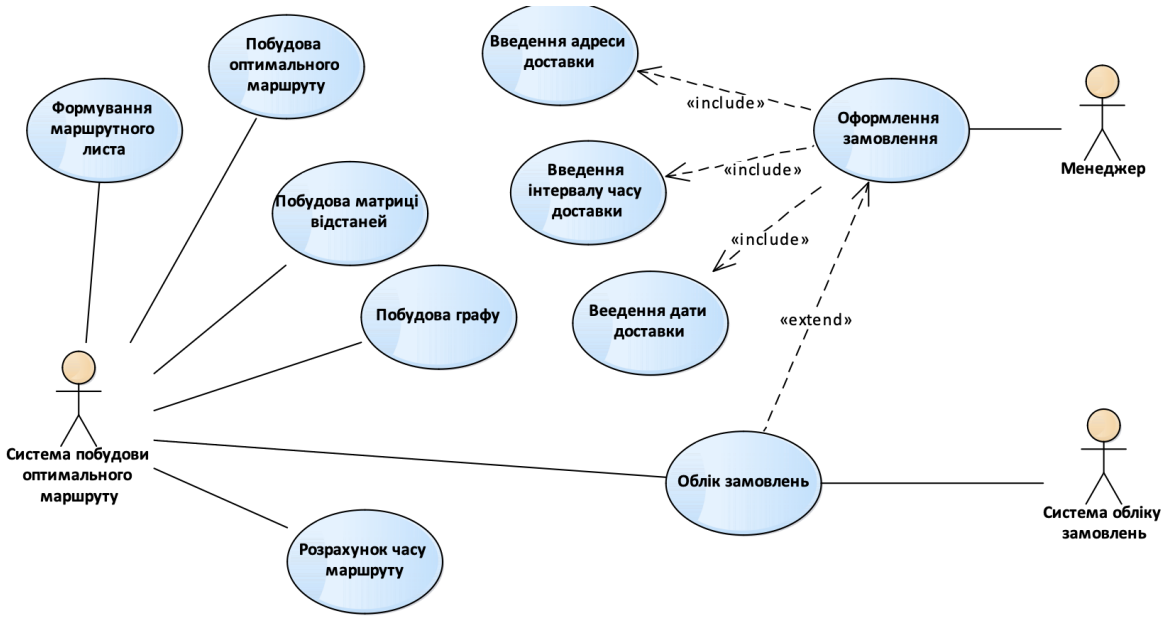
ДОДАТОК А

Згенерований звіт моделювання системи

use case diagram

Use Case diagram in package 'use case'

use case
Version 1.0
Profi created on 18.05.2020. Last modified 19.05.2020

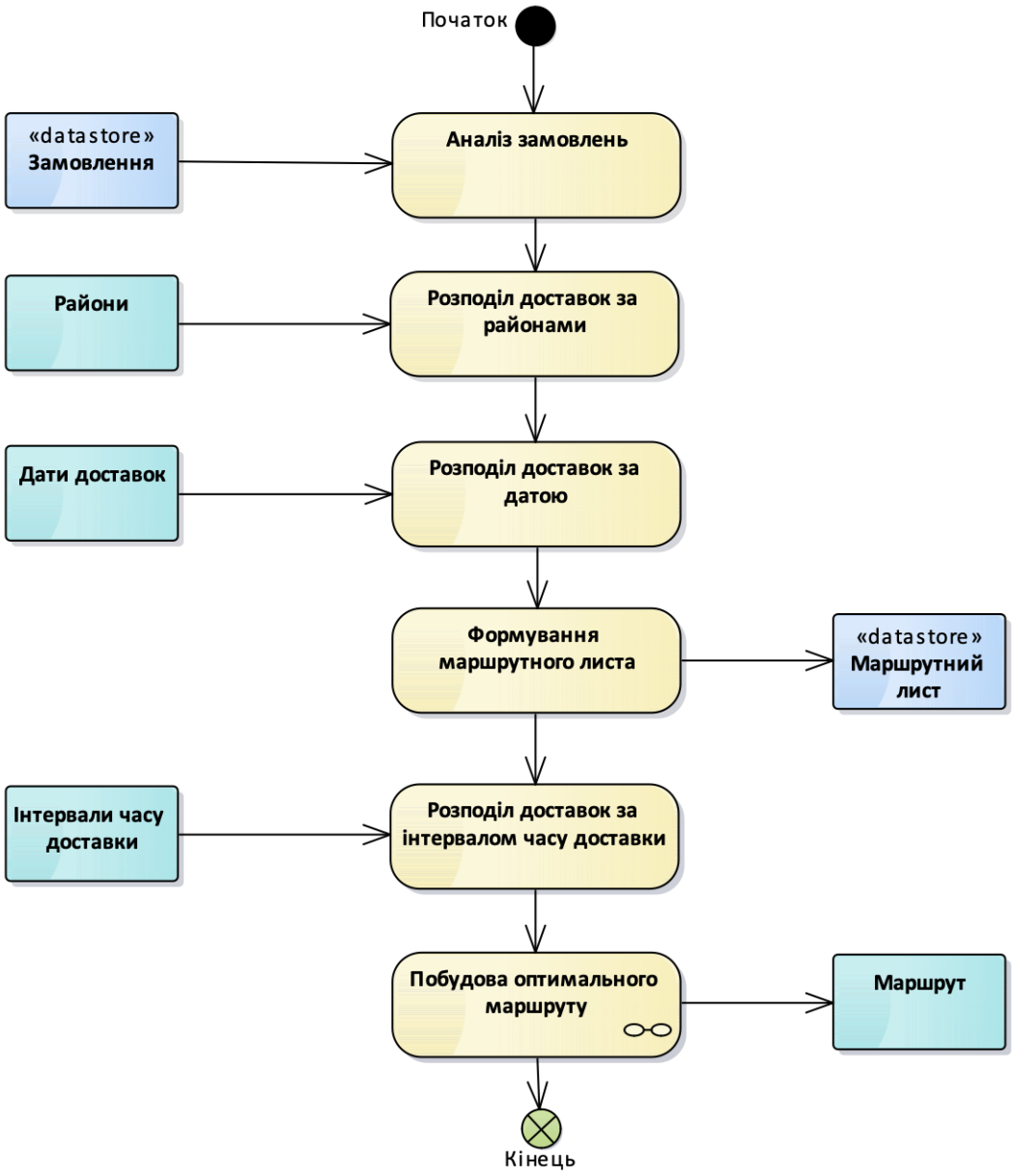


5. use case

алгоритм роботи diagram

Activity diagram in package 'алгоритм роботи'

алгоритм роботи
Version 1.0
Profi created on 18.05.2020. Last modified 19.05.2020

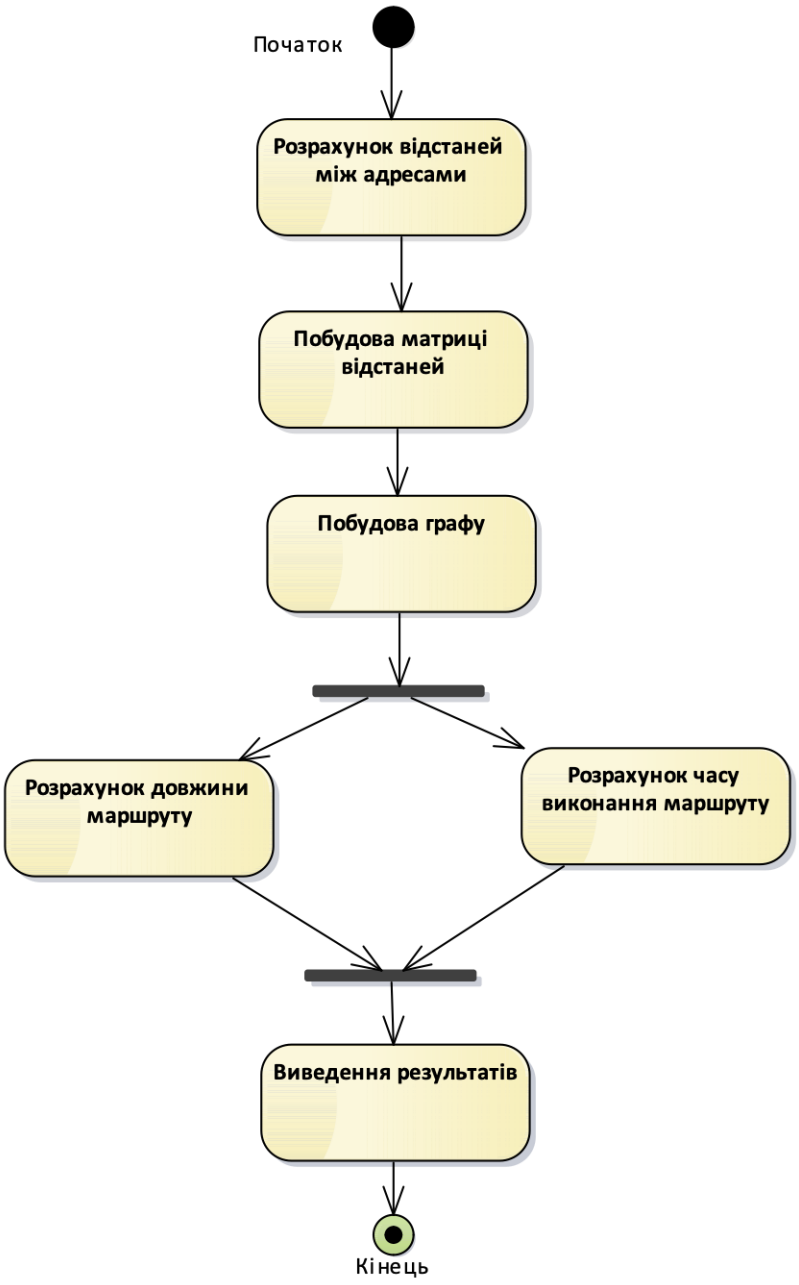


6. алгоритм роботи

Побудова оптимального маршруту diagram

Activity diagram in package 'алгоритм роботи'

Побудова оптимального маршруту
Version 1.0
Profi created on 19.05.2020. Last modified 19.05.2020

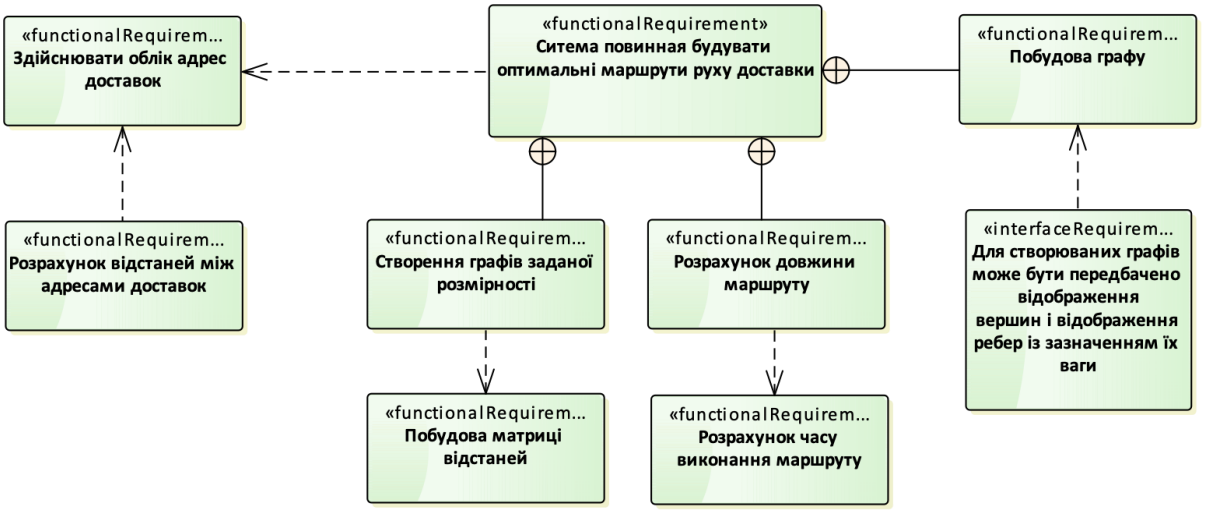


7. Побудова оптимального маршруту

вимоги diagram

SysML Requirements diagram in package 'вимоги'

вимоги
Version 1.0
Profi created on 18.05.2020. Last modified 19.05.2020

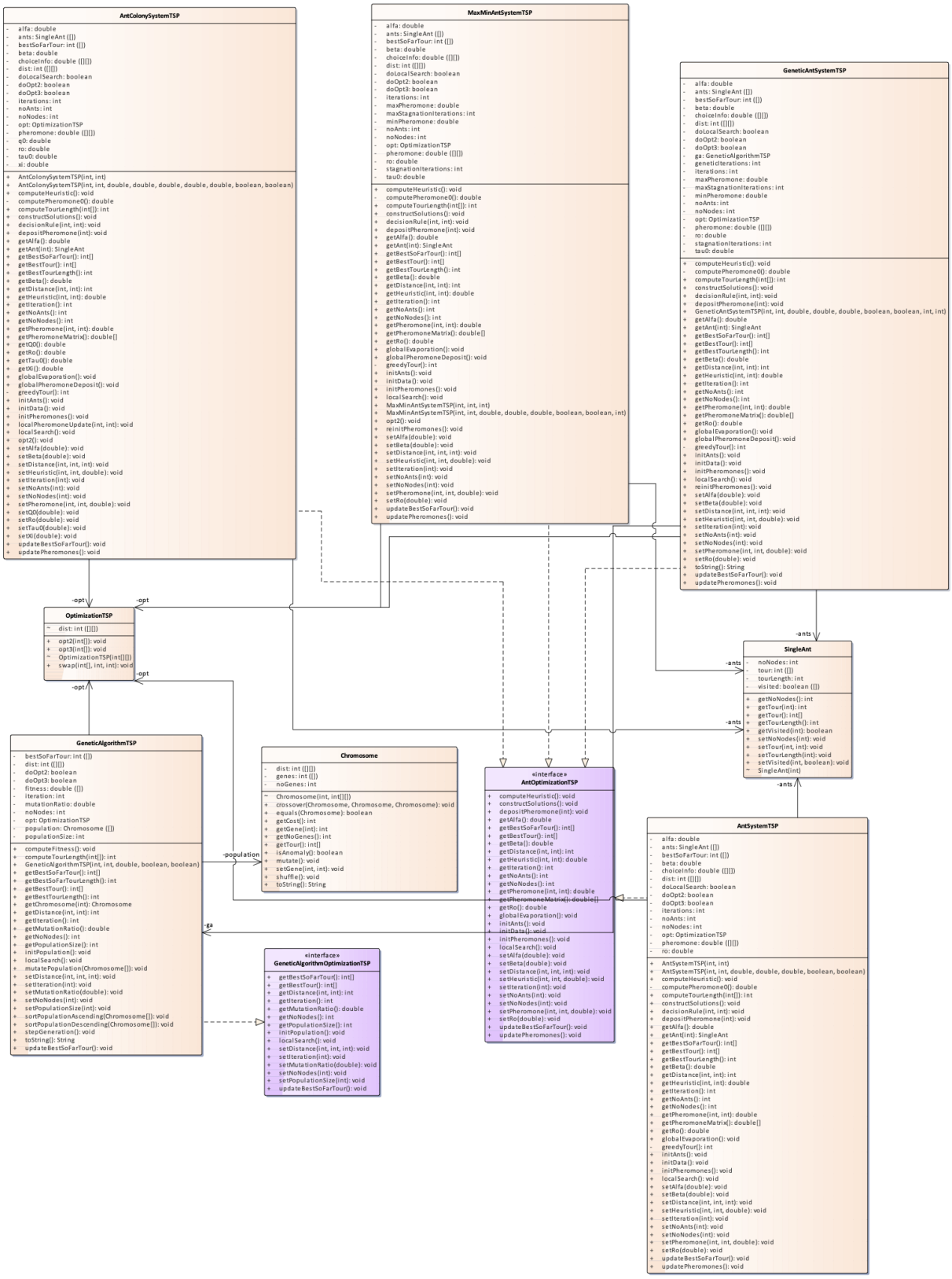


8. вимоги

класи diagram

Class diagram in package 'класи'

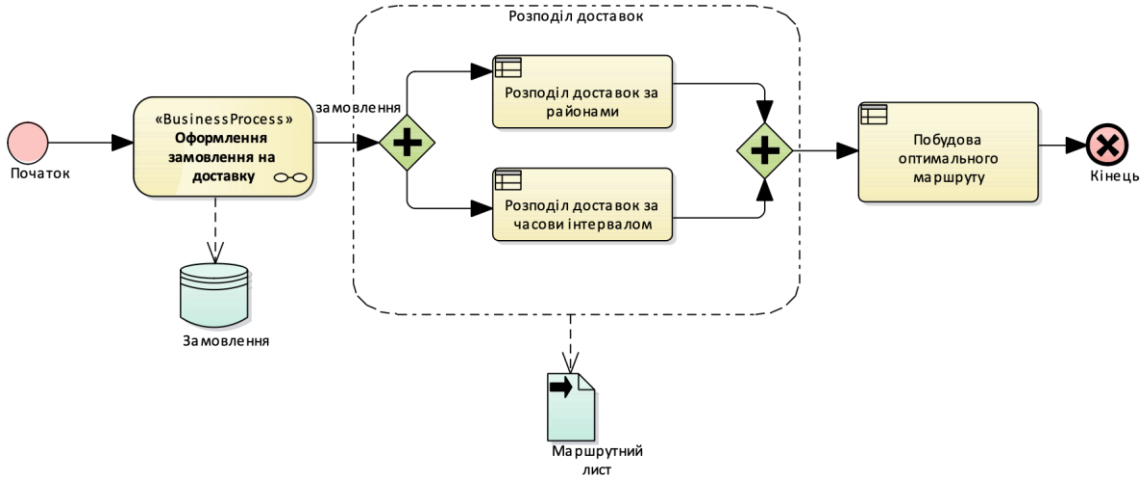
класи
Version 1.0
Profi created on 20.05.2020. Last modified 20.05.2020



процеси diagram

Business Process diagram in package 'процеси'

процеси
Version 1.0
Profi created on 18.05.2020. Last modified 19.05.2020

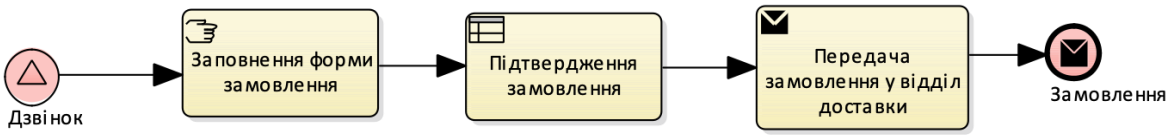


9. процеси

Формування diagram

Business Process diagram in package 'процеси'

Формування
Version 1.0
Profi created on 18.05.2020. Last modified 19.05.2020



10. Формування

ДОДАТОК Б

Код програми «Мурашиний алгоритм»

Додаток Б.1. Код класу “Проста мураха”

```
public class SingleAnt{
   SingleAnt(int n){
      tour = new int[n+1];
      visited = new boolean[n];
      tourLength = 0;
      noNodes = n;
   }
   private int noNodes;
   private int tourLength;
   private int tour[];
   private boolean visited[];

   public int getTourLength(){
      return tourLength;
   }
   public void setTourLength(int len){
      tourLength = len;
   }
   public int getNoNodes(){
      return tourLength;
   }
   public void setNoNodes(int len){
      tourLength = len;
   }
   public boolean getVisited(int idx){
      return visited[idx];
   }
   public void setVisited(int idx, boolean val){
      visited[idx] = val;

   }
   public int getTour(int idx){
      return tour[idx];
   }
   public void setTour(int idx, int val){
      tour[idx] = val;

   }
   public int[] getTour(){
      return tour;
   }
}
```

Додаток Б.2. Код класу “Мурашина колонія”

```
public class AntSystemTSP implements AntOptimizationTSP{

   private int dist[][];

   private double pheromone[][];

   private double choiceInfo[][];

   private SingleAnt ants[];

   private int noAnts;

   private int noNodes;
```

```
private double alfa;

private double beta;

private double ro;

private int iterations;

private boolean doLocalSearch;

private boolean doOpt2;

private boolean doOpt3;

private int[] bestSoFarTour;
private OptimizationTSP opt;

public AntSystemTSP(int noNodes, int noAnts){
   this.noNodes = noNodes;
   this.noAnts = noAnts;
   this.alfa = 1.0;
   this.beta = 3.0;
   this.ro = 0.5;
   this.iterations = 0;

   dist = new int[noNodes][noNodes];
   pheromone = new double[noNodes][noNodes];
   choiceInfo = new double[noNodes][noNodes];

   ants = new SingleAnt[noAnts];
   for(int i = 0; i < noAnts; i++){
      ants[i] = new SingleAnt(noNodes);
   }
   this.doOpt2 = false;
   this.doOpt3 = false;
   bestSoFarTour = new int[noNodes+1];
}
public AntSystemTSP(int noNodes, int noAnts, double alfa,
            double beta, double ro, boolean doOpt2, boolean doOpt3){
   this.noNodes = noNodes;
   this.noAnts = noAnts;
   this.alfa = alfa;
   this.beta = beta;
   this.ro = ro;
   this.iterations = 0;

   dist = new int[noNodes][noNodes];
   pheromone = new double[noNodes][noNodes];
   choiceInfo = new double[noNodes][noNodes];

   ants = new SingleAnt[noAnts];
   for(int i = 0; i < noAnts; i++){
      ants[i] = new SingleAnt(noNodes);
   }

   this.doOpt2 = doOpt2;
   this.doOpt3 = doOpt3;
   bestSoFarTour = new int[noNodes+1];
}

public SingleAnt getAnt(int k){
   return ants[k];
}
```

```
public double getPheromone(int i, int j){
   return pheromone[i][j];
}
public double getHeuristic(int i, int j){
   return choiceInfo[i][j];
}
public int getDistance(int i, int j){
   return dist[i][j];
}
public void setPheromone(int i, int j, double ph){
   pheromone[i][j] = ph;
}
public void setHeuristic(int i, int j, double h){
   choiceInfo[i][j] = h;
}
public void setDistance(int i, int j, int d){
   dist[i][j] = d;
}
public void setAlfa(double a){
   alfa = a;

}
public void setBeta(double b){
   beta = b;
}
public void setRo(double r){
   ro = r;
}
public double getAlfa(){
   return alfa;
}
public double getBeta(){
   return beta;
}
public double getRo(){
   return ro;
}
public int[] getBestTour(){
   int[] bestTour = new int[noNodes+1];
   int bestTourLength = Integer.MAX_VALUE;
   for(int i = 0; i < noAnts; i++){
      if(ants[i].getTourLength() < bestTourLength){
         bestTourLength = ants[i].getTourLength();
         for(int j = 0; j <=noNodes; j++)
            bestTour[j] = ants[i].getTour(j);
      }
   }
   return bestTour;
}
public int getBestTourLength(){
   return computeTourLength(getBestTour());
}
public int[] getBestSoFarTour(){
   return bestSoFarTour;
}
public void updateBestSoFarTour(){
   if(getBestTourLength() < computeTourLength(bestSoFarTour)){
      bestSoFarTour = getBestTour();
   }
}
public void setIteration(int iter){
   iterations = iter;
```

```
}
public int getIteration(){
   return iterations;
}
public int getNoAnts(){
   return noAnts;
}
public void setNoAnts(int ants){
   noAnts = ants;
}
public int getNoNodes(){
   return noNodes;
}
public void setNoNodes(int nodes){
   noNodes = nodes;
}
public void initData(){
   int i,j;
   for(i = 0; i < noNodes; i++)
      for(j = 0; j < noNodes; j++){
         dist[i][j] = 0;
         pheromone[i][j] = 0.0;
         choiceInfo[i][j] = 0.0;
      }
}
public void initPheromones(){
   int i,j;
   double tau0 = computePheromone0();

   for(i = 0; i < noNodes; i++)
      for(j = 0; j < noNodes; j++)
         pheromone[i][j] = tau0;

   for(i = 0; i < noNodes; i++)
      pheromone[i][i] = 0;

   opt = new OptimizationTSP(dist);
}
public void computeHeuristic(){
   double niu;
   int i,j;

   for(i = 0; i < noNodes; i++)
      for(j = 0; j < noNodes; j++){
         if(dist[i][j] > 0)
            niu = 1.0/dist[i][j];
         else
            niu = 1.0/0.0001;
      choiceInfo[i][j] = Math.pow(pheromone[i][j],alfa)*Math.pow(niu,beta);
   }
}
public void initAnts(){
   int i,j;
   for(i = 0; i < noAnts; i++){
      ants[i].setTourLength(0);

      for(j = 0; j < noNodes; j++)
         ants[i].setVisited(j, false);
      for(j = 0; j <= noNodes; j++)
         ants[i].setTour(j, 0);
   }
}
public void decisionRule(int k, int step){
```

```
        int c = ants[k].getTour(step-1);

double sumProb = 0.0;

        double selectionProbability[] = new double[noNodes];

        int j;
        for(j = 0; j < noNodes; j++){
           if((ants[k].getVisited(j)) || (j == c))
              selectionProbability[j] = 0.0;
           else{
              selectionProbability[j] = choiceInfo[c][j];
              sumProb+=selectionProbability[j];
           }

        }
        double prob = Math.random()*sumProb;
        j = 0;
        double p = selectionProbability[j];
        while(p < prob){
           j++;
           p += selectionProbability[j];
        }
        ants[k].setTour(step, j);
        ants[k].setVisited(j, true);

     }
     public void constructSolutions(){
        /* stergere memorie furnici */
        initAnts();

        int step = 0;
        int k;
        int r;

        Random rand = new Random();

        /* asignare oras initial */
        for(k = 0; k < noAnts; k++){
           r = Math.abs(rand.nextInt())%noNodes;

           ants[k].setTour(step,r);
           ants[k].setVisited(r,true);
        }
        /* construirea efectiva a solutiei */
        while(step < noNodes-1){
           step++;
           for(k = 0; k < noAnts; k++)
              decisionRule(k,step);
        }
        /* completarea turului */
        for(k = 0; k < noAnts; k++){
           ants[k].setTour(noNodes,ants[k].getTour(0));
           ants[k].setTourLength(computeTourLength(ants[k].getTour()));
        }
        updateBestSoFarTour();
     }
     public void globalEvaporation(){
        int i,j;
        for(i = 0; i < noNodes; i++)
           for(j = 0; j < noNodes; j++){
              pheromone[i][j] = (1-ro)*pheromone[i][j];
```

```
        }
    }
    public void depositPheromone(int k){
        double delta_tau = 1.0/ants[k].getTourLength();
        int left,right;
        for(int i = 0; i < noNodes; i++){
            left = ants[k].getTour(i);
            right = ants[k].getTour(i+1);
            pheromone[left][right]+=delta_tau;
            pheromone[right][left]+=delta_tau;
        }
    }
    public void updatePheromones(){
        globalEvaporation();
        for(int k = 0; k < noAnts; k++)
            depositPheromone(k);
        computeHeuristic();
    }
    private int greedyTour(){
        boolean visited[] = new boolean[noNodes];
        int tour[] = new int[noNodes+1];
        int length;
        int min, node;
        int i,j;

        for(i = 0; i < noNodes; i++)
            visited[i] = false;

        tour[0] = 0;
        bestSoFarTour[0] = 0;
        visited[0] = true;

        for(i = 1; i < noNodes; i++){
            min = Integer.MAX_VALUE;
            node = -1;
            for(j = 0; j < noNodes; j++){
                if((!visited[j])&&(j!=tour[i-1])){
                    if(min > dist[tour[i-1]][j]){
                        min = dist[tour[i-1]][j];
                        node = j;
                    }
                }
            }
            tour[i] = node;
            bestSoFarTour[i] = node;
            visited[node] = true;
        }
        tour[noNodes] = tour[0];
        bestSoFarTour[noNodes] = bestSoFarTour[0];
        return computeTourLength(tour);

    }
    public int computeTourLength(int tour[]){
        int len = 0;
        for(int i = 0; i < noNodes; i++){
            len+=dist[tour[i]][tour[i+1]];
        }
        return len;
    }
    private double computePheromone0(){
        return ((double)noAnts)/((double)greedyTour());
    }
```

```
    public void localSearch(){
    /* Procedurile de cautare locala */
        if(doOpt2){
            for(int i = 0; i < noAnts; i++){
                opt.opt2(ants[i].getTour());
            }
        }
        if(doOpt3){
            for(int i = 0; i < noAnts; i++){
                opt.opt3(ants[i].getTour());
            }
        }


        updateBestSoFarTour();
    }
    public double[][] getPheromoneMatrix(){
        return pheromone;
    }


}
```

Додаток Б.3. Інтерфейс мурашиної оптимізації

```
public interface AntOptimizationTSP {
    public double getPheromone(int i, int j);
    public double getHeuristic(int i, int j);
    public int getDistance(int i, int j);
    public void setPheromone(int i, int j, double ph);
    public void setHeuristic(int i, int j, double h);
    public void setDistance(int i, int j, int d);
    public void setAlfa(double a);
    public void setBeta(double b);
    public void setRo(double r);
    public double getAlfa();
    public double getBeta();
    public double getRo();
    public int[] getBestTour();
    public void setIteration(int iter);
    public int getIteration();
    public int getNoAnts();
    public void setNoAnts(int ants);
    public int getNoNodes();
    public void setNoNodes(int nodes);
    public void initData();
    public void initPheromones();
    public void computeHeuristic();
    public void initAnts();
    public void constructSolutions();
    public void globalEvaporation();
    public void updatePheromones();
    public void depositPheromone(int idx);
    public void localSearch();
    public int[] getBestSoFarTour();
    public void updateBestSoFarTour();
    public double[][] getPheromoneMatrix();
}
```

Додаток Б.4. Клас алгоритму оптимізації

```
public class AntColonySystemTSP implements AntOptimizationTSP{
```

```
private int dist[][];
private double pheromone[][];
private double choiceInfo[][];
private SingleAnt ants[];

private int noAnts;
private int noNodes;
private double alfa;
private double beta;
private double ro;
private double xi;
private double tau0;
private double q0;
private int iterations;
private boolean doLocalSearch;
private boolean doOpt2;
private boolean doOpt3;

private OptimizationTSP opt;

private int[] bestSoFarTour;


public AntColonySystemTSP(int noNodes, int noAnts){
   this.noNodes = noNodes;
   this.noAnts = noAnts;
   this.alfa = 1.0;
   this.beta = 3.0;
   this.ro = 0.1;
   this.xi = 0.1;
   this.iterations = 0;
   this.tau0 = 0;
   this.q0 = 0.9;

   dist = new int[noNodes][noNodes];
   pheromone = new double[noNodes][noNodes];
   choiceInfo = new double[noNodes][noNodes];

   ants = new SingleAnt[noAnts];
   for(int i = 0; i < noAnts; i++){
      ants[i] = new SingleAnt(noNodes);
   }
   doLocalSearch = false;
   bestSoFarTour = new int[noNodes+1];

}
public AntColonySystemTSP(int noNodes, int noAnts, double alfa,
            double beta, double ro, double xi, double q0, boolean doOpt2, boolean doOpt3){
   this.noNodes = noNodes;
   this.noAnts = noAnts;
   this.alfa = alfa;
   this.beta = beta;
   this.ro = ro;
   this.xi = xi;
   this.iterations = 0;
   this.tau0 = 0;
   this.q0 = q0;

   dist = new int[noNodes][noNodes];
   pheromone = new double[noNodes][noNodes];
   choiceInfo = new double[noNodes][noNodes];
```

```
    ants = new SingleAnt[noAnts];
    for(int i = 0; i < noAnts; i++){
        ants[i] = new SingleAnt(noNodes);
    }
    this.doOpt2 = doOpt2;
    this.doOpt3 = doOpt3;
    bestSoFarTour = new int[noNodes+1];
}

public SingleAnt getAnt(int k){
    return ants[k];
}

public double getPheromone(int i, int j){
    return pheromone[i][j];
}
public double getHeuristic(int i, int j){
    return choiceInfo[i][j];
}
public int getDistance(int i, int j){
    return dist[i][j];
}
public void setPheromone(int i, int j, double ph){
    pheromone[i][j] = ph;
}
public void setHeuristic(int i, int j, double h){
    choiceInfo[i][j] = h;
}
public void setDistance(int i, int j, int d){
    dist[i][j] = d;
}
public void setAlfa(double a){
    alfa = a;
}
public void setBeta(double b){
    beta = b;
}
public void setTau0(double tau){
    tau0 = tau;
}
public void setRo(double r){
    ro = r;
}
public void setXi(double x){
    xi = x;
}
public void setQ0(double q){
    q0 = q;
}
public double getAlfa(){
    return alfa;
}
public double getBeta(){
    return beta;
}
public double getRo(){
    return ro;
}
public double getXi(){
    return xi;
}
public double getTau0(){
    return tau0;
```

```
}
public double getQ0(){
   return q0;
}
public int[] getBestTour(){
   int[] bestTour = new int[noNodes+1];
   int bestTourLength = Integer.MAX_VALUE;
   int bestIdx = -1;
   for(int i = 0; i < noAnts; i++){
      if(ants[i].getTourLength() < bestTourLength){
         bestTourLength = ants[i].getTourLength();
         bestIdx = i;
      }
   }
   for(int j = 0; j <=noNodes; j++)
      bestTour[j] = ants[bestIdx].getTour(j);
   return bestTour;
}
public int[] getBestSoFarTour(){
   return bestSoFarTour;
}
public void updateBestSoFarTour(){
   if(getBestTourLength() < computeTourLength(bestSoFarTour)){
      bestSoFarTour = getBestTour();
   }
}
public int getBestTourLength(){
   return computeTourLength(getBestTour());
}
public void setIteration(int iter){
   iterations = iter;
}
public int getIteration(){
   return iterations;
}
public int getNoAnts(){
   return noAnts;
}
public void setNoAnts(int ants){
   noAnts = ants;
}
public int getNoNodes(){
   return noNodes;
}
public void setNoNodes(int nodes){
   noNodes = nodes;
}
public void initData(){
   int i,j;
   for(i = 0; i < noNodes; i++)
      for(j = 0; j < noNodes; j++){
         dist[i][j] = 0;
         pheromone[i][j] = 0.0;
         choiceInfo[i][j] = 0.0;
      }
}
public void initPheromones(){
   int i,j;
   tau0 = computePheromone0();

   for(i = 0; i < noNodes; i++)
      for(j = 0; j < noNodes; j++)
         pheromone[i][j] = tau0;
```

```
    for(i = 0; i < noNodes; i++)
        pheromone[i][i] = 0;

    opt = new OptimizationTSP(dist);
}
public void computeHeuristic(){
    double niu;
    int i,j;

    for(i = 0; i < noNodes; i++)
        for(j = 0; j < noNodes; j++){
            if(dist[i][j] > 0)
                niu = 1.0/dist[i][j];
            else
                niu = 1.0/0.0001;
        choiceInfo[i][j] = Math.pow(pheromone[i][j],alfa)*Math.pow(niu,beta);
    }
}
public void initAnts(){
    int i,j;
    for(i = 0; i < noAnts; i++){
        ants[i].setTourLength(0);

        for(j = 0; j < noNodes; j++)
            ants[i].setVisited(j, false);
        for(j = 0; j <= noNodes; j++)
            ants[i].setTour(j, 0);
    }
}
public void decisionRule(int k, int step){


    int c = ants[k].getTour(step-1);
    double sumProb = 0.0;

    double selectionProbability[] = new double[noNodes];

    int j;
    for(j = 0; j < noNodes; j++){
        if((ants[k].getVisited(j)) || (j == c))
            selectionProbability[j] = 0.0;
        else{
            selectionProbability[j] = choiceInfo[c][j];
            sumProb+=selectionProbability[j];
        }

    }
    double prob = Math.random()*sumProb;
    j = 0;
    double p = selectionProbability[j];
    while(p < prob){
        j++;
        p += selectionProbability[j];
    }

    int randomDecision = j;


    double maxHeuristic = -1;
    int maxHeuristicIdx = -1;
    for(j = 0; j < noNodes; j++){
        if(maxHeuristic < choiceInfo[c][j] && !(ants[k].getVisited(j))){
```

```
            maxHeuristic = choiceInfo[c][j];
            maxHeuristicIdx = j;
        }
    }

    if(Math.random() < q0){
        ants[k].setTour(step, maxHeuristicIdx);
        ants[k].setVisited(maxHeuristicIdx, true);
    }
    else{
        ants[k].setTour(step, randomDecision);
        ants[k].setVisited(randomDecision, true);
    }



}
public void constructSolutions(){
    /* stergere memorie furnici */
    initAnts();

    int step = 0;
    int k;
    int r;

    Random rand = new Random();

    for(k = 0; k < noAnts; k++){
        r = Math.abs(rand.nextInt())%noNodes;

        ants[k].setTour(step,r);
        ants[k].setVisited(r,true);
    }

    while(step < noNodes-1){
        step++;
        for(k = 0; k < noAnts; k++){
            decisionRule(k,step);
            localPheromoneUpdate(k,step);
        }
    }

    for(k = 0; k < noAnts; k++){
        ants[k].setTour(noNodes,ants[k].getTour(0));
        localPheromoneUpdate(k,noNodes);
        ants[k].setTourLength(computeTourLength(ants[k].getTour()));
    }
    updateBestSoFarTour();
}
public void globalEvaporation(){
    for(int i = 0; i <noNodes; i++){
        int idx1 = bestSoFarTour[i];
        int idx2 = bestSoFarTour[i+1];
        pheromone[idx1][idx2]*=(1-ro);
        pheromone[idx2][idx1]*=(1-ro);
    }

}
public void depositPheromone(int k){

}
public void globalPheromoneDeposit(){
    double delta = 1.0/((double) getBestTourLength());
```

```
    for(int i = 0; i <noNodes; i++){
        int idx1 = bestSoFarTour[i];
        int idx2 = bestSoFarTour[i+1];
        pheromone[idx1][idx2]+=ro*delta;
        pheromone[idx2][idx1]+=ro*delta;
    }
}
public void updatePheromones(){
    globalEvaporation();
    globalPheromoneDeposit();
    computeHeuristic();
}
public void localPheromoneUpdate(int ant, int step){
    int idx1 = ants[ant].getTour(step);
    int idx2 = ants[ant].getTour(step-1);
    double currentValue = pheromone[idx1][idx2];
    pheromone[idx1][idx2] = (1-xi)*currentValue+xi*tau0;
    pheromone[idx2][idx1] = pheromone[idx1][idx2];
    double niu = 0;
    if(dist[idx1][idx2] > 0)
        niu = 1.0/dist[idx1][idx2];
    else
        niu = 1.0/0.0001;
    choiceInfo[idx1][idx2] = Math.pow(pheromone[idx1][idx2],alfa)*Math.pow(niu,beta);
    choiceInfo[idx1][idx2] = choiceInfo[idx2][idx1];
}
private int greedyTour(){
    boolean visited[] = new boolean[noNodes];
    int tour[] = new int[noNodes+1];
    int length;
    int min, node;
    int i,j;

    for(i = 0; i < noNodes; i++)
        visited[i] = false;

    tour[0] = 0;
    bestSoFarTour[0] = 0;
    visited[0] = true;

    for(i = 1; i < noNodes; i++){
        min = Integer.MAX_VALUE;
        node = -1;
        for(j = 0; j < noNodes; j++){
            if((!visited[j])&&(j!=tour[i-1])){
                if(min > dist[tour[i-1]][j]){
                    min = dist[tour[i-1]][j];
                    node = j;
                }
            }
        }
        tour[i] = node;
        bestSoFarTour[i] = node;
        visited[node] = true;
    }
    tour[noNodes] = tour[0];
    bestSoFarTour[noNodes] = bestSoFarTour[0];
    return computeTourLength(tour);

}
public int computeTourLength(int tour[]){
    int len = 0;
    for(int i = 0; i < noNodes; i++){
```

```
            len+=dist[tour[i]][tour[i+1]];
        }
        return len;
    }
    private double computePheromone0(){
        return 1.0/(((double)greedyTour())*((double)noAnts));
    }
    public void opt2(){
        int i,j,k;
        int a1,a2,a3,b1,b2,b3,swap;
        for(k = 0; k < noAnts; k++){
            for(i = 1; i < noNodes-1; i++){
                a1 = dist[ants[k].getTour(i-1)][ants[k].getTour(i)];
                a2 = dist[ants[k].getTour(i)][ants[k].getTour(i+1)];
                a3 = dist[ants[k].getTour(i+1)][ants[k].getTour(i+2)];

                b1 = dist[ants[k].getTour(i-1)][ants[k].getTour(i+1)];
                b2 = dist[ants[k].getTour(i+1)][ants[k].getTour(i)];
                b3 = dist[ants[k].getTour(i)][ants[k].getTour(i+2)];

                if(a1+a2+a3 > b1+b2+b3){
                    swap = ants[k].getTour(i);
                    ants[k].setTour(i, ants[k].getTour(i+1));
                    ants[k].setTour(i+1, swap);
                }

            }

        }
    }
    public void localSearch(){
        if(doOpt2){
            for(int i = 0; i < noAnts; i++){
                opt.opt2(ants[i].getTour());
            }
        }
        if(doOpt3){
            for(int i = 0; i < noAnts; i++){
                opt.opt3(ants[i].getTour());
            }
        }
    }
    public double[][] getPheromoneMatrix(){
        return pheromone;
    }


}
```

Додаток Б.4. Головний клас запуску програми

```
public class GraphicalSolverFrame extends javax.swing.JFrame {
    private static final int STATE_NEW = 0;
    private static final int STATE_INIT = 1;
    private static final int STATE_RUNNING = 2;

    private static int currentState;

    private AntSystemTSP as;
    private MaxMinAntSystemTSP mmas;
    private AntColonySystemTSP acs;
    private GeneticAlgorithmTSP ga;
    private GeneticAntSystemTSP gas;
```

```
private ArrayList<PointCoords> points;

private int noNodes;
private int noAnts;
private int popSize;
private double alfa;
private double beta;
private double globalEvapRate;
private double localEvapRate;
private double pseudoRandCoef;
private double mutationRatio;
private boolean doOpt2;
private boolean doOpt3;



public GraphicalSolverFrame() {
    initComponents();

    points = new ArrayList<PointCoords>();
    currentState = STATE_NEW;
    jButton1.setEnabled(true);
    jButton2.setEnabled(true);
    jButton3.setEnabled(false);
    jButton4.setEnabled(false);
    jButton5.setEnabled(false);
    jButton6.setEnabled(true);

    jComboBox1.setEnabled(true);

}


@SuppressWarnings("unchecked")

private void initComponents() {

    jPanel1 = new javax.swing.JPanel();
    canvas1 = new java.awt.Canvas();
    jPanel2 = new javax.swing.JPanel();
    jComboBox1 = new javax.swing.JComboBox();
    jLabel1 = new javax.swing.JLabel();
    jCheckBox1 = new javax.swing.JCheckBox();
    jLabel2 = new javax.swing.JLabel();
    jSpinner1 = new javax.swing.JSpinner();
    jLabel3 = new javax.swing.JLabel();
    jTextField1 = new javax.swing.JTextField();
    jLabel4 = new javax.swing.JLabel();
    jTextField2 = new javax.swing.JTextField();
    jLabel5 = new javax.swing.JLabel();
    jTextField3 = new javax.swing.JTextField();
    jLabel6 = new javax.swing.JLabel();
    jTextField4 = new javax.swing.JTextField();
    jLabel7 = new javax.swing.JLabel();
    jTextField5 = new javax.swing.JTextField();
    jLabel8 = new javax.swing.JLabel();
    jLabel9 = new javax.swing.JLabel();
    jSpinner2 = new javax.swing.JSpinner();
    jLabel10 = new javax.swing.JLabel();
    jLabel11 = new javax.swing.JLabel();
    jTextField6 = new javax.swing.JTextField();
    jLabel12 = new javax.swing.JLabel();
```

```
        jCheckBox2 = new javax.swing.JCheckBox();
        jCheckBox3 = new javax.swing.JCheckBox();
        jPanel3 = new javax.swing.JPanel();
        jButton1 = new javax.swing.JButton();
        jButton2 = new javax.swing.JButton();
        jButton3 = new javax.swing.JButton();
        jButton4 = new javax.swing.JButton();
        jButton5 = new javax.swing.JButton();
        jButton6 = new javax.swing.JButton();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        setTitle("Graphical Traveling Salesman Problem Solver");

        jPanel1.setBorder(javax.swing.BorderFactory.createTitledBorder(new
javax.swing.border.SoftBevelBorder(javax.swing.border.BevelBorder.RAISED), "Input"));

        canvas1.setBackground(new java.awt.Color(255, 255, 255));
        canvas1.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                canvas1MouseClicked(evt);
            }
        });

        javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
        jPanel1.setLayout(jPanel1Layout);
        jPanel1Layout.setHorizontalGroup(
            jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addContainerGap()
                .addComponent(canvas1, javax.swing.GroupLayout.PREFERRED_SIZE, 680,
javax.swing.GroupLayout.PREFERRED_SIZE)
                .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
        );
        jPanel1Layout.setVerticalGroup(
            jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addComponent(canvas1, javax.swing.GroupLayout.DEFAULT_SIZE, 544, Short.MAX_VALUE)
                .addContainerGap())
        );

        jPanel2.setBorder(javax.swing.BorderFactory.createTitledBorder(new
javax.swing.border.SoftBevelBorder(javax.swing.border.BevelBorder.RAISED), "Options",
javax.swing.border.TitledBorder.DEFAULT_JUSTIFICATION,
javax.swing.border.TitledBorder.DEFAULT_POSITION, new java.awt.Font("Tahoma", 1, 11)));

        jComboBox1.setModel(new javax.swing.DefaultComboBoxModel(new String[] { "Ant System", "Ant
Colony System", "MaxMin Ant System", "Genetic Algorithm", "Genetic Ant System" }));

        jLabel1.setText("Method:");

        jCheckBox1.setSelected(true);
        jCheckBox1.setText("Show Best So Far Tour");

        jLabel2.setFont(new java.awt.Font("Tahoma", 1, 11));
        jLabel2.setText("Ant Colony options:");

        jSpinner1.setValue(10);

        jLabel3.setText("Number of Ants:");

        jTextField1.setText("1.0");

        jLabel4.setText("Alfa:");
```

```
        jTextField2.setText("3.0");

        jLabel5.setText("Beta:");

        jTextField3.setText("0.5");

        jLabel6.setText("Global evap. Rate:");

        jTextField4.setText("0.1");

        jLabel7.setText("Local evap. Rate:");

        jTextField5.setText("0.7");

        jLabel8.setText("Pseudo-random coeficient:");

        jLabel9.setFont(new java.awt.Font("Tahoma", 1, 11));
jLabel9.setText("Genetic Algorithm options:");

        jSpinner2.setValue(10);

        jLabel10.setText("Population Size:");

        jLabel11.setText("Mutation Ratio:");

        jTextField6.setText("0.1");

        jLabel12.setFont(new java.awt.Font("Tahoma", 1, 11));
        jLabel12.setText("Optimization options:");

        jCheckBox2.setText("2-opt");

        jCheckBox3.setText("3-opt");

        javax.swing.GroupLayout jPanel2Layout = new javax.swing.GroupLayout(jPanel2);
        jPanel2.setLayout(jPanel2Layout);
        jPanel2Layout.setHorizontalGroup(
            jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel2Layout.createSequentialGroup()
                .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addGroup(jPanel2Layout.createSequentialGroup()
                        .addContainerGap()
                        .addComponent(jLabel1)
                        .addGap(18, 18, 18)
                        .addComponent(jComboBox1, 0, 123, Short.MAX_VALUE))
                    .addComponent(jCheckBox1)
                    .addComponent(jLabel2)
                    .addComponent(jLabel9)
                    .addComponent(jLabel12)
                    .addGroup(jPanel2Layout.createSequentialGroup()
                        .addComponent(jCheckBox2)
                        .addGap(18, 18, 18)
                        .addComponent(jCheckBox3))
                    .addGroup(jPanel2Layout.createSequentialGroup()
                        .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                            .addComponent(jLabel5)
                            .addComponent(jLabel4)
                            .addComponent(jLabel3)
                            .addComponent(jLabel6)
                            .addComponent(jLabel7)
                            .addComponent(jLabel8)
                            .addComponent(jLabel10)
```

```
                    .addComponent(jLabel11))
                .addGap(4, 4, 4)
                .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(jTextField6, javax.swing.GroupLayout.DEFAULT_SIZE, 59,
Short.MAX_VALUE)
                    .addComponent(jTextField5, javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.DEFAULT_SIZE, 59, Short.MAX_VALUE)
                    .addComponent(jTextField4, javax.swing.GroupLayout.DEFAULT_SIZE, 59,
Short.MAX_VALUE)
                    .addComponent(jTextField3, javax.swing.GroupLayout.DEFAULT_SIZE, 59,
Short.MAX_VALUE)
                    .addComponent(jTextField1, javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.DEFAULT_SIZE, 59, Short.MAX_VALUE)
                    .addComponent(jTextField2, javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.DEFAULT_SIZE, 59, Short.MAX_VALUE)
                    .addComponent(jSpinner1, javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.DEFAULT_SIZE, 59, Short.MAX_VALUE)
                    .addComponent(jSpinner2, javax.swing.GroupLayout.DEFAULT_SIZE, 59,
Short.MAX_VALUE))))
            .addContainerGap())
        );
        jPanel2Layout.setVerticalGroup(
            jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jPanel2Layout.createSequentialGroup()
                .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(jComboBox1, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jLabel1))
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jCheckBox1)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addComponent(jLabel2)
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(jSpinner1, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jLabel3))
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(jTextField1, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jLabel4))
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(jTextField2, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jLabel5))
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(jTextField3, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jLabel6))
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(jTextField4, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jLabel7))
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
                    .addComponent(jTextField5, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                    .addComponent(jLabel8))
                .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
```

```
            .addComponent(jLabel9)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
               .addComponent(jSpinner2, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
               .addComponent(jLabel10))
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
               .addComponent(jLabel11)
               .addComponent(jTextField6, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(jLabel12)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addGroup(jPanel2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
               .addComponent(jCheckBox2)
               .addComponent(jCheckBox3))
            .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
      );

      jPanel3.setBorder(javax.swing.BorderFactory.createTitledBorder(new
javax.swing.border.SoftBevelBorder(javax.swing.border.BevelBorder.RAISED), "Controls",
javax.swing.border.TitledBorder.DEFAULT_JUSTIFICATION,
javax.swing.border.TitledBorder.DEFAULT_POSITION, new java.awt.Font("Tahoma", 1, 11)));

      jButton1.setText("New");
      jButton1.addActionListener(new java.awt.event.ActionListener() {
         public void actionPerformed(java.awt.event.ActionEvent evt) {
            jButton1ActionPerformed(evt);
         }
      });

      jButton2.setText("Init");
      jButton2.addActionListener(new java.awt.event.ActionListener() {
         public void actionPerformed(java.awt.event.ActionEvent evt) {
            jButton2ActionPerformed(evt);
         }
      });

      jButton3.setText("Step");
      jButton3.addActionListener(new java.awt.event.ActionListener() {
         public void actionPerformed(java.awt.event.ActionEvent evt) {
            jButton3ActionPerformed(evt);
         }
      });

      jButton4.setText("Start");
      jButton4.addActionListener(new java.awt.event.ActionListener() {
         public void actionPerformed(java.awt.event.ActionEvent evt) {
            jButton4ActionPerformed(evt);
         }
      });

      jButton5.setText("Stop");
      jButton5.addActionListener(new java.awt.event.ActionListener() {
         public void actionPerformed(java.awt.event.ActionEvent evt) {
            jButton5ActionPerformed(evt);
         }
      });

      jButton6.setText("Reset");
      jButton6.addActionListener(new java.awt.event.ActionListener() {
         public void actionPerformed(java.awt.event.ActionEvent evt) {
```

```
            jButton6ActionPerformed(evt);
        }
    });

    javax.swing.GroupLayout jPanel3Layout = new javax.swing.GroupLayout(jPanel3);
    jPanel3.setLayout(jPanel3Layout);
    jPanel3Layout.setHorizontalGroup(
        jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(jPanel3Layout.createSequentialGroup()
            .addGap(35, 35, 35)
            .addGroup(jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING,
false)
                .addComponent(jButton5, javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)
                .addComponent(jButton4, javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)
                .addComponent(jButton3, javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)
                .addComponent(jButton6, javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)
                .addComponent(jButton2, javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
Short.MAX_VALUE)
                .addComponent(jButton1, javax.swing.GroupLayout.Alignment.LEADING,
javax.swing.GroupLayout.DEFAULT_SIZE, 130, Short.MAX_VALUE))
            .addContainerGap(36, Short.MAX_VALUE))
    );
    jPanel3Layout.setVerticalGroup(
        jPanel3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(jPanel3Layout.createSequentialGroup()
            .addComponent(jButton1)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(jButton2)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
            .addComponent(jButton6)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(jButton3)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(jButton4)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addComponent(jButton5)
            .addContainerGap())
    );

    javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
    getContentPane().setLayout(layout);
    layout.setHorizontalGroup(
        layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(layout.createSequentialGroup()
            .addContainerGap()
            .addComponent(jPanel1, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
            .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
            .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addComponent(jPanel3, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                .addComponent(jPanel2, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
```

```
            .addContainerGap())
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(layout.createSequentialGroup()
                .addContainerGap()
                .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                    .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
                    .addGroup(layout.createSequentialGroup()
                        .addComponent(jPanel2, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                        .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
                        .addComponent(jPanel3, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)))
                .addContainerGap())
        );

        pack();
    }

    private void canvas1MouseClicked(java.awt.event.MouseEvent evt) {
        if(currentState == STATE_NEW){
            int coordX = evt.getX();
            int coordY = evt.getY();
            points.add(new PointCoords(coordX, coordY));
            drawCanvas();
        }
        else if ((currentState == STATE_INIT)||(currentState == STATE_RUNNING)){
            if(((String)jComboBox1.getSelectedItem()).equals("Ant System")){

            }
        }
    }/

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        currentState = STATE_NEW;

        jButton1.setEnabled(true);
        jButton2.setEnabled(true);
        jButton3.setEnabled(false);
        jButton4.setEnabled(false);
        jButton5.setEnabled(false);
        jButton6.setEnabled(true);

        jTextField1.setEnabled(true);
        jTextField2.setEnabled(true);
        jTextField3.setEnabled(true);
        jTextField4.setEnabled(true);
        jTextField5.setEnabled(true);
        jTextField6.setEnabled(true);

        jSpinner1.setEnabled(true);
        jSpinner2.setEnabled(true);

        jCheckBox2.setEnabled(true);
        jCheckBox3.setEnabled(true);

        jComboBox1.setEnabled(true);

        points.clear();
        drawCanvas();
```

```
    }

    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
        currentState = STATE_INIT;

        jButton1.setEnabled(true);
        jButton2.setEnabled(false);
        jButton3.setEnabled(true);
        jButton4.setEnabled(true);
        jButton5.setEnabled(false);
        jButton6.setEnabled(true);


        jTextField1.setEnabled(false);
        jTextField2.setEnabled(false);
        jTextField3.setEnabled(false);
        jTextField4.setEnabled(false);
        jTextField5.setEnabled(false);
        jTextField6.setEnabled(false);

        jSpinner1.setEnabled(false);
        jSpinner2.setEnabled(false);

        jCheckBox2.setEnabled(false);
        jCheckBox3.setEnabled(false);

        jComboBox1.setEnabled(false);

        noAnts = ((Integer)jSpinner1.getValue()).intValue();
        popSize = ((Integer)jSpinner2.getValue()).intValue();
        noNodes = points.size();
        try{
            alfa = Double.parseDouble(jTextField1.getText());
            beta = Double.parseDouble(jTextField2.getText());
            globalEvapRate = Double.parseDouble(jTextField3.getText());
            localEvapRate = Double.parseDouble(jTextField4.getText());
            pseudoRandCoef = Double.parseDouble(jTextField5.getText());
            mutationRatio = Double.parseDouble(jTextField6.getText());
        }
        catch(Exception e){
            System.err.println("Wrong params.");
        }
        doOpt2 = jCheckBox2.isSelected();
        doOpt3 = jCheckBox3.isSelected();

        as = new AntSystemTSP(noNodes, noAnts, alfa, beta, globalEvapRate, doOpt2, doOpt3);
        mmas = new MaxMinAntSystemTSP(noNodes, noAnts, alfa, beta, globalEvapRate, doOpt2, doOpt3,
1000);
        acs = new AntColonySystemTSP(noNodes, noAnts, alfa, beta, globalEvapRate, localEvapRate,
pseudoRandCoef, doOpt2, doOpt3);
        ga = new GeneticAlgorithmTSP(popSize, noNodes, mutationRatio, doOpt2, doOpt3);
        gas = new GeneticAntSystemTSP(noNodes, noAnts, alfa, beta, globalEvapRate, doOpt2, doOpt3, 1000,
10);

        as.initData();
        mmas.initData();
        acs.initData();
        gas.initData();
        for(int i = 0; i < noNodes; i++){
            for(int j = 0; j < noNodes; j++){
                if(i == j){
```

```
                as.setDistance(i, j, 0);
                mmas.setDistance(i, j, 0);
                acs.setDistance(i, j, 0);
                ga.setDistance(i, j, 0);
                gas.setDistance(i, j, 0);
            }
            else{
                as.setDistance(i, j, points.get(i).getIntegerDistance(points.get(j)));

                System.out.println(points.get(i).getX()+":"+points.get(i).getY());
                mmas.setDistance(i, j, points.get(i).getIntegerDistance(points.get(j)));
                acs.setDistance(i, j, points.get(i).getIntegerDistance(points.get(j)));
                ga.setDistance(i, j, points.get(i).getIntegerDistance(points.get(j)));
                gas.setDistance(i, j, points.get(i).getIntegerDistance(points.get(j)));



            }
        }
    }

    as.initPheromones();
    as.computeHeuristic();
    as.initAnts();

    mmas.initPheromones();
    mmas.computeHeuristic();
    mmas.initAnts();

    acs.initPheromones();
    acs.computeHeuristic();
    acs.initAnts();

    ga.initPopulation();

    gas.initPheromones();
    gas.computeHeuristic();
    gas.initAnts();

    drawCanvas();


}

private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
    step();
}

private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {
    currentState = STATE_RUNNING;

    jButton1.setEnabled(false);
    jButton2.setEnabled(false);
    jButton3.setEnabled(false);
    jButton4.setEnabled(false);
    jButton5.setEnabled(true);
    jButton6.setEnabled(false);
    jComboBox1.setEnabled(false);


    RunThread thread = new RunThread();
    thread.start();
```

```
    }
    private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {
        currentState = STATE_INIT;

        jButton1.setEnabled(true);
        jButton2.setEnabled(false);
        jButton3.setEnabled(true);
        jButton4.setEnabled(true);
        jButton5.setEnabled(false);
        jButton6.setEnabled(true);

        jComboBox1.setEnabled(false);
}

    private void jButton6ActionPerformed(java.awt.event.ActionEvent evt) {

        currentState = STATE_NEW;

        jButton1.setEnabled(true);
        jButton2.setEnabled(true);
        jButton3.setEnabled(false);
        jButton4.setEnabled(false);
        jButton5.setEnabled(false);
        jButton6.setEnabled(true);

        jTextField1.setEnabled(true);
        jTextField2.setEnabled(true);
        jTextField3.setEnabled(true);
        jTextField4.setEnabled(true);
        jTextField5.setEnabled(true);
        jTextField6.setEnabled(true);

        jSpinner1.setEnabled(true);
        jSpinner2.setEnabled(true);

        jCheckBox2.setEnabled(true);
        jCheckBox3.setEnabled(true);

        jComboBox1.setEnabled(true);

        drawCanvas();
    }
    private void drawBestTour(int[] tour, Graphics2D gmem){
        float dash1[] = {10.0f, 3.0f, 5.0f};
        BasicStroke dashed = new BasicStroke(5.0f,
                                BasicStroke.CAP_BUTT,
                                BasicStroke.JOIN_MITER,
                                10.0f, dash1, 0.0f);


        gmem.setStroke(dashed);
        gmem.setColor(Color.blue);
        for(int i = 0; i < tour.length-1; i++){
            gmem.drawLine(points.get(tour[i]).getX(), points.get(tour[i]).getY(),
                  points.get(tour[i+1]).getX(), points.get(tour[i+1]).getY());
        }

    }
    private void drawGraphPlain(Graphics2D gmem){
        for(PointCoords p: points){
            gmem.fillOval(p.getX()-5, p.getY()-5, 10, 10);
        }
```

```
    for(int i = 0; i < points.size()-1; i++){
        for(int j = i+1; j < points.size(); j++){
            gmem.drawLine(points.get(i).getX(), points.get(i).getY(),points.get(j).getX(), points.get(j).getY());
        }
    }
}
private void drawGraphPheromone(Graphics2D gmem, double[][] ph){
    int noIntervals = 5;
    double maxPh = Double.NEGATIVE_INFINITY;
    double minPh = Double.POSITIVE_INFINITY;
    for(int i = 0; i < noNodes; i++){
        for(int j = 0; j < noNodes; j++){
            if(i!=j){
                if(maxPh < ph[i][j])
                    maxPh = ph[i][j];
                if(minPh > ph[i][j])
                    minPh = ph[i][j];
            }
        }
    }

    double delta = (maxPh-minPh)/((double)noIntervals);

    for(int i = 0; i < noNodes-1; i++)
        for(int j = i+1; j < noNodes; j++){
            double phij = ph[i][j];
            if(delta>0.0){
                int intv = (int)((phij-minPh)/delta);
                gmem.setColor(new Color((intv)*(255/noIntervals),0,0));
                gmem.setStroke(new BasicStroke(1.0f+(float)((float)intv/(float)noIntervals)*2.0f));
            }
            else{
                gmem.setColor(Color.RED);
            }
            gmem.drawLine(points.get(i).getX(),points.get(i).getY(),points.get(j).getX(),points.get(j).getY());
        }
    gmem.setColor(Color.BLACK);

    for(int i = 0; i < noNodes; i++)
        gmem.fillOval(points.get(i).getX()-5, points.get(i).getY()-5, 10, 10);
}
private void drawCanvas(){
    int i,j;
    Graphics g = (Graphics2D)canvas1.getGraphics();
    Dimension d = canvas1.getSize();
    Image img = canvas1.createImage(d.width,d.height);
    Graphics2D gmem = (Graphics2D)img.getGraphics();



    if(currentState == STATE_NEW){
        drawGraphPlain(gmem);
    }
    else if((currentState == STATE_INIT)||(currentState == STATE_RUNNING)){
        if(((String)jComboBox1.getSelectedItem()).equals("Ant System")){

            drawGraphPheromone(gmem, as.getPheromoneMatrix());
            if(jCheckBox1.isSelected()){
                drawBestTour(as.getBestSoFarTour(), gmem);
            }
            gmem.setColor(Color.black);
            gmem.drawString("Iterations: "+as.getIteration(), 10, 10);
```

```
            gmem.drawString("Tour Length: "+as.computeTourLength(as.getBestSoFarTour()), 10, 20);
        }
        if(((String)jComboBox1.getSelectedItem()).equals("MaxMin Ant System")){

            drawGraphPheromone(gmem, mmas.getPheromoneMatrix());
            if(jCheckBox1.isSelected()){
                drawBestTour(mmas.getBestSoFarTour(), gmem);
            }
            gmem.setColor(Color.black);
            gmem.drawString("Iterations: "+mmas.getIteration(), 10, 10);
            gmem.drawString("Tour Length: "+mmas.computeTourLength(mmas.getBestSoFarTour()), 10,
20);
        }
        if(((String)jComboBox1.getSelectedItem()).equals("Ant Colony System")){

            drawGraphPheromone(gmem, acs.getPheromoneMatrix());
            if(jCheckBox1.isSelected()){
                drawBestTour(acs.getBestSoFarTour(), gmem);
            }
            gmem.setColor(Color.black);
            gmem.drawString("Iterations: "+acs.getIteration(), 10, 10);
            gmem.drawString("Tour Length: "+acs.computeTourLength(acs.getBestSoFarTour()), 10, 20);
        }
        if(((String)jComboBox1.getSelectedItem()).equals("Genetic Algorithm")){

            drawGraphPlain(gmem);
            if(jCheckBox1.isSelected()){
                drawBestTour(ga.getBestSoFarTour(), gmem);
            }
            gmem.setColor(Color.black);
            gmem.drawString("Iterations: "+ga.getIteration(), 10, 10);
            gmem.drawString("Tour Length: "+ga.computeTourLength(ga.getBestSoFarTour()), 10, 20);
        }
        if(((String)jComboBox1.getSelectedItem()).equals("Genetic Ant System")){

            drawGraphPheromone(gmem, gas.getPheromoneMatrix());
            if(jCheckBox1.isSelected()){
                drawBestTour(gas.getBestSoFarTour(), gmem);
            }
            gmem.setColor(Color.black);
            gmem.drawString("Iterations: "+gas.getIteration(), 10, 10);
            gmem.drawString("Tour Length: "+gas.computeTourLength(gas.getBestSoFarTour()), 10, 20);
        }
    }
    g.drawImage(img, 0, 0, canvas1);

    gmem.dispose();
}

private void step(){
    if(((String)jComboBox1.getSelectedItem()).equals("Ant System")){

        as.constructSolutions();
        as.localSearch();
        as.updatePheromones();
        as.setIteration(as.getIteration()+1);
    }
    if(((String)jComboBox1.getSelectedItem()).equals("MaxMin Ant System")){

        mmas.constructSolutions();
        mmas.localSearch();
        mmas.updatePheromones();
        mmas.setIteration(mmas.getIteration()+1);
```

```
    }
    if(((String)jComboBox1.getSelectedItem()).equals("Ant Colony System")){

        acs.constructSolutions();
        acs.localSearch();
        acs.updatePheromones();
        acs.setIteration(acs.getIteration()+1);
    }
    if(((String)jComboBox1.getSelectedItem()).equals("Genetic Ant System")){

        gas.constructSolutions();
        gas.localSearch();
        gas.updatePheromones();
        gas.setIteration(gas.getIteration()+1);
    }
    if(((String)jComboBox1.getSelectedItem()).equals("Genetic Algorithm")){

        ga.stepGeneration();
        ga.localSearch();
        ga.setIteration(ga.getIteration()+1);
    }
    drawCanvas();
}

public static void main(String args[]) {
    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new GraphicalSolverFrame().setVisible(true);
        }
    });
}
class RunThread extends Thread{
    @Override
    public void run(){
        while(currentState == STATE_RUNNING){
            step();
            try{
                sleep(50);
            }
            catch(Exception e){
                System.err.println("Thread sleep error.");
            }
        }
    }
}

/
private java.awt.Canvas canvas1;
private javax.swing.JButton jButton1;
private javax.swing.JButton jButton2;
private javax.swing.JButton jButton3;
private javax.swing.JButton jButton4;
private javax.swing.JButton jButton5;
private javax.swing.JButton jButton6;
private javax.swing.JCheckBox jCheckBox1;
private javax.swing.JCheckBox jCheckBox2;
private javax.swing.JCheckBox jCheckBox3;
private javax.swing.JComboBox jComboBox1;
private javax.swing.JLabel jLabel1;
private javax.swing.JLabel jLabel10;
private javax.swing.JLabel jLabel11;
private javax.swing.JLabel jLabel12;
private javax.swing.JLabel jLabel2;
```

```
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JLabel jLabel5;
    private javax.swing.JLabel jLabel6;
    private javax.swing.JLabel jLabel7;
    private javax.swing.JLabel jLabel8;
    private javax.swing.JLabel jLabel9;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JPanel jPanel2;
    private javax.swing.JPanel jPanel3;
    private javax.swing.JSpinner jSpinner1;
    private javax.swing.JSpinner jSpinner2;
    private javax.swing.JTextField jTextField1;
    private javax.swing.JTextField jTextField2;
    private javax.swing.JTextField jTextField3;
    private javax.swing.JTextField jTextField4;
    private javax.swing.JTextField jTextField5;
    private javax.swing.JTextField jTextField6;

}
```

ДОДАТОК В

Відомість атестаційної роботи

Позначення	Найменування	Дод. відомості
	Текстові документи	
1.	Пояснювальна записка	__с.
	Інші документи	
2.	Презентаційні матеріали	__ плакатів

.	.	Прізвище та ініціали.	Підп.	Дата	
Розробив		Авдусь А.В.			Дослідження та використання алгоритмів еволюційного моделювання в інтелектуальних системах прийняття рішень
Перевірив		Шевченко О.Ю.			
Н.контр.					
Затв.		Філатов В.О.			

Шифр групи	Код напр./спец.
СШІмзд-18-1	122
ХНУРЕ кафедра ШІ	