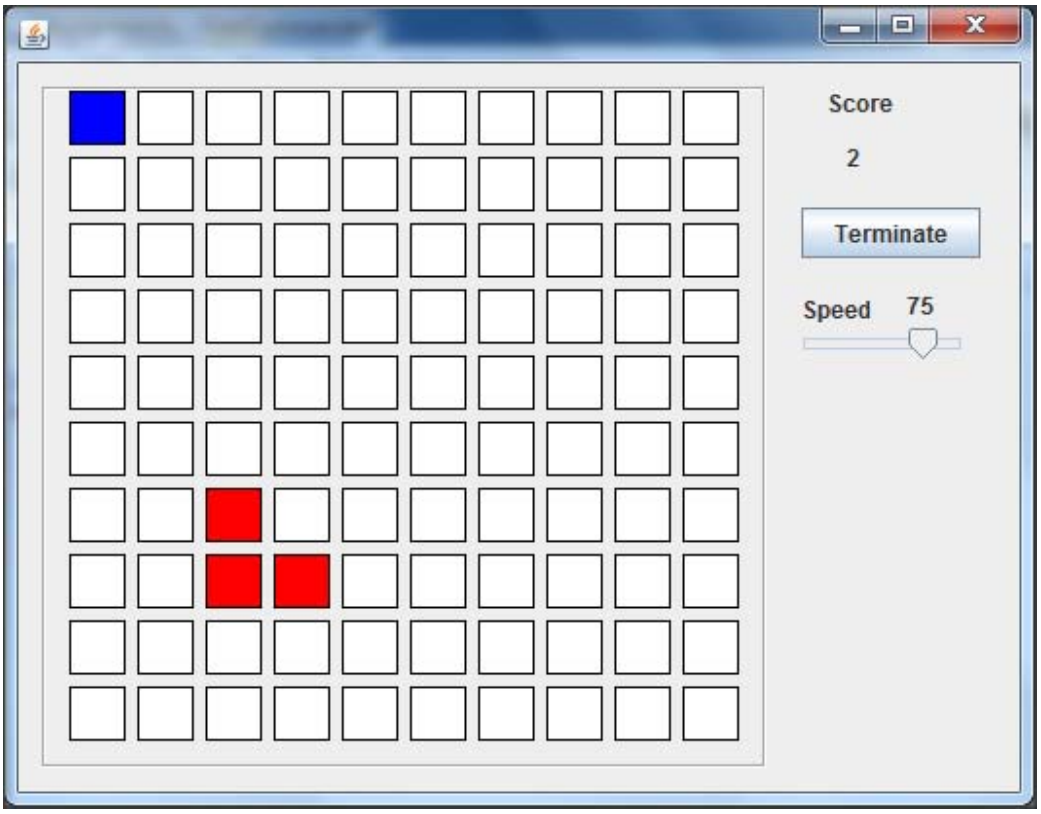
جامعة الأهرام الكندية
AHRAM CANADIAN UNIVERSITY

Fundamentals of Programming II Assignment 08
Snake Phase 03




```
int board[][] = {{0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
      {0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
      {0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
      {0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
      {0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
      {0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
      {0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
      {0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
      {0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
      {0, 0, 0, 0, 0, 0, 0, 0, 0, 0}};
   int targets[][] = {{0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
      {0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
      {0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
      {0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
      {0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
      {0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
      {0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
      {0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
      {0, 0, 0, 0, 0, 0, 0, 0, 0, 0},
      {0, 0, 0, 0, 0, 0, 0, 0, 0, 0}};
   int move[][]={{9,0},
      {-1,-1},
      {-1,-1},
      {-1,-1},
```

```
    {-1,-1},
    {-1,-1},
    {-1,-1},
    {-1,-1},
    {-1,-1},
    {-1,-1},
  };
  int length=1;
  boolean target=false;
  String direction = "RIGHT";
  int delay = 1000;
  Timer t;
```

```
boolean checkWin()
 {
   for(int r=0;r<10;r++)
   {
     for(int c=0;c<10;c++)
     {
       if(targets[r][c]==1)
       {
         //if(board[r][c]==1)
         if((move[0][0]==r) && (move[0][1]==c))
         {
           target=false;
           targets[r][c]=0;
           Score.setText(Integer.toString(Integer.parseInt(Score.getText())+1));
           length++;
           if(length==10)
           {
             finishGame();
           }
           updateBoard();
           return true;
         }
       }
     }
   }
   return false;
 }
```

```
void renderTargets()
    {
     if(target==false)
     {
     Random rand = new Random();
    int r = rand.nextInt(10);
    int c=rand.nextInt(10);
    targets[r][c]=1;
    target=true;
     }

     {
        if(targets[0][0]==1)_00.setBackground(Color.blue);
        if(targets[0][1]==1)_01.setBackground(Color.blue);
       .
       .
        if(targets[9][8]==1)_98.setBackground(Color.blue);
        if(targets[9][9]==1)_99.setBackground(Color.blue);
     }
```

```
    }
```

```
void renderBoard() {
    for(int r=0;r<10;r++)
    {
       for(int c=0;c<10;c++)
          board[r][c]=0;
    }
    for (int n=0;n<length;n++)
    {
       board[move[n][0]][move[n][1]]=1;

    }
    if(board[0][0]==0)_00.setBackground(Color.white); else  _00.setBackground(Color.red);
    if(board[0][1]==0)_01.setBackground(Color.white); else  _01.setBackground(Color.red);
.
.
    if(board[9][8]==0)_98.setBackground(Color.white); else  _98.setBackground(Color.red);
    if(board[9][9]==0)_99.setBackground(Color.white); else  _99.setBackground(Color.red);
  }
```

```
void updateBoard() {
        int r, c;
        int[][] head=new int[1][2];
        head[0][0]=move[0][0];
        head[0][1]=move[0][1];
        // move the head
        if (direction.equals("LEFT")) {
            if(move[0][1]==0)move[0][1]=9;
            else move[0][1]=move[0][1]-1;
        }
        if (direction.equals("RIGHT")) {
            move[0][1]=(move[0][1]+1)%10;
        }
        if (direction.equals("UP")) {
            if(move[0][0]==0)move[0][0]=9;
            else move[0][0]=move[0][0]-1;
        }
        if (direction.equals("DOWN")) {
            move[0][0]=(move[0][0]+1)%10;
        }
       // move the body
        for(int n=length-1;n>=2;n--)
        {
                move[n][0]=move[n-1][0];
                move[n][1]=move[n-1][1];
        }
        if(length>1)
        {
            move[1][0]=head[0][0];move[1][1]=head[0][1];
        }
        boolean b=checkWin();
    renderBoard();
    renderTargets();
}
```

```
void finishGame()
  {

    jButton1.setText("Restart");
    t.stop();
```

```
    JOptionPane.showMessageDialog(this, "You Win.");
  }
```

```
private void formKeyPressed(java.awt.event.KeyEvent evt) {//GEN-FIRST:event_formKeyPressed
    int r,c;
    switch (evt.getKeyCode())
    {
      case  KeyEvent.VK_LEFT:
          direction="LEFT";
        break;
      case  KeyEvent.VK_RIGHT:
        direction="RIGHT";
        break;
      case  KeyEvent.VK_UP:
        direction="UP";
        break;
      case  KeyEvent.VK_DOWN:
        direction="DOWN";
        break;
    }

  }
```

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-
FIRST:event_jButton1ActionPerformed
        delay=(101-jSlider1.getValue())*20;
        if(jButton1.getText().equals("Start"))
        {
            t= new Timer(delay,new  ActionListener() {
                @Override
                      public void actionPerformed(ActionEvent e) {
                          updateBoard();
                      }
                  });
            t.start();
            jButton1.setText("Terminate");
            jSlider1.disable();
            renderBoard();
            renderTargets();
        }
        else if(jButton1.getText().equals("Terminate"))
        {
            t.stop();
            jButton1.setText("Start");
            jSlider1.enable();
        }
        else if(jButton1.getText().equals("Restart"))
        {
        t.stop();
        jButton1.setText("Start");
        jSlider1.enable();
        move[0][0]=9;move[0][1]=0;
        for(int r=1;r<10;r++)
            {move[r][0]=-1;move[r][1]=-1;}
        for(int r=0;r<10;r++)
        {
            for(int c=0;c<10;c++)
            {
                targets[r][c]=0;
            }
        }
```

```
        length=1;
        Score.setText("0");
        target=false;
        direction = "RIGHT";
        renderBoard();
        renderTargets();

        }
    }//
```