[Write on your own sheets of paper. On each sheet, on the front, write in the top right corner your **lastname, firstname(s), group id and** (A). Leave a 2cm margin at the left on the front page / at the right on the back for stapling. Open book. Time: 90 minutes]

The Bear and the Seals

You are to develop a standalone application which implements a board game, simulating something happening in/on a frozen lake somewhere up North. The game is played on a 10 by 20 rectangular board.
On the board – representing the lake – there is one bear, a number of seal holes, between 2 and 6, and a number of seals – between 3 and 7. This numbers are read from the configuration file, named `Bear_Seals.txt` when the game starts.

- A seal can swim under water for up to 5 steps. After taking a breath the seal moves away at some random position, leaving the hole empty for another seal. At some moment (from 1 and up to 5 steps away from the previous breath) some seals will pop up to the nearest hole to take a fresh breath of air. If they get to a hole where the bear is waiting, the first to reach they gets eaten. A seal can die if it does not get to a hole for more than 5 steps.
- The bear waits at one of the seal holes for a seal to come and take a fresh breath. The number of steps the bear waits for a seal to appear at a hole is between 2 and 6. After eating one seal, the bear will not move or eat for the next 3 steps. The moves of the bear are random, i.e. after the wait period, if no seal appears, the bear will jump to a *randomly chosen* hole. The bear is initially upon a hole (actually in very close proximity) waiting for a seal.

Figure 1 shows an example initial configuration and how it is represented in the text file. A step is taken when the user presses the 'Enter' key.
At the beginning of the game, and also after each step you should print to `System.out` – *text mode* – the current configuration. On that printout, the Bear is symbolized by capital letter 'B' (rememember that also marks a hole), the seals beneath ice with lowercase 's', the holes where there are seals with 'S', the holes with no seals with 'H', and the rest of the squares with dots ('.').
The game ends when the bear has eaten all seals or the seals have all died or 300 steps passed without this unhappy (for the seals, of course) event to occur.
Draw the class diagram and develop a java standalone application to simulate this game. Don't forget to briefly document it.

For the following contents of `Bear_Seals.txt`:

```
B 5 11
5 H 7 3 H 8 18 H 2 16 H 2 18 H 1 19
7 S 1 3 S 8 18 S 3 5 S 0 0 S 1 1 S 0 1 S 9 19
```

, the initial printout is:

```
 00000000001111111111
 01234567890123456789
0ss.................H
1.s.s...............H
2.................H.H.
3.....s..............
4....................
5...........B........
6....................
7...H...............s.
8..................S.
9...................s
Press Enter...
```

The digits at the top represent column numbers – row 1 is for tens, row 2 for units. The digits on the left represent row numbers.

```
/**
 * Write a description of class Game here.
 *
 * @author (your name)
 * @version (a version number or a date)
 */
import java.util.*;
public class Game
{
   public static void main()
   {
      Scanner sc=new Scanner(System.in);
      Lake frozen=new Lake();
      while(!frozen.gameOver()){
         frozen.view();
         frozen.animate();
         System.out.println("Press Enter to continue...");
         sc.nextLine();
      }
      System.out.println("Simulation over!");
      frozen.view();
   }
}
```

```
/**
 * Write a description of class Animals here.
 *
 * @author (your name)
 * @version (a version number or a date)
 */
import java.awt.*;
public class Animals
{
   private Point coord;
   private int step=1;
   public Animals(int x, int y)
   {
      coord=new Point(x,y);

   }
   public int getStep()
   {
   return step;
   }
   public void setStep(int x)
   {
   step=x;
   }
   public void increaseStep()
```

```
    {
    step++;
    }
    public void setCoord(int x, int y)
    {
    coord=new Point(x,y);
    }
    public Point getCoord()
    {
    return coord;
    }
    public boolean move(Point[] holes)
    {
     return false;
    }
}




/**
 * Write a description of class Lake here.
 *
 * @author (your name)
 * @version (a version number or a date)
 */
import java.util.*;
import java.util.Iterator;
import java.awt.*;
import java.io.*;
public class Lake
{
    ArrayList <Seals> seals=new ArrayList<Seals>();
    Bear bear;
    Point [] holes;
    private int nrseals;
    private int nrholes;
    private int totalSteps;
    public Lake()
    {
        totalSteps=0;
        Scanner s=null;
        BufferedReader in=null;
        String[] line=new String[3];
        nrholes=0;
        nrseals=0;
        int x,y;
        try{
            s=new Scanner(new BufferedReader(new FileReader("bearseals.txt")));
            //s.useDelimiter("//H//B//S");
            s.next();//citim B
            x=s.nextInt();
            y=s.nextInt();
            bear=new Bear(x,y);
```

```
        nrholes=s.nextInt()+1;
        holes=new Point[nrholes];
        holes[0]=new Point(x,y);//the bear stays at a hole too
        for(int i=1;i<nrholes;i++)
        {
           s.next();
           x=s.nextInt();
           y=s.nextInt();
           holes[i]=new Point(x,y);
        }
        nrseals=s.nextInt();
        for(int i=0;i<nrseals;i++)
        {
           s.next();
           x=s.nextInt();
           y=s.nextInt();
           seals.add(new Seals(x,y));
           //check if seal at a hole, then set step to 5
           for(int j=0;j<nrholes;j++)
           {
              if((holes[j]).equals((seals.get(i)).getCoord())) (seals.get(i)).setStep(0);
           }
        }
     }catch(IOException e){System.out.println("File not open for input!");}
     finally{
        try{
           if(in!=null)in.close();
        }catch(IOException e2){System.out.println("Error closing file!");}
     }

}

public boolean gameOver()
{
   boolean ok=false;
   if(totalSteps>300) ok=true;
   if(seals.size()==0) ok=true;// ok=seals.isEmpty();
   return ok;
}
public void view()
{
    Point auxp;
    boolean ok=true;
    for(int i=0;i<10;i++)
    {
        for(int j=0;j<20;j++)
        {
           auxp=new Point(i,j);
           //verificam daca e urs
           if(auxp.equals(bear.getCoord()))System.out.print("B");
           else
           {
              //verificam daca e gaura
              ok=true;
```

```
            Seals aus;
            Iterator <Seals> it = seals.iterator();
            while((ok)&&(it.hasNext()))
            {
               aus=it.next();
               if(auxp.equals(aus.getCoord())){
                  ok=false;

                  if(aus.getStep()==0) System.out.print("S");
                  else  System.out.print("s");
                  // System.out.print("s");
               }
            }
            if(ok){
               //nu e nici gaura, verificam foca
               for(int k=0;k<nrholes;k++)
                  if((ok)&&(auxp.equals(holes[k]))) {
                  ok=false;
                  System.out.print("H");
               }

            }
            if(ok)System.out.print("_");//not bear, nor hole nor seal
         }

      }
      System.out.println();
   }

}
public void animate()
{
  totalSteps++;
  bear.move(holes);
  Iterator <Seals> it=  seals.iterator();
  boolean ok;
  Seals aus=null;
  System.out.println("Seals remaining "+nrseals);
  while(it.hasNext())
  {
   aus=it.next();
   aus.move(holes);
   //ursul vaneaza foca, daca aceasta e la o gaura
   if(!bear.eat){
   if(aus.getStep()==0){
     ///if((aus.getCoord()).equals(holes[bear.currHole])) {
        if((aus.getCoord()).equals(bear.getCoord())) {
        it.remove();
        nrseals--;
        bear.eat=true;
        System.out.println("Seal eaten!");
        bear.setStep(0);
     }
   }
```

```
        }
    }

    }
}
```

```
/**
 * Write a description of class Bear here.
 *
 * @author (your name)
 * @version (a version number or a date)
 */
import java.awt.*;
import java.util.*;
public class Bear extends Animals
{
    public boolean eat=false;
    private int waitSteps=6;
    public int currHole=0;
    public Bear(int x, int y)
    {
      super(x,y);
      waitSteps=6;
    }
   public boolean move(Point[] holes)
   {
     int currSteps=super.getStep();
     Random r=new Random();
     //mutam ursul la alta gaura
     if((!eat)&&(currSteps>=waitSteps)){
        waitSteps=r.nextInt(5)+2;//minim 2 maxim 6
        //move near a next hole
        currHole=r.nextInt(holes.length);
        this.setCoord((int)holes[currHole].getX(),(int)holes[currHole].getY());
        System.out.println("Move bear to hole number "+currHole+" and wait for "+waitSteps+" steps");
        this.setStep(0);
        }
        if(eat){
         if(currSteps==3) eat=false;
        }
    this.increaseStep();
    return true;
    }

}
```

```
/**
```

```
 * Write a description of class Seals here.
 *
 * @author (your name)
 * @version (a version number or a date)
 */
import java.awt.*;
import java.util.*;
public class Seals extends Animals
{

   public Seals(int x, int y)
   {
      super(x,y);
   }


   public double distance(Point p, Point q)
   {
      double d=((p.getX()-q.getX()))*((p.getX()-q.getX()))+
         ((q.getY()-p.getY()))*((q.getY()-p.getY()));
      d=Math.sqrt(d);
      return d;

   }


   public boolean move(Point[] holes){
      boolean ok=true;
      Random r=new Random();
      if(this.getStep()==5){
         //move to the closest hole
         Point aus=this.getCoord();
         int hmin=0;
         double dmin=distance(aus,holes[0]);
         for(int i=1;i<holes.length;i++)
            if(distance(aus,holes[i])<dmin)
            {
               dmin=distance(aus,holes[i]);
               hmin=i;
         }
         this.setCoord((int)holes[hmin].getX(),(int)holes[hmin].getY());
         this.setStep(0);//seal in hole
      }else
      {
      if(this.getStep()==0) {//seal in hole
         //move to a random position
         int move= r.nextInt(200);
         this.setCoord(move/20,move%20);
      }
         this.increaseStep();}
      return ok;
   }

}
```