Приклади до лабораторних робіт

**Приклад 1 Конфлікт доступу до об’єкту**

public class Counter {

private int c=0;

public synchronized void increment(){

c++;

}

public synchronized void decrement(){

c--;

}

@Override

public String toString(){

return "\n counter = "+c;

}

public void print(){

System.out.println(this.toString());

}

}

public class ProcessCounter {

Counter counter = new Counter();

public void test() throws InterruptedException {

Thread first = new Thread(new Runnable() {

@Override

public void run() {

for (int j = 0; j < 100; j++) {

counter.increment();

counter.print();

}

}

});

Thread second = new Thread(new Runnable() {

@Override

public void run() {

for (int j = 0; j < 100; j++) {

counter.decrement();

counter.print();

}

}

});

first.start();

second.start();

Thread.sleep(50);

System.out.println("RESULT = "+counter.toString());

}

public static void main(String[] args) throws InterruptedException{

ProcessCounter test = new ProcessCounter();

test.test();

}

}

**Приклад 2 Експериментальне дослідження ефекту розпаралелювання**

class A implements Cloneable, Comparable{

private static int next = 0;

private int num;

private String name;

public A(){

next++;

num=next;

name = "untitled";

}

public void setName(String s){

name = s;

}

public String getName(){

return name;

}

public int getNum(){

return num;

}

/\* @Override

public int compareTo(Object o) {

return (this.getNum()-((A)o).getNum());

}\*/

@Override

public A clone() throws CloneNotSupportedException{

return (A)super.clone();

}

@Override

public int compareTo(Object o) {

return this.getNum()-((A)o).getNum();

}

}

public class ParallelTest {

public static void main(String[] args){

int size = 10000000;

A[] aaa = new A[size];

char c = 'a';

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

aaa[j] = new A();

aaa[j].setName(String.valueOf(c));

c = (char) (c+1);

if(c=='z')

c='a';

}

long start = System.currentTimeMillis();

Arrays.sort(aaa,Comparator.comparing(

A::getName ).thenComparing(A::getNum));

long time = System.currentTimeMillis()-start;

System.out.println("Time of ordinary sorting = "+time);

start = System.currentTimeMillis();

Arrays.parallelSort(aaa, Comparator.comparing(

A::getName ).thenComparing(A::getNum));

time = System.currentTimeMillis()-start;

System.out.println("Time of parallel sorting = "+time);

}

}