1) Błądzenie przypadkowe (d=2) pojedynczej cząstki na sieci z periodycznymi warunkami brzegowymi. Do apletu dodawany jest obiekt klasy Wykres dziedziczącej po klasie JPanel. Metoda paintComponent w klasie Wykres rekursywnie wywołuje samą siebie (repaint()). W metodzie paintComponent wprowadzono uśpienie na 10ms (Thread.sleep(20)).

```
import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class particle_pbc extends JApplet implements ActionListener {
       int applet_width=200;
       int applet_height=200;
       Wykres wykr = new Wykres();
        public void init(){
               setSize(applet_width,applet_height);
               JButton przycisk = new JButton("START");
              przycisk.addActionListener(this);
              add(wykr,BorderLayout.CENTER);
              add(przycisk,BorderLayout.SOUTH);

        }
        public void actionPerformed(ActionEvent e){
                  wykr.rysowac=true;
              wykr.repaint();
           }

}

class Wykres extends JPanel{
       boolean rysowac;
       int wielkosc=50;
       int [][] siec = new int [wielkosc][wielkosc];
       int x,y;
       public Wykres(){setSize(wielkosc,wielkosc);
                       rysowac=false;
                       for (int i=0;i<wielkosc;i++){
                          for (int j=0;j<wielkosc;j++){
                                 siec[i][j]=0;
                       }
                       }
                       x=wielkosc/2;
                       y=wielkosc/2;
                       }
       public Dimension getPreferredSize()
       {return new Dimension(2*wielkosc,2*wielkosc);
       }

       public void paintComponent(Graphics graf){
              super.paintComponent(graf);
              graf.drawLine(0, 0, 0, wielkosc);
              graf.drawLine(0, 0, wielkosc,0);
              graf.drawLine(0, wielkosc, wielkosc, wielkosc);
              graf.drawLine(wielkosc, 0,wielkosc, wielkosc);
              if(rysowac){
                     siec[x][y]=0;
                     double losowa=Math.random();
                     if(losowa<0.25) {x=x+1;if (x==wielkosc){x=0;}}
                     else if(losowa<0.5) {x=x-1;if (x==-1){x=wielkosc-1;}}
                     else if(losowa<0.75) {y=y+1;if (y==wielkosc){y=0;}}
```

```
                    else {y=y-1;if (y==-1){y=wielkosc-1;}}
                    siec[x][y]=1;

                for (int i=0;i<wielkosc;i++){
              for (int j=0;j<wielkosc;j++){
                    if(siec[i][j]==1){graf.drawRect(i,j,1,1);}
             }
             }
               try {
                   Thread.sleep(10);                // sleep for 10 msec
                } catch (InterruptedException t){}
              repaint();
              }
        }
}
```

2) Trzy oddziałujące cząstki – cząstka nie może przemieścić sią na miejsce już zajęte przez inną cząstkę. Zdefinuj metodę step(int ktory): cząstka o indeksie ktory przemieszcza się w losowo wybranym kierunku (o ile jest to wykonalne).

```
import java.applet.Applet;
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class czastki_3 extends JApplet implements ActionListener {
       int applet_width=300;
       int applet_height=300;
       Wykres1 wykr = new Wykres1();
        public void init(){
              setSize(applet_width,applet_height);
              JButton przycisk = new JButton("START");
             przycisk.addActionListener(this);
             setLayout(new FlowLayout());
             add(wykr);
             add(przycisk);

        }
        public void actionPerformed(ActionEvent e){
                  wykr.czystart=true;
             wykr.repaint();
          }

}

class Wykres1 extends JPanel{
       boolean czystart;
       int wielkosc=200;
       int [][] siec = new int [wielkosc][wielkosc];
       int lczastek=3;
       int [][] walker = new int [lczastek][2];
       public Wykres1(){
             setBorder(BorderFactory.createLineBorder(Color.blue));
             czystart=false;
                       for (int i=0;i<wielkosc;i++){
                          for (int j=0;j<wielkosc;j++){
```

```
                                    siec[i][j]=0;
                        }
                        }
                        walker[0][0]=wielkosc/4;
                             walker[0][1]=wielkosc/4;
                       siec[wielkosc/4][wielkosc/4]=1;
                        walker[1][0]=wielkosc/2;
                          walker[1][1]=wielkosc/2;
                          siec[wielkosc/2][wielkosc/2]=1;
                       walker[2][0]=wielkosc/4;
                          walker[2][1]=wielkosc/2;
                          siec[wielkosc/4][wielkosc/2]=1;
                        }
        public Dimension getPreferredSize()
        {return new Dimension(wielkosc,wielkosc);
        }

       void step (int ktory){

       }
        public void paintComponent(Graphics graf){
              super.paintComponent(graf);
              graf.drawLine(0, 0, 0, wielkosc);
              graf.drawLine(0, 0, wielkosc,0);
              graf.drawLine(0, wielkosc, wielkosc, wielkosc);
              graf.drawLine(wielkosc, 0,wielkosc, wielkosc);
              for (int i=0;i<wielkosc;i++){
              for (int j=0;j<wielkosc;j++){
                    if(siec[i][j]==1){graf.drawRect(i,j,1,1);}
          }
          }
              if(czystart){
                    for (int czastka=0;czastka<lczastek;czastka++){
                          int indeks=(int)(lczastek*Math.random());
                          step(indeks);
                    }

                 try {
                     Thread.sleep(10);                // sleep for 10 msec
                   } catch (InterruptedException t){}
              repaint();
              }
        }
}
```

3) Konfiguracja początkowa zawiera 10000 cząstek ułożonych w kwadracie 100x100

```
import java.applet.Applet;
```

```
import java.awt.*;
import java.awt.event.*;
import javax.swing.*;
public class jparticles extends JApplet implements ActionListener {
       int applet_width=800;
       int applet_height=800;
       Wykres2 wykr = new Wykres2();
        public void init(){
              setSize(applet_width,applet_height);
              JButton przycisk = new JButton("START");
             przycisk.addActionListener(this);
             add(wykr,BorderLayout.CENTER);
             add(przycisk,BorderLayout.SOUTH);

        }
        public void actionPerformed(ActionEvent e){
                 wykr.rysowac=true;
             wykr.repaint();
          }

}

class Wykres2 extends JPanel{
       boolean rysowac;
       int wielkosc=600;
       int [][] siec = new int [wielkosc][wielkosc];
       int lsquare=100;
       int lczastek=lsquare*lsquare;
       int [][] walker = new int [lczastek][2];
       public Wykres2(){rysowac=false;
                       for (int i=0;i<wielkosc;i++){
                          for (int j=0;j<wielkosc;j++){
                                 siec[i][j]=0;
                       }
                       }

                       for (int i=0;i<lsquare;i++){
                          for (int j=0;j<lsquare;j++){
                    siec[i+(wielkosc-lsquare)/2][j+(wielkosc-lsquare)/2]=1;
                    walker[i*lsquare+j][0]=i+(wielkosc-lsquare)/2;
                    walker[i*lsquare+j][1]=j+(wielkosc-lsquare)/2;
                       }
                       }



                       }
       public Dimension getPreferredSize()
       {return new Dimension(wielkosc,wielkosc);
       }
      void step (int ktory){
        int x=walker[ktory][0];
        int y=walker[ktory][1];
                    double losowa=Math.random();
                    if(losowa<0.25) {x=x+1;if (x==wielkosc){x=0;}}
                    else if(losowa<0.5) {x=x-1;if (x==-1){x=wielkosc-1;}}
                    else if(losowa<0.75) {y=y+1;if (y==wielkosc){y=0;}}
                    else {y=y-1;if (y==-1){y=wielkosc-1;}}
            if(siec[x][y]==0){
```

```
                siec[walker[ktory][0]][walker[ktory][1]]=0;
                walker[ktory][0]=x;
                walker[ktory][1]=y;
                siec[x][y]=1;
               }
        }
        public void paintComponent(Graphics graf){
                super.paintComponent(graf);
                graf.drawLine(0, 0, 0, wielkosc);
                graf.drawLine(0, 0, wielkosc,0);
                graf.drawLine(0, wielkosc, wielkosc, wielkosc);
                graf.drawLine(wielkosc, 0,wielkosc, wielkosc);
                if(rysowac){
                        for (int czastka=0;czastka<lczastek;czastka++){
                                int indeks=(int)(lczastek*Math.random());
                                step(indeks);
                        }
                  for (int i=0;i<wielkosc;i++){
                for (int j=0;j<wielkosc;j++){
                        if(siec[i][j]==1){graf.drawRect(i,j,1,1);}
               }
               }
                  try {
                      Thread.sleep(20);                 // sleep for 10 msec
                    } catch (InterruptedException t){}
                repaint();
                }
        }
}
```

a) Wprowadź asymetryczne błądzenie (faworyzujące pewien kierunek).

b) Wprowadź przeszkodę, którą cząstki będą zmuszone omijać.

c) Zmodyfikuj program dodając Wykres2 i przycisk START do obiektu JFrame:

```
import java.awt.*;
import javax.swing.*;
class GuiSwing extends JFrame {

  // uzyskujemy contentPane okna
  Container cp = getContentPane();

GuiSwing() {
     // ustalenie tytułu okna
     super("Okno aplikacji");
     // ustalenie rozkładu; jeśli trzeba np:
     cp.setLayout(new FlowLayout());
     // tworzenie komponentów np.
     JLabel lab = new JLabel("Etykieta");
     JButton b  = new JButton("Przycisk");
     // Ustalenie własciwości komponentów,
     // np:
     lab.setForeground(Color.red);
     b.setForeground(Color.blue);
     // Dodanie komponentów do okna np.
     cp.add(lab);
     cp.add(b);
     // ustalenie rozmiarów okna, np.:
```

```
     pack();
    // pokazanie okna (domyślnie niewidoczne)
     setVisible(true);
}

public static void main(String[] args) {
   // w metodzie main tworzymy
   //obiekt naszej klasy
   // i wywołujemy konstruktor
    new GuiSwing();
}
} // koniec klasy GuiSwing
```