Kpi-best

МІНІСТЕРСТВО НАУКИ І ОСВІТИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»

Кафедра обчислювальної техніки

Лабораторна робота №3,4

з Теорії Проектування Комп’ютерних Систем-1

Виконав студент групи ІО-11

Ротенберг О.В

Київ — 2014

***Варіант***

***Лабораторна Робота №3***

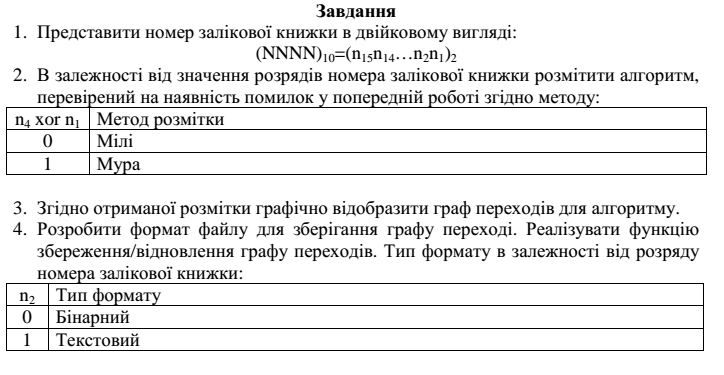
***Тема:***

Автоматизація розмітки блок-схем алгоритмів.

***Мета роботи:***

Здобуття навичок з автоматизації процедури розмітки алгоритмів за

методами Мілі та Мура.



***Лабораторна Робота №4***

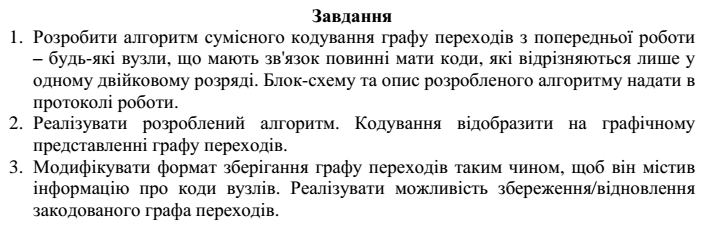
***Тема:***

Автоматизація розмітки блок-схем алгоритмів.

***Мета роботи:***

Здобуття навичок з автоматизації процедури сумісного кодування графу

переходів.



Лістинг:

**public** **class** NewMealyFrame **extends** JFrame {

**private** JMenuBar menuBar;

**private** List<String[]> listOfWays;

**private** Mediator mediator;

**private** TransitionData transitionData;

**private** **int** counter;

**public** NewMealyFrame(List<String[]> list, Mediator mediator1, **int** counter, TransitionData transitionData1) {

listOfWays = list;

**this**.counter = counter;

mediator = mediator1;

transitionData = transitionData1;

BinaryMarkerCreator binaryMarkerCreator = **new** BinaryMarkerCreator(listOfWays, counter);

binaryMarkerCreator.useCreateMarking();

List<TreeVertex> listOfMarked = binaryMarkerCreator.getTreeVertexList();

Toolkit kit = Toolkit.*getDefaultToolkit*();

Dimension screenSize = kit.getScreenSize();

setBounds(screenSize.width / 8, screenSize.height / 8, 6 \* screenSize.width / 10, 6 \* screenSize.height / 10);

setTitle("Mealy automate");

menuBar = **new** JMenuBar();

JMenu menu = **new** JMenu("File");

menuBar.add(menu);

addMenuItem(menu,"Save Graph", 6);

menuBar.setEnabled(**true**);

setJMenuBar(menuBar);

mxGraph graph = **new** mxGraph();

Object parent = graph.getDefaultParent();

graph.getModel().beginUpdate();

**try** {

mxStylesheet stylesheet = graph.getStylesheet();

Hashtable<String, Object> style = **new** Hashtable<String, Object>();

style.put(mxConstants.*STYLE\_SHAPE*, mxConstants.***SHAPE\_ELLIPSE***);

// style.put(mxConstants.STYLE\_EDGE, mxConstants.SHAPE\_CONNECTOR);

style.put(mxConstants.*STYLE\_OPACITY*, 100);

style.put(mxConstants.*STYLE\_FONTCOLOR*, "#774400");

stylesheet.putCellStyle("ROUNDED", style);

Hashtable<String, Object> style2 = **new** Hashtable<String, Object>();

// style2.put(mxConstants.STYLE\_SHAPE, mxConstants.SHAPE\_TRIANGLE);

style2.put(mxConstants.*STYLE\_EDGE*, mxConstants.***EDGESTYLE\_TOPTOBOTTOM***);

style2.put(mxConstants.*STYLE\_OPACITY*, 100);

style2.put(mxConstants.*STYLE\_FONTCOLOR*, "#774400");

stylesheet.putCellStyle("ARROW", style2);

ArrayList<Object> vertexes = **new** ArrayList<Object>();

ArrayList<Object> edges = **new** ArrayList<Object>();

**for** (**int** i = 1; i<=counter; i++){

Object vertex = graph.insertVertex(parent, **null**, "Z"+i+"\n"+Integer.*toBinaryString*(listOfMarked.get(i - 1).getBinaryMarker()), 30, 30, 40, 40, "ROUNDED");

vertexes.add(vertex);

}

**for** (**int** i = 0; i<listOfWays.size(); i++){

String[] way = listOfWays.get(i);

**int** begin = Integer.*valueOf*(way[0].substring(1));

**int** end = Integer.*valueOf*(way[3].substring(1));

String passageValue = way[1]+"/"+way[2];

Object edge = graph.insertEdge(parent, **null**, passageValue, vertexes.get(begin-1), vertexes.get(end-1),"ARROW");

edges.add(edge);

}

} **finally** {

graph.getModel().endUpdate();

}

// graph.setCellsDisconnectable(false);

graph.setCellsEditable(**false**);

graph.setCellsResizable(**false**);

graph.setCellsBendable(**false**);

graph.setConnectableEdges(**false**);

// getContentPane().add(graphLayout);

// mxGraphComponent graphComponent = new mxGraphComponent(graphLayout);

mxGraphComponent graphComponent = **new** mxGraphComponent(graph);

getContentPane().add(graphComponent);

mxGraphLayout graphLayout = **new** mxCircleLayout(graph);

// graphLayout.setEdgeStyleEnabled(graph,true);

graphLayout.execute(parent);

// mxParallelEdgeLayout edgeLayout = new mxParallelEdgeLayout(graph,10);

// edgeLayout.execute(parent);

}

**public** Action createActionForItem(String menuName, **final** **int** typeOfMessage){

**return** **new** AbstractAction(menuName) {

**public** **void** actionPerformed(**final** ActionEvent event) {

Message message = **new** Message(typeOfMessage);

mediator.update(message);

}

};

}

**public** **void** addMenuItem(JMenu m, String menuName, **final** **int** typeOfMessage){

Action action = **new** AbstractAction(menuName) {

**public** **void** actionPerformed(**final** ActionEvent event) {

Message message = **new** Message(typeOfMessage);

mediator.update(message);

}

};

JMenuItem item = **new** JMenuItem(action);

m.add(item);

}

}