Міністерство освіти і науки, молоді та спорту України

Національний технічний університет України

«Київський політехнічний інститут»

Факультет інформатики та обчислювальної техніки

Кафедра обчислювальної техніки

**Лабораторна робота №5**

З дисципліни «Комп’ютерне моделювання»

Виконав: Перевірив:

Студент групи ІО-21 доц. Марковський О.П.

Коноз А.О.

Дата здачі\_\_\_\_\_\_\_\_\_\_\_\_\_

Захищено з балом\_\_\_\_\_

Київ 2014

**Лістинг коду**

**import** java.util.ArrayList;

**public** **class** Main {

**public** ArrayList<Object> generateStates(**int** per[][]){

// int per[] ={1,3,2}; //new int[10];

**int** obr[][] = **new** **int**[10][2];

**int** i, j,min,raz,n;

**int** tmp[];

**boolean** flag;

n = per.length;

min = 0;

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

**while** (**true**) {

// for (int k = 0; k < per.length; k++) {

// System.out.print(per[k][0] + " ");

// }

result.add(per.clone());

flag = **false**;

**for** (i = n-2; i >= 0; i--) {

**if**(per[i][0] < per[i+1][0]){

flag = **true**;

**break**;

}

}

**if**(!flag){

**break**;

}

// System.out.println();

raz = per[i+1][0];

**for** (j = i+1; j < n; j++) {

**if**(((per[j][0] - per[i][0]) < raz) && (per[i][0] < per[j][0])){

min = j;

}

}

tmp = per[i];

per[i] = per[min];

per[min] = tmp;

**for** (j = i+1; j < n; j++) {

obr[j] = per[j];

}

j = i+1;

**for** (**int** k = n-1; k >= i+1; k--) {

per[j] = obr[k];

j++;

}

}

**return** result;

}

**void** random(**int** tasks[][]){

**boolean** swapped = **true**;

**while** (swapped){

swapped = **false**;

**for** (**int** i = 0; i < tasks.length - 1; i++) {

**if** (tasks[i][0] > tasks[i + 1][0]) {

**int** a[] = tasks[i];

tasks[i] = tasks[i + 1];

tasks[i + 1] = a;

swapped = **true**;

}

}

}

Object list[] = generateStates(tasks).toArray();

**int** results[][] = **new** **int**[list.length][3];

**for** (**int** i = 0; i < list.length; i++) {

results[i] = countGant((**int**[][])list[i]);

}

**int** minValue = Integer.***MAX\_VALUE***;

**int** colNumber = 0;

**for** (**int** i = 0; i < results.length; i++) {

// System.out.println(results[i][2]);

**if**(results[i][2] < minValue){

minValue = results[i][2];

colNumber = i;

}

}

System.***out***.println("Minimal system time with standart search: " + minValue);

}

**public** **int**[] countGant(**int** tasks[][]){

**int** solving1 = 0;

**int** solving2 = 0;

**int** result [] = **new** **int**[3];

**for** (**int** i = 0; i < tasks.length; i++) {

solving1 += tasks[i][0];

**if**(solving2 <= solving1){

solving2 = tasks[i][1] + solving1;

}**else**{

solving2 += tasks[i][1];

}

}

result[0] = solving1;

result[1] = solving2;

**if**(solving1 < solving2){

result[2] = solving2;

}**else**{

result[2] = solving1;

}

**return** result;

}

**int**[][] optimize(**int** tasks[][]){

**boolean** usage[] = **new** **boolean**[tasks.length];

**int** result[][] = **new** **int**[tasks.length][2];

**int** minValue = Integer.***MAX\_VALUE***;

**int** processorNumber = 0;

**int** tasksNumber = 0;

**int** headPointer = 0;

**int** tailPointer = tasks.length - 1;

**for** (**int** i = 0; i < usage.length; i++) {

**for** (**int** j = 0; j < usage.length; j++) {

**if**(!usage[j]){

**if**(tasks[j][0] < minValue){

minValue = tasks[j][0];

processorNumber = 1;

tasksNumber = j;

}

**if**(tasks[j][1] < minValue){

minValue = tasks[j][1];

processorNumber = 2;

tasksNumber = j;

}

}

}

**if**(processorNumber == 1){

result[headPointer] = tasks[tasksNumber];

headPointer++;

usage[tasksNumber] = **true**;

}**else**{

result[tailPointer] = tasks[tasksNumber];

tailPointer--;

usage[tasksNumber] = **true**;

}

minValue = Integer.***MAX\_VALUE***;

}

**return** result;

}

**public** **static** **void** main(String[] args) {

**int** tasks[][] = {{5,8},{2,11},{4,3},{3,1},{10,7},{9,6}};

**int** optTasks[][] ={{2,11},{5,8},{10,7},{9,6},{4,3},{3,1}};

**int** per[][] = {{2,7},{4,4},{3,1},{1,6}};

Main core = **new** Main();

**int** result[] = core.countGant(core.optimize(tasks));

System.***out***.println("First processor summary time: " + result[0]);

System.***out***.println("Second processor summary time: " + result[1]);

System.***out***.println("--------------------------------------------------------");

System.***out***.println("Minimal system summary time using calendar planning: " + result[2]);

core.random(tasks.clone());

}

}