Uzupełnij definicję metody addborder(int i,int j)**, która jeżeli trzeba dodaje węzeł [i][j] do tablicy** boder.

```
import java.awt.*;
import java.applet.*;
import java.awt.event.*;
import javax.swing.*;
public class eden_swing extends JApplet implements ActionListener {

    int size=600;   // applet size

    JTextField tfield;
    Wykres_eden wykr_eden = new Wykres_eden();
      public void init(){
            resize(size,size);
//          setBackground(Color.yellow);
            add(wykr_eden,BorderLayout.CENTER);
            add(new JLabel("number of steps"),BorderLayout.NORTH);
            tfield = new JTextField(10);
            add(tfield,BorderLayout.SOUTH);
            tfield.addActionListener(this);
      }  // end of init
       public void actionPerformed(ActionEvent e){
             wykr_eden.n=Integer.parseInt(tfield.getText());
             wykr_eden.repaint();
       }   // end of actionPerformed
}
      class Wykres_eden extends JPanel{
             int n;  // liczba kroków
             int square_size=5;
                int lsize=100;
                int wykres_size=square_size*lsize;
                int [][] lattice = new int [lsize][lsize];
                // lattice=0 virgin
                // lattice=1 ocuppied
                // lattice=2 border
                int lborder;
                int [][] border = new int [lsize*lsize/10][2];
                int step;
      public Wykres_eden(){
             // konfiguracja początkowa
             for (int i=0;i<lsize;i++){
                         for (int j=0;j<lsize;j++){
                               lattice[i][j]=0;
                         }}
                int k=lsize/2;
                 lborder=0;
                 border[0][0]=k;
                 border[0][1]=k;
                 lattice[k][k]=2;
                 step=0;
                setBorder(BorderFactory.createLineBorder(Color.orange));
      }

      public Dimension getPreferredSize()
      {return new Dimension(wykres_size,wykres_size);
      }
      public void paintComponent(Graphics g){
            super.paintComponent(g);
```

```
            for (int i=0;i<lsize;i++){
                  for (int j=0;j<lsize;j++){
                        if(lattice[i][j]==1) {g.setColor(Color.blue);}
                        if(lattice[i][j]==2) {g.setColor(Color.black);}
                        if(lattice[i][j]>0) g.fillRect(square_size*i,
square_size*j, square_size, square_size);
                  }}
            if ((n>0)&&(step<n)){
                  step++;
                        int k=(int)(Math.random()*(lborder+1));
                        int i = border[k][0];
                        int j = border[k][1];
                        border[k][0]=border[lborder][0];
                        border[k][1]=border[lborder][1];
                        lborder=lborder-1;
                        lattice[i][j]=1;
                        addborder(i,j+1);
                        addborder(i,j-1);
                        addborder(i+1,j);
                        addborder(i-1,j);
                  try {
                      Thread.sleep(10);                // sleep for 10 msec
                    } catch (InterruptedException t){}
            repaint();
                    }
      } // end of paint



       void addborder(int i,int j){
      } // end of addborder
}  // end of class
```

Dodaj przycisk START, który jest nieaktywny dopóki użytkowni nie wpisze ilości kroków. Odrysowanie (po naciśnięciu START) ponownie dezaktywuje przycisk. Obsługa zdarzeń przycisku poprzez anonimową klasę wewnętrzną.

```
przycisk.addActionListener(new ActionListener() {
                  public void actionPerformed(ActionEvent e){
                ……..
                  }  // end of actionPerformed
            });
```

Węzły należące do klastra pokoloruj w sposób odpowiadający kolejności ich dodawania. Np. najpóźniej dodane węzły na czerwono (Color(255,0,0)) a najwcześniej na niebieski (Color(0,0,255)).

Zmodyfikuj konfigurację początkową, tak aby wzrost klastra rozpoczynał się od górnej krawędzi. Pomocne mogą okazać się poziome periodyczne warunki brzegowe.

W wersji gdzie początkowo klaster umieszczony jest na górnej krawędzi wprowadź aktywność węzła (0<aktywność[][]<1). Wybrany z tablicy border węzeł jest akceptowany z prawdopodobieństwem aktywność[][]. W przypadku odrzucenia, nowy węzeł brzegowy jest losowany. Aktywność jest dziedziczna: dodając węzeł do brzegu, przepisujemy aktywność z węzła, który aktywował ten dodawany węzeł.