Частное учреждение образования

«Колледж бизнеса и права»

Тема: «Многопоточные программы на языке Java»

Учащийся М.Н. Гончаров.

группа: Т-992

2022

Выполните задания:

1) Выведете состояние потока перед его запуском, после

запуска и во время выполнения.

2) Напишите программу, в которой создаются два потока,

которые выводят на консоль своё имя по очереди.

3) Дано два потока — производитель и потребитель.

Производитель генерирует некоторые данные (в примере — числа).

Производитель «потребляет» их. Два потока разделяют общий буфер данных,

размер которого ограничен. Если буфер пуст, потребитель должен ждать,

пока там появятся данные. Если буфер заполнен полностью, производитель

должен ждать, пока потребитель заберёт данные и место освободится.

package core;

import java.util.ArrayList;

import java.util.Random;

public class ClassMain {

static forThread mThing; //mThing - объект класса, реализующего интерфейс Runnable

public static void main(String[] args)

{

// Task 1

System.out.println("[Task 1]");

mThing = new forThread();

Thread myThready = new Thread(mThing); //Создание потока "myThready"

System.out.println("Это message 1 - Перед запуском потока");

System.out.println(myThready.getState());

myThready.start(); //Запускаю поток

System.out.println("Это message 2 - После запуска потока");

System.out.println(myThready.getState());

// Task 2

System.out.println("[Task 2]");

Thread FirstThread = new Thread(new Runnable()

{

public void run()

{

while(true)

{

try

{

System.out.println("Name: FirstThread; Id [" + Thread.currentThread().getId() + "]");

Thread.sleep(2000);

}

catch (InterruptedException e)

{

e.printStackTrace();

}

}

}

});

Thread SecondThread = new Thread(new Runnable()

{

public void run()

{

while(true)

{

try

{

System.out.println("Name: SecondThread; Id [" + Thread.currentThread().getId() + "]");

Thread.sleep(2100);

}

catch (InterruptedException e)

{

e.printStackTrace();

}

}

}

});

// Что бы тестануть 3 задание не забыть закоментить, а то там в цикле кручу бесканечно

//FirstThread.start();

//SecondThread.start();

// Task 3

System.out.println("[Task 3]");

ArrayList<Integer> buffer = new ArrayList();

/\*

Дано два потока — производитель и потребитель.

Производитель генерирует некоторые данные (в примере — числа).

Производитель «потребляет» их. Два потока разделяют общий буфер данных,

размер которого ограничен. Если буфер пуст, потребитель должен ждать,

пока там появятся данные. Если буфер заполнен полностью, производитель

должен ждать, пока потребитель заберёт данные и место освободится.

\*/

Thread myThread\_1 = new Thread(new Runnable()

{

public void run()

{

while(true)

{

try

{

Thread.sleep(5000);

Random rnd = new Random();

buffer.add(rnd.nextInt(100));

}

catch (InterruptedException e)

{

e.printStackTrace();

}

}

}

});

Thread myThread\_2 = new Thread(new Runnable()

{

public void run()

{

while(true)

{

try

{

Thread.sleep(1000);

if (!buffer.isEmpty())

{

System.out.println("Удалось считать буферную переменну: " + buffer.get(0));

buffer.clear();

}

else

{

System.out.println("Переменная пока что не доступна");

}

}

catch (InterruptedException e)

{

e.printStackTrace();

}

}

}

});

//myThread\_1.start();

//myThread\_2.start();

}

}

// for thread

**package** core;

**public** **class** forThread

**implements** Runnable //(содержащее метод run())

{

**public** **void** run() //Этот метод будет выполняться в побочном потоке

{

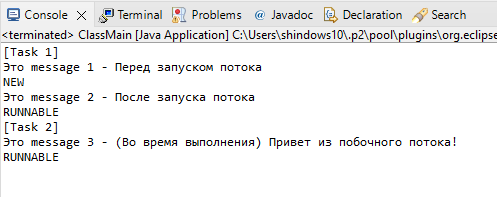
System.***out***.println("Это message 3 - (Во время выполнения) Привет из побочного потока!");

System.***out***.println(Thread.*currentThread*().getState()); // Вывод состояния текущего потока

}

}

// 1, 2



// 3

