  private TextField tf = new TextField(40);
  Button[] button = new Button[19];
  String[] buttonString = { "M+", "7", "8", "9", "/",
                            "M-", "4", "5", "6", "*",
                            "MR", "1", "2", "3", "-",
                            "CE", "0", ".", "=", "+"};
  ActionListener als [] = {mpl,    nl, nl , nl,      command,
                           minus,  nl, nl , nl,      command,
                           mr,     nl, nl , nl,      command,
                           clr,    nl, nl , command, command};
  Calculator1() {
    setLayout (new BorderLayout());
    add(tf,  BorderLayout.NORTH);
    Panel keys = new Panel(new GridLayout(4,5));
    //for schleife
    for(int i=0;i<buttonString.length;i++){
      Button b = new Button(buttonString[i]);
      keys.add(b);
      b.addActionListener(als[i]);}
    add(keys, BorderLayout.CENTER);}
  public void calculate(double x) {
    System.out.println("Result: "+result +" x: "+x);
    if (lastCommand.equals("+")) result += x;
    else if (lastCommand.equals("-")) result -= x;
    tf.setText("" + result);}
  public static void main(String args[])
  {Muster.java}}
```

```
  private TextField tf = new TextField (50);
  private static final String [][] buttonString =
{ {"M+","7","8","9","/" },
                    {  "M-","4","5","6","*" },
                    {  "MR","1","2","3","-" },
                    {  "CE","0",".","=","+" }};
  ActionListener als [][] =  { {mpl,   nl, nl, nl,      command},
                    {minus,   nl, nl, nl,      command},
                    {mr,      nl, nl, nl,      command},
                    {clr,     nl, nl, command,  command}};
  CalcGridBag(){
    int rows = buttonString.length;
    int cols = buttonString[0].length;
    setLayout(new GridBagLayout());
    C.weightx = 1.0; C.weighty = 1.0;
    C.fill = GridBagConstraints.BOTH;
    C.anchor = GridBagConstraints.WEST;
    C.gridx = 0; C.gridy = 0;
    add(l, C);
    C.gridx = 1;
    C.gridwidth = GridBagConstraints.REMAINDER;
    add(tf, C);

    C.gridwidth = 1;
    //C.gridheight = 1;
    C.gridy = 4;
    C.gridx = GridBagConstraints.RELATIVE;
    for(int i=0; i < rows; i++){
      for(int j=0; j < cols; j++){
        //C.gridx = GridBagContraints.RELATIVE;
        Button b = new Button(buttonString[i][j]);
        add(b, C);
        b.addActionListener(als[i][j]);
      } C.gridy = C.gridy + 1;  //spaltenweise überspringen}}
  public static void main (String args[]){
    Frame F = new Frame();
    F.addWindowListener...
    CalcGridBag P = new CalcGridBag();

    F.setLayout(new BorderLayout());
    F.add(P, BorderLayout.CENTER);

    F.add(P);
    F.pack();
    F.setVisible(true); }}
```

```
import java.lang.*;
import java.io.*;
import java.awt.*;
import java.awt.event.*;
  HexDumpArea(int rows, int cols)
  {super(rows, cols);}
  public String getHexString(){
    String s="";
    //Ausgabe von i in Hexadezimal
    for (int i=0; i < data.length; i+=16){
      s+= '\n'+new String(hexByte(i,4))+": ";
      //Ausgabe des HexBytes mit '|' nach jeder 4 Zeilen
      for (int j=0; j<16 && (i+j)< data.length; j++) {
        s+= new String(hexByte(data[i+j],2))+' ';
        if(((i+j+1)%4)==0)
          s+=" | "; }
      //Ausgabe des Textes args[0]
      for(int j =0; j<16 && (i+j)<data.length; j++){
        s+=(char)data[i+j]; }}
    return s; }
  public void replace(String s){
    data = s.getBytes();
    setText(getHexString());}
```

```
public class HexDumpPanel extends Panel{
  private TextField tf = new TextField(70);
  HexDumpArea area     = new HexDumpArea(20,50);

  ActionListener url1 = new ActionListener(){
    public void actionPerformed(ActionEvent e){
      String eingabe = tf.getText();
      URL url = null;
      try{url = new URL(eingabe);
      }catch(MalformedURLException me){}
      String buf, ausgabe="";
      InputStreamReader i = null;
      BufferedReader BR = null;
      try{
        i= new InputStreamReader(url.openStream());
        BR = new BufferedReader(i);
        while((buf=BR.readLine()) != null){
          System.out.println(buf);
          ausgabe = ausgabe + buf;}
        area.replace(ausgabe);
        BR.close();
      }catch(IOException ioe){}}};

  ActionListener c = new ActionListener(){
    public void actionPerformed(ActionEvent e){
      String eingabe = tf.getText();
      URL url = null;
      URLConnection c = null;
      try{
        url = new URL(eingabe);
        try{c = url.openConnection();
        }catch(IOException ioe){}
        try{ c.connect();
        }catch(IOException ioe){}
      }catch(MalformedURLException me){}
      String buf, ausgabe="";
      InputStreamReader i = null;
      BufferedReader BR = null;
      try{ i = new InputStreamReader(url.openStream());
        BR = new BufferedReader(i);
        while((buf = BR.readLine()) != null){
          System.out.println(buf);
          ausgabe = ausgabe + buf;}
        area.replace(ausgabe);
        BR.close();
      }catch(IOException ioe){System.out.println(ioe);
        System.exit(1); } } };

  ActionListener asock = new ActionListener(){
    public void actionPerformed(ActionEvent e){
      String eingabe = tf.getText();
      URL url = null;
      try{
        url = new URL(eingabe);
      }catch(MalformedURLException me){}
      try{
        Socket s = new Socket(url.getHost(), 80);
        PrintStream O = new PrintStream(s.getOutputStream());
        O.println("GET"+url.getPath());
        O.println("\r");
        BufferedReader I = new BufferedReader(new
InputStreamReader(s.getInputStream()));
        String X = "", buf;
        while((buf = I.readLine()) != null)
          X = X + buf;
          area.replace(X);
        s.close();
      }catch(Exception ee){System.out.println(ee);
      ee.printStackTrace();} } };

  Button [] button = new Button[3];
  private String [] buttonString = {"Get URL-Object","Get URL-
Connection","Get Socket"};
  ActionListener als [] = {url1, c, asock};

  public HexDumpPanel(){
    Panel P = new Panel();
    Label l = new Label("Enter File");
    P.add(l);
    P.add(tf);
    Panel keys = new Panel();
    for(int i = 0; i < buttonString.length; i++){
      Button b = new Button(buttonString[i]);
      keys.add(b);
      b.addActionListener(als[i]);}
    setLayout(new BorderLayout());
    add(P, BorderLayout.NORTH);
    add(area, BorderLayout.CENTER);
    add(keys, BorderLayout.SOUTH);}
  public static void main (String [] args){Muster.java}}
```

```
public class ImgPanel extends Panel{
  private Image Img;
  float [] prozentzahl = {0,0,0,0};
  float [] winkel = {0,0,0,0};
  float [] h = {0,0,0,0};
  int n = 4;
  public ImgPanel(Image Img, String args[]){
    arc_angle(args);
    this.Img = Img;
    MediaTracker M = new MediaTracker(this);
    M.addImage(Img, 1);
    try{
      M.waitForID(1);
    }catch(Exception e){}}
  public void arc_angle(String args []){
  for(int i = 0; i < n; i++){
  prozentzahl[i] = Float.parseFloat(args[i]);
      winkel[i] = prozentzahl[i]*360/100;
      h[i] = prozentzahl[i]*10;}}
  @Override
  public void paint (Graphics g){
    g.drawImage(Img, 0, 0, this);
  Color  [] col  = {new Color (200,0 ,0  ,127),
                    new Color (76,175,80 ,127),
                    new Color (200,10,140,127),
                    new Color (0  ,0 ,200,127)};
  String [] name = {"Atom", "Kohle", "Gas", "Sonne/Wind"};
  int  angle = 0;
    for(int i = 0; i < n; i++){
    g.setColor(col[i]);
    g.fillArc(Img.getWidth(this)/2-550, Img.getHeight(this)/2-
300/2, 400, 400, angle, (int)winkel[i]);
    g.fillRect(600+i*150, Img.getHeight(this)/n + (450-(int)h[i]),
100, (int)h[i]);   //Img.getWidth(this)/n
      g.drawString(name[i]+": "+prozentzahl[i]+"%", 420,
80+i*20 );
    angle += (int)winkel[i];}}
    @Override
  public Dimension getPreferredSize(){
    return new Dimension(Img.getWidth(this),
Img.getHeight(this)); }
  public static void main(String args []){
    Frame F = new Frame("Diagram");
    Image Img = F.getToolkit().getImage(args[4]);
    F.addWindowListener...
    ImgPanel P = new ImgPanel(Img, args);
    F.add(P);
    F.pack();
    F.setVisible(true);} }
```

```
public class Client {
  public static void main (String[] args){
    DatagramSocket skt=null;
    try{ skt = new DatagramSocket();
      String msg = args[0];
      //konvertieren unsere Nachricht
      byte [] b = msg.getBytes();
      InetAddress host = InetAddress.getByName("localhost");
      int serverSocket = 6755;
      //datagrammpacket zum Senden
      DatagramPacket request = new
DatagramPacket(b,b.length,host,serverSocket);
      //wir benutzen skt um Daten zu senden
      skt.send(request);
      //um antwort zu bekommen
      byte [] buffer = new byte[1000];
      DatagramPacket reply = new DatagramPacket(buffer,
buffer.length);
      //jetzt bekommen wir die antwort
      skt.receive(reply);
      // Paket auspacken
      msg=new String(reply.getData(),0,reply.getLength());
      //antwort auf die konsole ausgeben
      System.out.println(msg);
      skt.close();
    }catch (Exception ex){}}}
```

```
public class Server {
  public static void main (String[] args){
    DatagramSocket skt=null;
    try{skt = new DatagramSocket(6755);
      byte [] buffer = new byte[1000];
      while(true){
        DatagramPacket request = new DatagramPacket(buffer,
buffer.length);
        skt.receive(request);
        String msg = (new
String(request.getData(),0,request.getLength()));
        File file = new File(msg);
        if (file.exists()) {
          msg+=" existiert und hat die Länge "+ file.length();
          // ask for new file name }
        else msg+=" existiert nicht!";
        byte [] sendMsg = msg.getBytes();
        DatagramPacket reply = new DatagramPacket (sendMsg,
sendMsg.length, request.getAddress(), request.getPort());
        skt.send(reply);
        skt.close();}
    }catch (Exception ex){} }}
```

```
import java.io.*;
public class Materialartikel{
  private String artikelname;
  private int    mindeststueckzahl;
  private int    bestellmenge;
  private int    lagerbestand;
  private int    entmenge;
  Materialartikel (String artikelname,//Konstruktor
  int    mindeststueckzahl,
  int    bestellmenge,
  int    lagerbestand,
  int    entmenge,){
    this.artikelname       =       artikelname;
    this.mindeststueckzahl  =  mindeststueckzahl;
    this.bestellmenge      =      bestellmenge;
    this.lagerbestand      =      lagerbestand;
    this.entmenge          =          entmenge; }
  public String getArtikelname(){
    return artikelname;}
  public int getMindeststueckzahl(){
    return mindeststueckzahl; }
  public int getBestellmenge(){
    return bestellmenge;}
  public int getLagerbestand(){
    return lagerbestand;}
  public int getEntmenge(){
    return entmenge;}
    public void entnahme(){
    lagerbestand = lagerbestand - entmenge;}
  public void nachbestellung(){
    lagerbestand = lagerbestand + bestellmenge;
    lagerbestand = lagerbestand; }
  public void setLagerbestand(int bestand){
    lagerbestand = bestand;}
  public void setBestellmenge(int nachbestellmenge){
    bestellmenge = nachbestellmenge;}
  public void setEntmenge(int e_menge){
    entmenge = e_menge; }
  public void setArtikelname(String aname){
    artikelname = aname;}
  public void SetMindeststueckzahl(int zahl){
    mindeststueckzahl = zahl;}
  public String toString(){
  return("  Artikelname: "       +artikelname
    +"\nMindeststueckzahl: " +mindeststueckzahl
    +"\nNachbestellmenge: "  +bestellmenge
    +"\nLagerbestand: "      +lagerbestand); }}
```

```
import java.io.*;
public class Lagerhaltung{
  public static String einlesen(){
    String str = "";   //String auf 0 setzen
    try{  InputStreamReader isr  =  new
InputStreamReader(System.in);
      BufferedReader     bur  =  new BufferedReader(isr);
      //Hier lesen wir einen String ein
      str  =  bur.readLine();
      //und geben ihn gleich wieder aus
      //System.out.println(str);
    }catch(Exception e){}
    return str;}
  public static void main (String [] args) {
/* Materialartikel artikel1 = new Materialartikel(args[0],
    Integer.parseInt(args[4]),
    Integer.parseInt(args[5]),
    Integer.parseInt(args[6]),*/
    System.out.println( "Artikel einfügen: ");
    System.out.println("\nArtikelname: "); String artikelname =
einlesen();
    while(artikelname.equals("")){
      System.out.println("\nBitte geben Sie den Artikelnamen ein.");
      artikelname = einlesen();}
    System.out.println("\nMindeststueckzahl: "); String
mindeststueckzahl = einlesen();
    while(mindeststueckzahl.equals("")  ||
Integer.parseInt(mindeststueckzahl)<0){
      System.out.println("\nFalsche Eingabe. Versuchen Sie es noch
mal.");
      mindeststueckzahl  = einlesen();}
    System.out.println("\nLagerbestand: ");
    Materialartikel  artikel  = new Materialartikel(
    artikelname,
    Integer.parseInt(mindeststueckzahl),
    0,
    Integer.parseInt(lagerbestand),
    0);
    while(true){
      System.out.println("\nAuswählen \n0 - Anzeigen \n1 -
Lagerbestand \n2 - Materialentnahme \n3 - Nachbestellung \n4 -
Verlassen");
      String str = einlesen();
      if(str.equals("4")) break;
      switch(str){
        case "0":
          System.out.println(artikel.toString()); break;
        case "1":
          System.out.println("Lagerbestand ist: " +
artikel.getLagerbestand());
          break;
        case "2":
          System.out.println("Wie viel möchten Sie aus dem Lager
entnehmen?");
          String e_menge = einlesen();
          while(e_menge.equals("")   ||
Integer.parseInt(e_menge)<0){
            System.out.println("Falsche Eingabe. Versuchen Sie es
nochmal.");
            e_menge = einlesen();}
          artikel.setEntmenge(Integer.parseInt(e_menge));
          if(artikel.getLagerbestand() < artikel.getEntmenge()){
            System.out.println("\n Lagerbestand überschritten.
Möchten Sie nachbestellen?(ja/nein)");
            String antwort = einlesen();

            if(antwort.equals("ja")){
              int nachbestellmenge = artikel.getEntmenge() -
artikel.getLagerbestand() + artikel.getMindeststueckzahl();
              artikel.setBestellmenge(nachbestellmenge);   }
            else System.out.println("\nMaterialentnahme
abgebrochen");}
          else if ((artikel.getLagerbestand() -
artikel.getEntmenge())>= artikel.getMindeststueckzahl()) {
            artikel.entnahme();}
          else { System.out.println("\nMindesstueckzahl
unterschritten. Möchten Sie nachbestellen?(ja/nein)");
            String antwort = einlesen();
            if(antwort.equals("ja")){
              artikel.entnahme();
              int nachbestellmenge = artikel.getMindeststueckzahl() -
artikel.getLagerbestand();
              artikel.setBestellmenge(nachbestellmenge); }
            else System.out.println("\nMaterialentnahme
abgebrochen.");  }
          break;
        case "3":
          System.out.println("Wieviel möchten Sie nachbestellen?");
String menge = einlesen();
          artikel.setBestellmenge(Integer.parseInt(menge));
          artikel.nachbestellung();
          break;
        default: System.out.println("...");}}}}
```

```
class UDPClient {
  public static void main(String [] args)throws IOException,
InterruptedException{
    //Liefert Internet Adresse
    InetAddress addr = InetAddress.getByName("localhost");
    while(true){
      String s = new Date().toString();
      byte [] raw = s.getBytes();
      //Packet bereitstellen  Das Datenpacket
      DatagramPacket packet = new DatagramPacket(raw,
raw.length, addr, 4711);
      //Der SendeSocket
      DatagramSocket dSocket = new DatagramSocket();
      dSocket.send(packet);
      System.out.println("Weg is' es");
      Thread.sleep(1000);}}}
```

```
public class UDPServer{
  public static void main (String [] args)throws IOException{
    DatagramSocket socket = new DatagramSocket();
    while(true){
      //Auf Anfrage warten
      //Leeres Datagram bereitstellen
      DatagramPacket packet= new DatagramPacket(new
byte[1024], 1024);
      socket.receive(packet);
      //Empfänger auslesen
      InetAddress address = packet.getAddress();
      int        port    = packet.getPort();
      int        len     = packet.getLength();
      byte []    data    = packet.getData();
      System.out.printf("Anfrage von %s vom Port %d mit der
Länge %d:%n%s%n",
      address, port, len, new String(data, 0, len));}}}
```