**МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ**

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ**

**«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»**

*Факультет прикладної математики*

*Кафедра «Спеціалізовані комп’ютерні системи»*

Лабораторна робота №3

з

Операційних систем

**Виконав:** Січкаренко В.О. гр. КВ-81

м. Київ

2011

*Завдання для лабораторної роботи*

**Вариант 1**

***Разработать модель алгоритма управления памятью ( без использования***

***внешней памяти) фиксированными разделами. Количество разделов должно быть меньше числа процессов.***

***Процессы образуют общую очередь к разделам памяти. Структуризация адресного пространства – линейная. Размеры процессов задаются случайно. Продемонстрировать процесс преобразования заданного виртуального адреса в физический.***

*Текст програми:*

--------Manager.java-------

/\*

\* To change this template, choose Tools | Templates

\* and open the template in the editor.

\*/

package os\_3;

import java.awt.List;

import java.util.ArrayList;

//import java.util.List;

/\*\*

\*

\* @author SichikUA

\*/

public class Manager {

private ArrayList<Process> ram;

//private List l;

// private List<Process> ram;

private int flag;

private PrQueue pq;

private myGui mg;

public Manager(PrQueue pq, myGui mg) {

flag = 1;

ram = new ArrayList<Process>(5);

for (int i = 0 ; i < 5; i++) {

ram.add(null);

}

this.pq = pq;

this.mg = mg;

}

public void myManager() {

Process bufProc;

System.out.println(ram.size());

while ( true ) {

System.out.println("fuck");

if ( flag == 1 ) {

System.out.println("fuck1");

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

if ( ram.get(i)== null) {

System.out.println("fuck2");

bufProc = pq.getFromQueue();

if ( bufProc == null) {

System.out.println("End program");

return;

}

ram.set(i, bufProc );

RunProc rp = new RunProc(bufProc.getTime(), ram, i, mg);

Thread t = new Thread(rp);

Thread.yield();

// try{

//Thread.sleep(500);

//} catch (InterruptedException e) {

//}

mg.setProgBar(i, bufProc.getSize());

t.start();

Thread.yield();

}

}

}

}

}

}

--------Process.java-------

/\*

\* To change this template, choose Tools | Templates

\* and open the template in the editor.

\*/

package os\_3;

import java.util.\*;

/\*\*

\*

\* @author SichikUA

\*/

public class Process {

private int time;

private int size;

public Process(int time, int size){

this.time = time;

this.size = size;

}

public int getTime(){

return this.time;

}

public int getSize() {

return this.size;

}

}

class PrQueue {

private Queue<Process> queue;

public PrQueue() {

queue = new LinkedList<Process>();

}

public void addToQueue(Process pr) {

queue.offer(pr);

}

public Process getFromQueue() {

return queue.poll();

}

public Process getFromQueueNotDel() {

return queue.peek();

}

}

--------RunProc.java-------

/\*

\* To change this template, choose Tools | Templates

\* and open the template in the editor.

\*/

package os\_3;

import java.util.ArrayList;

/\*\*

\*

\* @author SichikUA

\*/

public class RunProc implements Runnable {

private int time;

private ArrayList<Process> que;

private int num;

private myGui mg;

RunProc(int time, ArrayList<Process> que, int num, myGui mg) {

this.time = time;

this.que = que;

this.num = num;

this.mg = mg;

}

/\* private void myRun( int time, ArrayList<Process> que, int num) {

for (int i = 0; i < 100 \* time; i ++ ) {

Thread.yield();

}

que.add(num, null);

}\*/

public void run() {

System.out.println("Process run");

//mg.setProgBar(num, que.get(num).getSize());

for (int i = 0; i < 10000 \* time; i ++ ) {

Thread.yield();

}

System.out.println("End Proc ");

que.set(num, null);

}

}

--------MyGui.java-------

/\*

\* To change this template, choose Tools | Templates

\* and open the template in the editor.

\*/

package os\_3;

import java.awt.CardLayout;

import java.awt.Color;

import java.awt.Dimension;

import java.awt.FlowLayout;

import java.awt.GridLayout;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import javax.swing.JButton;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JPanel;

import javax.swing.JProgressBar;

import javax.swing.JScrollBar;

import javax.swing.JScrollPane;

import javax.swing.JSlider;

import javax.swing.JTable;

import javax.swing.JTextField;

import javax.swing.table.DefaultTableModel;

import javax.swing.table.TableCellEditor;

/\*\*

\*

\* @author SichikUA

\*/

public class myGui {

JFrame myFrame;

JPanel myPanel1, myPanel2, myPanel3,myPanel4, myPanel5, myPanel6, myPanel7,

myPanel8, myPanel9, myPanel10 ;

JLabel myLabel1, myLabel2, myLabel3;

JButton myButton1, myButton2;

JTextField myTF;

JSlider mS;

JTable myTable;

JScrollPane mySP;

JProgressBar myProgBar1, myProgBar2, myProgBar3,myProgBar4, myProgBar5;

private int column;

String str, str1;

PrQueue myQ;

Manager myManag;

public myGui() {

myQ = new PrQueue();

myManag = new Manager(myQ,this);

column = 19;

str = new String();

str1 = new String();

myFrame = new JFrame();

myPanel1 = new JPanel();

myPanel2 = new JPanel();

myPanel3 = new JPanel();

myPanel4 = new JPanel(new FlowLayout(FlowLayout.LEFT));

myPanel5 = new JPanel(new FlowLayout(FlowLayout.LEFT));

myPanel6 = new JPanel(new FlowLayout(FlowLayout.LEFT));

myPanel7 = new JPanel(new FlowLayout(FlowLayout.LEFT));

DefaultTableModel myModel = new DefaultTableModel();

myTable = new JTable(myModel);

myModel.setColumnCount(20);

myModel.setRowCount(1);

myTable.setAutoResizeMode(JTable.AUTO\_RESIZE\_OFF);

for (int i = 0; i < myModel.getColumnCount(); i++)

myTable.getColumnModel().getColumn(i).setPreferredWidth(50);

myTable.setTableHeader(null);

// myTable.setSize(new Dimension(900,40));

// myTable.setPreferredSize(new Dimension(100, 20));

mySP = new JScrollPane(myTable);

//mySP.setPreferredSize(new Dimension(200, 50));

myButton1 = new JButton("Add");

myButton1.setSize(100, 400);

myButton2 = new JButton("Start");

myLabel1 = new JLabel(" \*\*ADD PROCESS\*\*");

myLabel2 = new JLabel(" Process Time:");

myTF = new JTextField("");

mS = new JSlider(0,8, 4);

mS.setPreferredSize(new Dimension(140, 43));

mS.setPaintTicks(true);

mS.setMinorTickSpacing(1);

mS.setMajorTickSpacing(4);

mS.setPaintLabels(true);

mS.setSnapToTicks(true);

// mS.setSize(20,20);

//myTF.setSize(140, 140);

myTF.setColumns(5);

myPanel1.setSize(200, 300);

// myPanel1.setBackground(Color.red);

// myPanel4.setBackground(Color.red);

myPanel1.setLayout(new GridLayout(11,0));

myPanel1.add(myLabel1);

myPanel1.add(new JLabel(" -------------------------------------------"));

myPanel1.add(myLabel2);

myPanel4.add(myTF);

myPanel1.add(myPanel4);

myPanel1.add(new JLabel(" Process Size:"));

myPanel5.add(mS);

myPanel1.add(myPanel5);

myPanel6.add(new JLabel(" 0 4 8"));

myPanel1.add(myPanel6);

myPanel7.add(myButton1);

// myPanel7.add(new JButton());

myPanel1.add(myPanel7);

myPanel1.add(new JLabel(" -------------------------------------------"));

myPanel1.add(myButton2);

myPanel2.setSize(400, 300);

myPanel2.setLocation(202, 0);

myPanel2.setBackground(Color.gray);

myPanel8 = new JPanel();

myPanel9 = new JPanel();

myPanel8.setSize(50,100);

myPanel8.setPreferredSize(new Dimension(388, 80));

// myPanel8.setBackground(Color.blue);

myPanel8.add(new JLabel("Queue"));

myTable.setPreferredScrollableViewportSize(new Dimension(380,16));

myPanel8.add(mySP);

//myTable.setCellSelectionEnabled(false);

myPanel9.setPreferredSize(new Dimension(388, 204));

// myPanel9.setBackground(Color.orange);

myPanel2.add(myPanel8);

myPanel2.add(myPanel9);

myPanel9.add(new JLabel(" Physical memory (40 Mb) :"

+ " "));

myPanel10 = new JPanel(new GridLayout(10,1));

myPanel10.add(new JLabel(" 0"));

myPanel10.add(new JLabel(""));

myPanel10.add(new JLabel(" 8"));

myPanel10.add(new JLabel(""));

myPanel10.add(new JLabel(" 16"));

myPanel10.add(new JLabel(""));

myPanel10.add(new JLabel(" 24"));

myPanel10.add(new JLabel(""));

myPanel10.add(new JLabel(" 32"));

myPanel10.setPreferredSize(new Dimension(35,170));

//myPanel10.setBackground(Color.red);

// myPanel10.setLocation(209, 100);

myPanel9.add(myPanel10);

JPanel p11 = new JPanel(new GridLayout(5,1));

p11.setPreferredSize(new Dimension(300,160));

p11.setBackground(Color.YELLOW);

myPanel9.add(p11);

myProgBar1 = new JProgressBar(0,8);

myProgBar2 = new JProgressBar(0,8);

myProgBar3 = new JProgressBar(0,8);

myProgBar4 = new JProgressBar(0,8);

myProgBar5 = new JProgressBar(0,8);

p11.add(myProgBar1);

p11.add(myProgBar2);

p11.add(myProgBar3);

p11.add(myProgBar4);

p11.add(myProgBar5);

//myProgBar1.setValue(2);

//myProgBar1.setStringPainted(true);

//myProgBar1.setString("4Mb");

myPanel3.setSize(200, 300);

//myPanel3.setBackground(Color.blue);

myFrame.setSize(620, 338);

myFrame.add(myPanel1);

myFrame.add(myPanel2);

myFrame.add(myPanel3);

myFrame.setVisible(true);

myFrame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

class AddActionListener implements ActionListener {

public void actionPerformed(ActionEvent e) {

System.out.println("fuck add");

str = myTF.getText();

try {

int time = Integer.parseInt(str);

} catch (NumberFormatException nb) {

//this.actionPerformed(e);

return;

}

int time = Integer.parseInt(str);

str1 = Integer.toString(mS.getValue());

myTable.setValueAt(str+" / "+str1, 0, column--);

myQ.addToQueue(new Process(time, mS.getValue()));

}

}

ActionListener addListener = new AddActionListener();

myButton1.addActionListener(addListener);

class tooGuiProc implements Runnable {

public void run() {

myManag.myManager();

}

}

class EnterActionListener implements ActionListener {

public void actionPerformed (ActionEvent e) {

//myManag.myManager();

Thread th = new Thread(new tooGuiProc());

th.start();

}

}

ActionListener enterListener = new EnterActionListener();

myButton2.addActionListener(enterListener);

}

public void setProgBar (int numBar,int sizeValue) {

System.out.println("Bar");

if (numBar == 0) {

myProgBar1.setValue(sizeValue);

myProgBar1.setStringPainted(true);

myProgBar1.setString(Integer.toString(sizeValue) + "Mb");

//myProgBar1.updateUI();

return;

}

if (numBar == 1) {

myProgBar2.setValue(sizeValue);

myProgBar2.setStringPainted(true);

myProgBar2.setString(Integer.toString(sizeValue) + "Mb");

return;

}

if (numBar == 2) {

myProgBar3.setValue(sizeValue);

myProgBar3.setStringPainted(true);

myProgBar3.setString(Integer.toString(sizeValue) + "Mb");

return;

}

if (numBar == 3) {

myProgBar4.setValue(sizeValue);

myProgBar4.setStringPainted(true);

myProgBar4.setString(Integer.toString(sizeValue) + "Mb");

return;

}

if (numBar == 4) {

myProgBar5.setValue(sizeValue);

myProgBar5.setStringPainted(true);

myProgBar5.setString(Integer.toString(sizeValue) + "Mb");

return;

}

}

--------Main.java-------

package os\_3;

/\*\*

\*

\* @author SichikUA

\*/

public class Main {

/\*\*

\* @param args the command line arguments

\*/

public static void main(String[] args) {

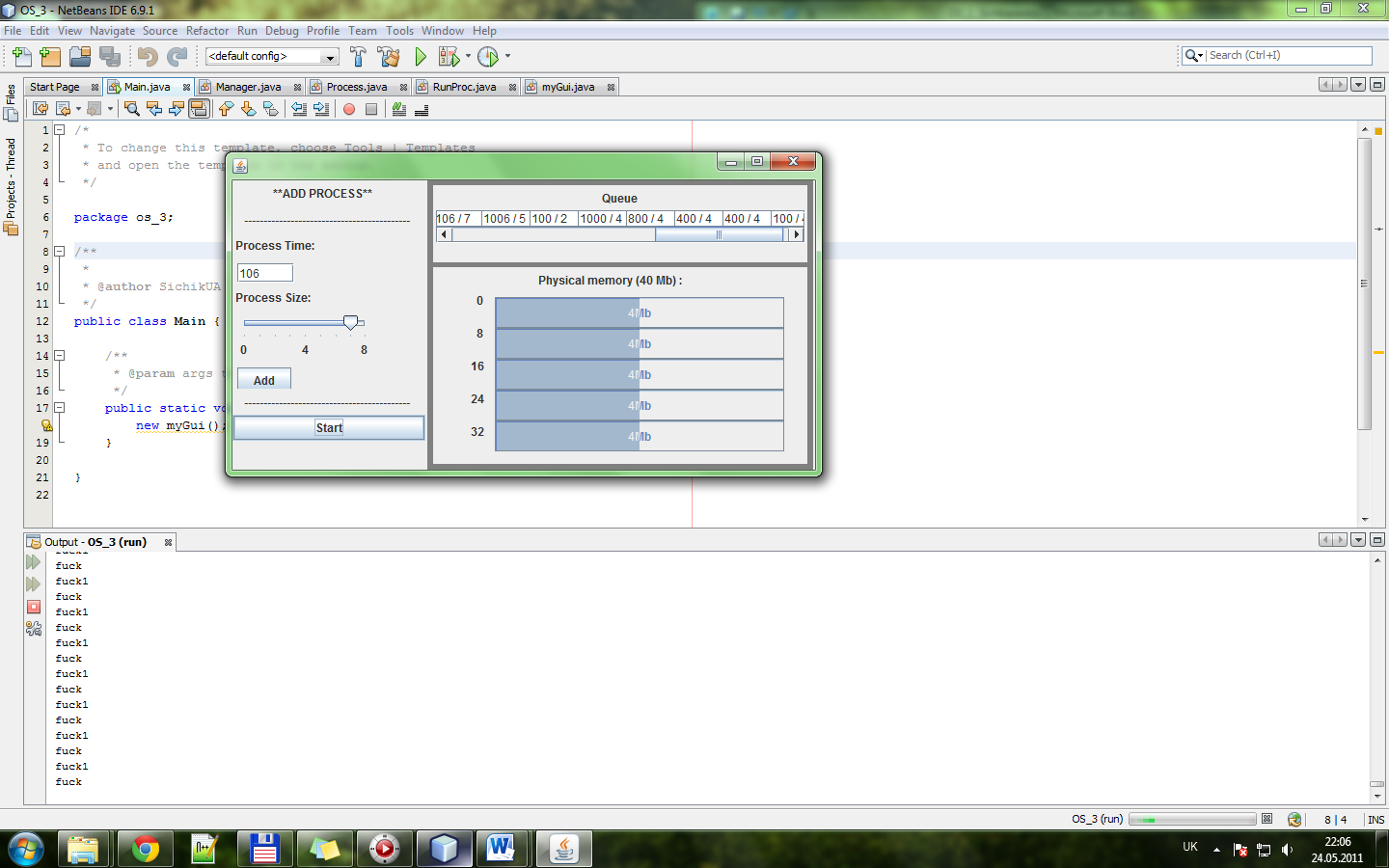
new myGui();

}

}

*Результат роботи програми зображено в таблиці:*

Початок роботи программи



Завершальний етап

