**МІНІСТЕРСТВО ОСВІТИ ТА НАУКИ УКРАЇНИ**

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ**

**«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ»**

*Факультет прикладної математики*

*Кафедра «Спеціалізовані комп’ютерні системи»*

Лабораторна робота №1

з

Операційних систем

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

м. Київ

2011

*Завдання для лабораторної роботи*

*Варіант 1*

. **FCFS (**First Come First Serve) – першим прийшов – першим обслуговується.

Це – невитісняльна стратегія. Процеси виконуються в послідовності їх надходження до системи. i-й процес може бути запущений на виконання тільки після завершення (i-1)-го процесу. Якщо в момент надходження i-го процесу (i-1)-й процес ще не закінчився, тоді i-й процес переходить в стан очікування до момента вивільнення процесора.

*Текст програми:*

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

/\*\*

\*

\* @author SichikUA

\*/

public class Process {

private int id;

private int startTime;

private int workingTime;

//private int realStartTime;

private int endTime;

private int delayTime;

public Process (int id, int startTime, int workingTime) {

this.id = id;

this.startTime = startTime;

this.workingTime = workingTime;

this.endTime = -1;

this.delayTime = -1;

}

public int getId() {

return this.id;

}

public int getStartTime() {

return this.startTime;

}

public int getWorkingTime() {

return this.workingTime;

}

public int getEndTime() {

return this.endTime;

}

public int getDelayTime() {

return this.delayTime;

}

public void setEndTime(int endTime) {

this.endTime = endTime;

}

public void setDelayTime(int delayTime) {

this.delayTime = delayTime;

}

}

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

import java.util.\*;

/\*\*

\*

\* @author SichikUA

\*/

public class Manager {

private int currentTime;

private LinkedList<Process> procList;

private LinkedList<Process> executProc;

public Manager() {

currentTime = 0;

procList = new LinkedList<Process>();

procList.add(new Process(1,0,4));

procList.add(new Process(2,3,5));

procList.add(new Process(3,7,2));

procList.add(new Process(4,13,7));

executProc = new LinkedList<Process>();

}

public void taskManager() {

Process tmp;

while ( !procList.isEmpty() ) {

tmp = procList.remove();

if (currentTime > tmp.getStartTime()) {

tmp.setDelayTime(currentTime - tmp.getStartTime());

currentTime += tmp.getWorkingTime();

}

else {

tmp.setDelayTime(0);

currentTime = tmp.getStartTime() + tmp.getWorkingTime();

}

tmp.setEndTime(currentTime);

executProc.add(tmp);

}

}

public void displayResults() {

System.out.println(" id " + " StatrtTime " +" WorkingTime "

+ " EndTime " + " DelayTime" );

for (Process p : executProc){

System.out.format("%4d " , p.getId());

System.out.format("%7d " , p.getStartTime());

System.out.format("%12d" , p.getWorkingTime());

System.out.format("%8d " , p.getEndTime());

System.out.format("%9d " , p.getDelayTime());

System.out.println();

}

}

}

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

package os1;

/\*\*

\*

\* @author SichikUA

\*/

public class Main {

/\*\*

\* @param args the command line arguments

\*/

public static void main(String[] args) {

Manager m = new Manager();

m.taskManager();

m.displayResults();

}

}

*Результат роботи програми зображено в таблиці:*

