МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»

ІНСТИТУТ ПРИКЛАДНОГО СИСТЕМНОГО АНАЛІЗУ

Кафедра системного проектування

**ЛАБОРАТОРНА РОБОТА №2**

з дисципліни: «ПАРАЛЕЛЬНІ ОБЧИСЛЕННЯ»

Виконав:

студент 3 курсу

групи ДА-81

Головін Б. О.

Варіант 5

Київ – 2021

***Лабораторна робота 2***

**Використання і синхронізація потоків**

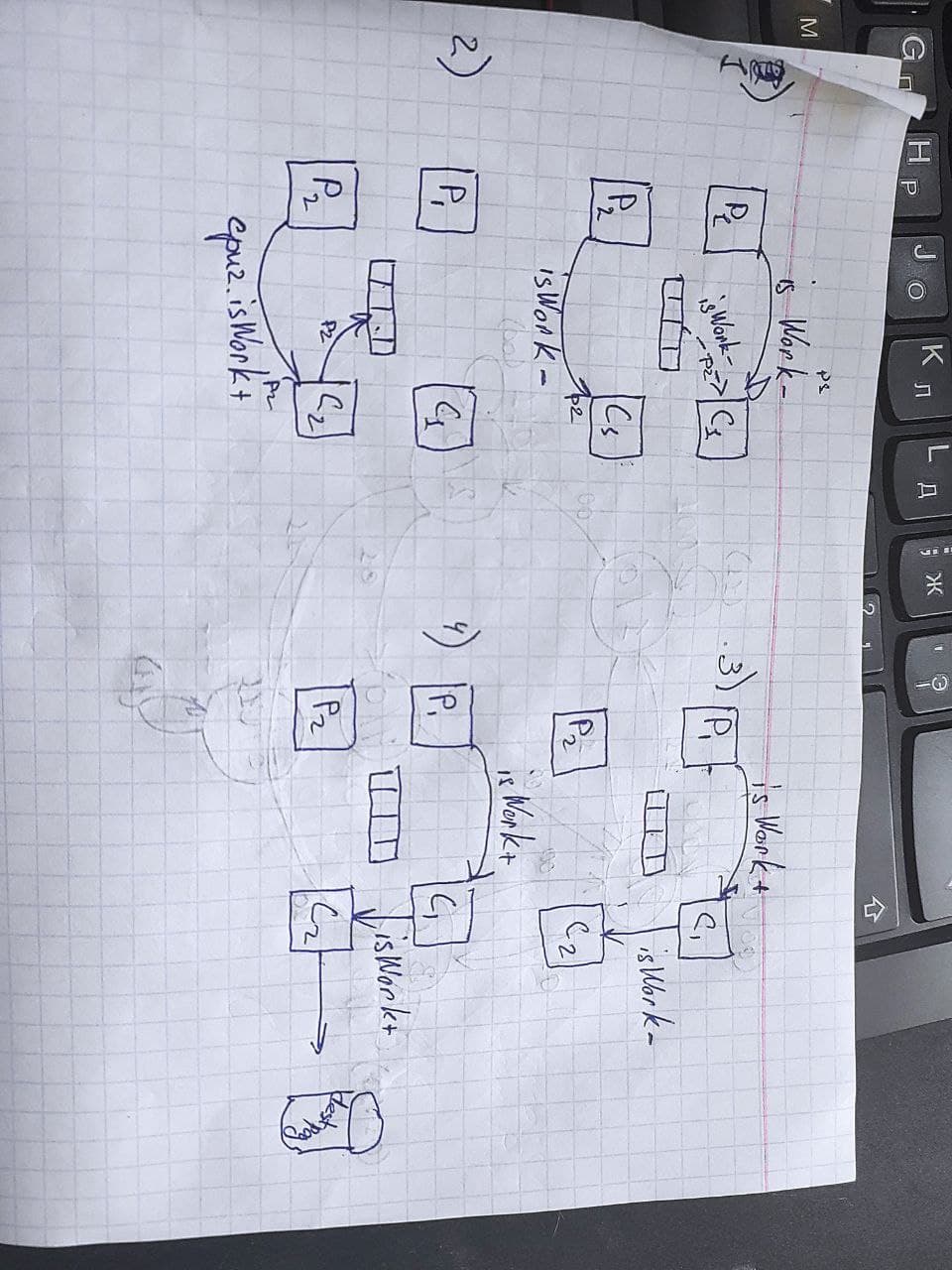
***Мета роботи:***

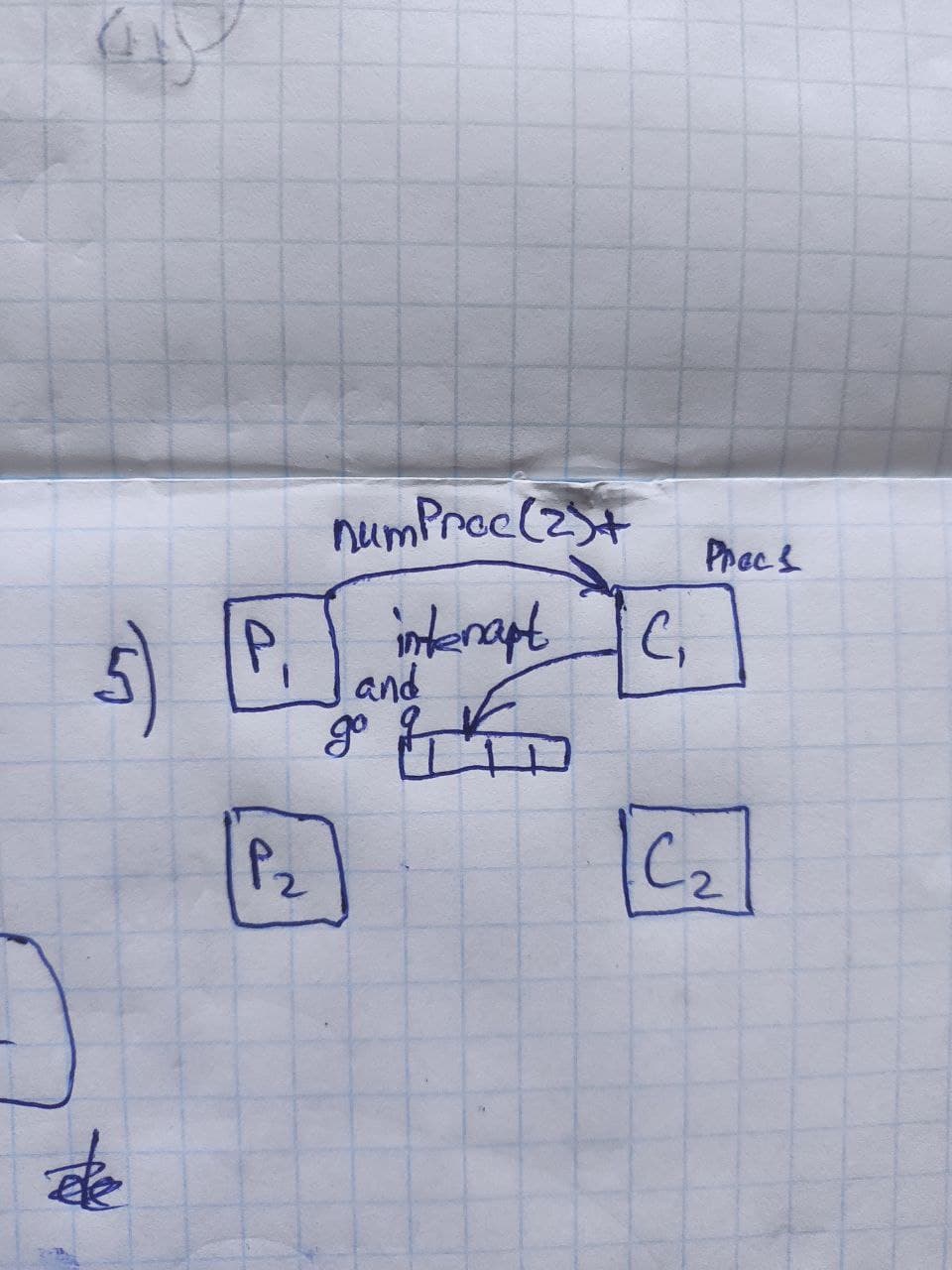
Метою роботи є набуття навичок роботи з потоками при програмуванні на мові Java або будь-якій іншій.

***Завдання*** Варіант 5

Програма моделює обслуговування двох потоків процесів з різними параметрами двома центральними процесорами комп’ютера з однією чергою. Якщо згенеровано процес другого потоку і другий процесор занятий, процес поміщається в чергу. Якщо згенеровано процес першого потоку, то, якщо перший процесор занятий обробкою першого потоку, то процес надходить на обробку на другий процесор. Якщо і другий процесор занятий, то процес знищується. Якщо в момент генерації процесу першого потоку на першому процесорі обробляється процес другого потоку, то процес переривається і повертається в чергу. Визначити максимальну довжину черги, процент знищених процесів для першого потоку і процент перерваних запитів другого потоку.

**Схеми до задачі**

****

****

**Лістинг програми**

**Main.java**

public class Main {  
  
 static int *deleted* = 0;  
 static int *inter* = 0;  
 static int *processToGenerate* = 20;  
  
 public static void main(String[] args) throws InterruptedException {  
 int queueCapacity = 10;  
 //int processToGenerate = 20;  
 System.*out*.println("\nОбъем очереди " + queueCapacity + "\nЗадач будет сгенерировано " + *processToGenerate* \*2 + "\n");  
 QueueClass queue = new QueueClass(queueCapacity);  
  
 CPU1 cpu1 = new CPU1(queue);  
 CPU2 cpu2 = new CPU2(queue);  
  
 Process1 process1 = new Process1(queue, *processToGenerate*, cpu1, cpu2);  
 Process2 process2 = new Process2(queue, *processToGenerate*, cpu1, cpu2);  
  
 new Thread(process1).start();  
 new Thread(process2).start();  
  
 cpu1.start();  
 cpu2.start();  
 }  
}

**QueueClass.java**

import java.util.LinkedList;  
import java.util.Queue;  
  
class QueueClass {  
  
 private Queue<String> queue = new LinkedList<>();  
 private int capacity;  
 private int maxSize = 0;  
  
 public QueueClass(int capacity) {  
 this.capacity = capacity;  
 }  
  
 public int getMaxSize() {  
 return maxSize;  
 }  
  
 public boolean queueIsEmpty() {  
 return queue.isEmpty();  
 }  
  
 public synchronized void put(String element) throws InterruptedException {  
// while (queue.size() == capacity) {  
// System.out.println("QUEUE : очередь полная. Ожидание");  
// wait();  
// }  
 queue.add(element);  
  
 if (queue.size() > maxSize)  
 maxSize = queue.size();  
  
 System.*out*.println("QUEUE : Пришла задача " + element + "; Размер очереди: " + queue.size());  
 // notify(); // notifyAll() for multiple CPU/CPUProcess threads  
 }  
  
 public synchronized String get() throws InterruptedException {  
// while (queue.isEmpty()) {  
// System.out.println("QUEUE : очередь пустая. Ожидание");  
// wait();  
// }  
 String item = queue.remove();  
 System.*out*.println("QUEUE : Задача 2 удалена. Размер очереди = [" + queue.size() + "]");  
 // notify(); // notifyAll() for multiple CPU/CPUProcess threads  
 return item;  
 }  
  
  
}

**Process1.java**

class Process1 implements Runnable {  
  
 QueueClass queue;  
 CPU1 cpu1;  
 CPU2 cpu2;  
 int generateNumber;  
  
 Process1(QueueClass q, int gN, CPU1 cpu1, CPU2 cpu2) {  
 this.queue = q;  
 this.generateNumber = gN;  
 this.cpu1 = cpu1;  
 this.cpu2 = cpu2;  
 }  
  
 private void destroy() throws InterruptedException {  
 Main.*deleted*++;  
 System.*out*.println("PRO 1 : Задача 1 удалена // CPU 1 занят, CPU 2 занят // deleted = " + Main.*deleted*);  
 }  
  
 public void run() {  
 long generateDelay;  
 for (int i = 0; i < generateNumber; i++) {  
 int randMin = 10;  
 int randMax = 40; // rand = [10,50]  
 generateDelay = randMin + (int) (Math.*random*() \* randMax);  
 try {  
 Thread.*sleep*(generateDelay);  
 } catch (InterruptedException e) {  
 e.printStackTrace();  
 }  
 try {  
 System.*out*.println("PRO 1 : Задача 1 была сгенерирована за (" + generateDelay + ")");  
 if (!cpu1.checkWork()) {  
 cpu1.setData("1");  
 System.*out*.println("PRO 1 : Задача 1 пошла на CPU 1");  
 } else if (cpu1.checkWork() && cpu1.data.equals("1")) { // перехожим на 2 CPU с проверкой  
 if (cpu2.checkWork()) {  
 destroy();  
 } else {  
 cpu2.setData("1");  
 System.*out*.println("PRO 1 : Задача 1 пошла на CPU 2 // CPU 1 занят");  
 }  
  
 } else if (cpu1.checkWork() && cpu1.data.equals("2")) {  
 System.*out*.println("---");  
 queue.put(cpu1.getData());  
 cpu1.interrupt();  
 Main.*inter*++;  
 // Thread.currentThread().interrupt();  
 cpu1.setData("1");  
 System.*out*.println("PRO 1 : Задача 1 пошла на CPU 1, Задача 2 прервана c CPU 2 пошла в очередь // interrupt = " + Main.*inter*);  
 System.*out*.println("---");  
 }  
 } catch (InterruptedException e) {  
 e.printStackTrace();  
 }  
 }  
 System.*out*.println("PRO 1 : задачи закончились" + queue.getMaxSize());  
 try {  
 Thread.*sleep*(3300);  
 System.*out*.println("\n\n\n-----------------------------------\nМаксимальная длина очереди = " + queue.getMaxSize());  
 System.*out*.println("Процент уничтоженых задач для первого потока = " +(double) Main.*deleted*/Main.*processToGenerate*\*100);  
 System.*out*.println("Процент прерваных задач для второго потока = " +(double) Main.*inter*/Main.*processToGenerate*\*100);  
 } catch (InterruptedException e) {  
 e.printStackTrace();  
 }  
  
 }  
}

**Process2.java**

class Process2 implements Runnable {  
  
 QueueClass queue;  
 CPU1 cpu1; CPU2 cpu2;  
 int generateNumber;  
  
  
 Process2(QueueClass q, int gN, CPU1 cpu1, CPU2 cpu2) {  
 this.queue = q;  
 this.generateNumber = gN;  
 this.cpu1 = cpu1;  
 this.cpu2 = cpu2;  
 }  
  
 public void run() {  
 long generateDelay;  
 for (int i = 0; i < generateNumber; i++) {  
 int randMin = 10;  
 int randMax = 50; // rand = [10,50]  
 generateDelay = randMin + (int) (Math.*random*() \* randMax);  
 try {  
 Thread.*sleep*(generateDelay);  
 } catch (InterruptedException e) {  
 e.printStackTrace();  
 }  
 try {  
 System.*out*.println("PRO 2 : Задача 2 была сгенерирована за (" + generateDelay + ")");  
 if (!cpu2.checkWork()) {  
 cpu2.setData("2");  
 System.*out*.println("PRO 2 : Задача 2 пошла на CPU 2");  
 } else {  
 System.*out*.println("PRO 2 : Задача 2 пошла на очередь");  
 queue.put("2");  
 }  
 } catch (InterruptedException e) {  
 e.printStackTrace();  
 }  
 }  
 System.*out*.println("PRO 2 : задачи закончились" + queue.getMaxSize());  
 }  
}

**CPU1.java**

class CPU1 extends Thread {  
  
 QueueClass queue;  
 String data = "";  
  
 private boolean isWork = false;  
  
 public boolean checkWork() {  
 if (isWork) return true;  
 return false;  
 }  
  
 CPU1(QueueClass q) {  
 this.queue = q;  
 }  
  
 public String getData(){  
 return data;  
 }  
  
 public void setData(String newData){  
 data = newData;  
 }  
  
 @Override  
 public void run() {  
 long processingTime;  
 while (true) {  
 int randMin = 20;  
 int randMax = 80; // rand = [20,100]  
 processingTime = randMin + (int) (Math.*random*() \* randMax);  
 try {  
 if (data != "") {  
 isWork = true;  
 System.*out*.println("CPU 1 : Начал работу напрямую - задача " + data + "; Время работы (" + processingTime + ")");  
 Thread.*sleep*(processingTime);  
 setData("");  
 isWork = false;  
 System.*out*.println("CPU 1 : Закончил работу");  
 } else if (!queue.queueIsEmpty()) {  
 isWork = true;  
 data = queue.get();  
 System.*out*.println("CPU 1 : Начал работу с очереди - задача " + data + "; Время работы (" + processingTime + ")");  
 Thread.*sleep*(processingTime);  
 setData("");  
 isWork = false;  
 System.*out*.println("CPU 1 : Закончил работу");  
 }  
 } catch (InterruptedException e) {  
 //e.printStackTrace();  
 }  
 }  
 }  
}

**CPU2.java**

class CPU2 extends Thread {  
  
 QueueClass queue;  
 String data = "";  
 private boolean isWork = false;  
  
 public boolean checkWork() {  
 if (isWork) return true;  
 return false;  
 }  
  
 public String getData(){  
 return data;  
 }  
  
 public void setData(String newData){  
 data = newData;  
 }  
  
 CPU2(QueueClass q) {  
 this.queue = q;  
 }  
  
 @Override  
 public void run() {  
 long processingTime;  
 while (true) {  
 int randMin = 20;  
 int randMax = 100; // rand = [20,100]  
 processingTime = randMin + (int) (Math.*random*() \* randMax);  
 try {  
 if (data != "") {  
 isWork = true;  
 System.*out*.println("CPU 2 : Начал работу напрямую - задача " + data + "; Время работы " + processingTime);  
 Thread.*sleep*(processingTime);  
 setData("");  
 isWork = false;  
 }  
// else if (!queue.queueIsEmpty()) {  
// isWork = true;  
// System.out.println("CPU 2 : Начал работу с очереди " + processingTime);  
// data = queue.get();  
// Thread.sleep(processingTime);  
// data = "";  
// isWork = false;  
// System.out.println("CPU 2 : Processed end ");  
 // }  
 } catch (InterruptedException e) {  
 e.printStackTrace();  
 }  
  
 }  
 }  
}

**Результати роботи програми**

C:\Users\Bogdan\.jdks\openjdk-15.0.2\bin\java.exe -javaagent:C:\Users\Bogdan\AppData\Local\JetBrains\Toolbox\apps\IDEA-U\ch-0\203.7148.57\lib\idea\_rt.jar=60546:C:\Users\Bogdan\AppData\Local\JetBrains\Toolbox\apps\IDEA-U\ch-0\203.7148.57\bin -Dfile.encoding=UTF-8 -classpath C:\Projects\Parallels\lab2\out\production\lab2 Main

Объем очереди 10

Задач будет сгенерировано 40

PRO 2 : Задача 2 была сгенерирована за (10)

PRO 2 : Задача 2 пошла на CPU 2

PRO 1 : Задача 1 была сгенерирована за (17)

PRO 1 : Задача 1 пошла на CPU 1

CPU 2 : Начал работу напрямую - задача 2; Время работы 102

CPU 1 : Начал работу напрямую - задача 1; Время работы (57)

PRO 1 : Задача 1 была сгенерирована за (15)

PRO 1 : Задача 1 удалена // CPU 1 занят, CPU 2 занят // deleted = 1

PRO 2 : Задача 2 была сгенерирована за (44)

PRO 2 : Задача 2 пошла на очередь

QUEUE : Пришла задача 2; Размер очереди: 1

PRO 1 : Задача 1 была сгенерирована за (38)

PRO 1 : Задача 1 удалена // CPU 1 занят, CPU 2 занят // deleted = 2

CPU 1 : Закончил работу

QUEUE : Задача 2 удалена. Размер очереди = [0]

CPU 1 : Начал работу с очереди - задача 2; Время работы (53)

PRO 1 : Задача 1 была сгенерирована за (13)

---

QUEUE : Пришла задача 2; Размер очереди: 1

CPU 1 : Начал работу напрямую - задача 1; Время работы (55)

PRO 1 : Задача 1 пошла на CPU 1, Задача 2 прервана c CPU 2 пошла в очередь // interrupt = 1

---

PRO 1 : Задача 1 была сгенерирована за (11)

PRO 1 : Задача 1 удалена // CPU 1 занят, CPU 2 занят // deleted = 3

PRO 2 : Задача 2 была сгенерирована за (47)

PRO 2 : Задача 2 пошла на очередь

QUEUE : Пришла задача 2; Размер очереди: 2

PRO 1 : Задача 1 была сгенерирована за (35)

PRO 1 : Задача 1 пошла на CPU 2 // CPU 1 занят

CPU 2 : Начал работу напрямую - задача 1; Время работы 98

CPU 1 : Закончил работу

PRO 2 : Задача 2 была сгенерирована за (23)

PRO 2 : Задача 2 пошла на очередь

QUEUE : Задача 2 удалена. Размер очереди = [1]

CPU 1 : Начал работу с очереди - задача 2; Время работы (68)

QUEUE : Пришла задача 2; Размер очереди: 2

PRO 1 : Задача 1 была сгенерирована за (10)

---

QUEUE : Пришла задача 2; Размер очереди: 3

PRO 1 : Задача 1 пошла на CPU 1, Задача 2 прервана c CPU 2 пошла в очередь // interrupt = 2

---

CPU 1 : Начал работу напрямую - задача 1; Время работы (98)

PRO 1 : Задача 1 была сгенерирована за (28)

PRO 1 : Задача 1 удалена // CPU 1 занят, CPU 2 занят // deleted = 4

PRO 2 : Задача 2 была сгенерирована за (34)

PRO 2 : Задача 2 пошла на очередь

QUEUE : Пришла задача 2; Размер очереди: 4

PRO 2 : Задача 2 была сгенерирована за (40)

PRO 2 : Задача 2 пошла на очередь

PRO 1 : Задача 1 была сгенерирована за (42)

QUEUE : Пришла задача 2; Размер очереди: 5

PRO 1 : Задача 1 удалена // CPU 1 занят, CPU 2 занят // deleted = 5

CPU 1 : Закончил работу

QUEUE : Задача 2 удалена. Размер очереди = [4]

CPU 1 : Начал работу с очереди - задача 2; Время работы (56)

PRO 1 : Задача 1 была сгенерирована за (35)

---

PRO 2 : Задача 2 была сгенерирована за (35)

QUEUE : Пришла задача 2; Размер очереди: 5

PRO 2 : Задача 2 пошла на CPU 2

CPU 2 : Начал работу напрямую - задача 2; Время работы 58

CPU 1 : Начал работу напрямую - задача 1; Время работы (70)

PRO 1 : Задача 1 пошла на CPU 1, Задача 2 прервана c CPU 2 пошла в очередь // interrupt = 3

---

PRO 1 : Задача 1 была сгенерирована за (33)

PRO 1 : Задача 1 удалена // CPU 1 занят, CPU 2 занят // deleted = 6

PRO 2 : Задача 2 была сгенерирована за (54)

PRO 2 : Задача 2 пошла на очередь

QUEUE : Пришла задача 2; Размер очереди: 6

CPU 1 : Закончил работу

QUEUE : Задача 2 удалена. Размер очереди = [5]

CPU 1 : Начал работу с очереди - задача 2; Время работы (22)

PRO 2 : Задача 2 была сгенерирована за (24)

PRO 2 : Задача 2 пошла на CPU 2

CPU 2 : Начал работу напрямую - задача 2; Время работы 77

PRO 1 : Задача 1 была сгенерирована за (47)

---

QUEUE : Пришла задача 2; Размер очереди: 6

PRO 1 : Задача 1 пошла на CPU 1, Задача 2 прервана c CPU 2 пошла в очередь // interrupt = 4

---

CPU 1 : Начал работу напрямую - задача 1; Время работы (41)

PRO 2 : Задача 2 была сгенерирована за (27)

PRO 2 : Задача 2 пошла на очередь

QUEUE : Пришла задача 2; Размер очереди: 7

PRO 2 : Задача 2 была сгенерирована за (14)

PRO 2 : Задача 2 пошла на очередь

QUEUE : Пришла задача 2; Размер очереди: 8

CPU 1 : Закончил работу

PRO 1 : Задача 1 была сгенерирована за (41)

QUEUE : Задача 2 удалена. Размер очереди = [7]

CPU 1 : Начал работу с очереди - задача 2; Время работы (58)

PRO 1 : Задача 1 была сгенерирована за (19)

---

QUEUE : Пришла задача 2; Размер очереди: 8

PRO 1 : Задача 1 пошла на CPU 1, Задача 2 прервана c CPU 2 пошла в очередь // interrupt = 5

---

CPU 1 : Начал работу напрямую - задача 1; Время работы (92)

PRO 2 : Задача 2 была сгенерирована за (24)

PRO 2 : Задача 2 пошла на очередь

QUEUE : Пришла задача 2; Размер очереди: 9

PRO 1 : Задача 1 была сгенерирована за (29)

PRO 1 : Задача 1 пошла на CPU 2 // CPU 1 занят

CPU 2 : Начал работу напрямую - задача 1; Время работы 110

PRO 2 : Задача 2 была сгенерирована за (26)

PRO 2 : Задача 2 пошла на очередь

QUEUE : Пришла задача 2; Размер очереди: 10

PRO 1 : Задача 1 была сгенерирована за (25)

PRO 1 : Задача 1 удалена // CPU 1 занят, CPU 2 занят // deleted = 7

PRO 1 : Задача 1 была сгенерирована за (19)

PRO 1 : Задача 1 удалена // CPU 1 занят, CPU 2 занят // deleted = 8

PRO 2 : Задача 2 была сгенерирована за (54)

PRO 2 : Задача 2 пошла на очередь

QUEUE : Пришла задача 2; Размер очереди: 11

PRO 1 : Задача 1 была сгенерирована за (13)

PRO 1 : Задача 1 удалена // CPU 1 занят, CPU 2 занят // deleted = 9

CPU 1 : Закончил работу

QUEUE : Задача 2 удалена. Размер очереди = [10]

CPU 1 : Начал работу с очереди - задача 2; Время работы (75)

PRO 2 : Задача 2 была сгенерирована за (38)

PRO 2 : Задача 2 пошла на очередь

QUEUE : Пришла задача 2; Размер очереди: 11

PRO 1 : Задача 1 была сгенерирована за (44)

---

QUEUE : Пришла задача 2; Размер очереди: 12

PRO 1 : Задача 1 пошла на CPU 1, Задача 2 прервана c CPU 2 пошла в очередь // interrupt = 6

---

CPU 1 : Начал работу напрямую - задача 1; Время работы (86)

PRO 2 : Задача 2 была сгенерирована за (18)

PRO 2 : Задача 2 пошла на CPU 2

CPU 2 : Начал работу напрямую - задача 2; Время работы 77

PRO 1 : Задача 1 была сгенерирована за (16)

PRO 1 : Задача 1 удалена // CPU 1 занят, CPU 2 занят // deleted = 10

PRO 1 : задачи закончились12

PRO 2 : Задача 2 была сгенерирована за (52)

PRO 2 : Задача 2 пошла на очередь

QUEUE : Пришла задача 2; Размер очереди: 13

PRO 2 : Задача 2 была сгенерирована за (19)

PRO 2 : Задача 2 пошла на очередь

QUEUE : Пришла задача 2; Размер очереди: 14

CPU 1 : Закончил работу

QUEUE : Задача 2 удалена. Размер очереди = [13]

CPU 1 : Начал работу с очереди - задача 2; Время работы (58)

PRO 2 : Задача 2 была сгенерирована за (51)

PRO 2 : Задача 2 пошла на CPU 2

CPU 2 : Начал работу напрямую - задача 2; Время работы 45

CPU 1 : Закончил работу

QUEUE : Задача 2 удалена. Размер очереди = [12]

CPU 1 : Начал работу с очереди - задача 2; Время работы (46)

PRO 2 : Задача 2 была сгенерирована за (36)

PRO 2 : Задача 2 пошла на очередь

QUEUE : Пришла задача 2; Размер очереди: 13

PRO 2 : задачи закончились14

CPU 1 : Закончил работу

QUEUE : Задача 2 удалена. Размер очереди = [12]

CPU 1 : Начал работу с очереди - задача 2; Время работы (65)

CPU 1 : Закончил работу

QUEUE : Задача 2 удалена. Размер очереди = [11]

CPU 1 : Начал работу с очереди - задача 2; Время работы (67)

CPU 1 : Закончил работу

QUEUE : Задача 2 удалена. Размер очереди = [10]

CPU 1 : Начал работу с очереди - задача 2; Время работы (59)

CPU 1 : Закончил работу

QUEUE : Задача 2 удалена. Размер очереди = [9]

CPU 1 : Начал работу с очереди - задача 2; Время работы (39)

CPU 1 : Закончил работу

QUEUE : Задача 2 удалена. Размер очереди = [8]

CPU 1 : Начал работу с очереди - задача 2; Время работы (41)

CPU 1 : Закончил работу

QUEUE : Задача 2 удалена. Размер очереди = [7]

CPU 1 : Начал работу с очереди - задача 2; Время работы (32)

CPU 1 : Закончил работу

QUEUE : Задача 2 удалена. Размер очереди = [6]

CPU 1 : Начал работу с очереди - задача 2; Время работы (97)

CPU 1 : Закончил работу

QUEUE : Задача 2 удалена. Размер очереди = [5]

CPU 1 : Начал работу с очереди - задача 2; Время работы (32)

CPU 1 : Закончил работу

QUEUE : Задача 2 удалена. Размер очереди = [4]

CPU 1 : Начал работу с очереди - задача 2; Время работы (42)

CPU 1 : Закончил работу

QUEUE : Задача 2 удалена. Размер очереди = [3]

CPU 1 : Начал работу с очереди - задача 2; Время работы (96)

CPU 1 : Закончил работу

QUEUE : Задача 2 удалена. Размер очереди = [2]

CPU 1 : Начал работу с очереди - задача 2; Время работы (89)

CPU 1 : Закончил работу

QUEUE : Задача 2 удалена. Размер очереди = [1]

CPU 1 : Начал работу с очереди - задача 2; Время работы (89)

CPU 1 : Закончил работу

QUEUE : Задача 2 удалена. Размер очереди = [0]

CPU 1 : Начал работу с очереди - задача 2; Время работы (92)

CPU 1 : Закончил работу

-----------------------------------

Максимальная длина очереди = 14

Процент уничтоженых задач для первого потока = 50.0

Процент прерваных задач для второго потока = 30.0

**Висновки**

У даній лабораторній роботі ми ознайомитись з поняттями використання і синхронізація потоків у Java. А також проводилась розробка і реалізація програми яка моделює обслуговування двох потоків процесів з різними параметрами двома центральними процесорами комп’ютера з однією чергою. За результатами роботи програми було визначено максимальну довжину черги, процент знищених процесів для першого потоку і процент перерваних запитів другого потоку.