БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

ИИТ БГУИР

Программное обеспечение информационных технологий.

Дисциплина: «Сетевые технологии и администрирование операционных систем»

Отчет

По лабораторной работе №2

«ThreadPool»

Выполнил студент группы 681072: Иванов К.Е.

Проверил преподаватель: Калитеня И.Л.

Минск 2019

**Задание:** класс "ThreadPool"Разработать переносимый (portable) класс ThreadPool.Объект этого класса содержит N потоков, находящихся в состоянии ожидания. К объекту можно обратиться с запросом на выполнение некоторого метода в потоке. Для выполнения процедуры объект класса ThreadPool подключает поток из пула ожидающих потоков. При создании объекта класса ThreadPool вводится максимальное число создаваемых потоков (K, K >= N).

Предусмотреть логгирование процесса работы ThreadPool. Файл логов должен включать в себя следующие сообщения:

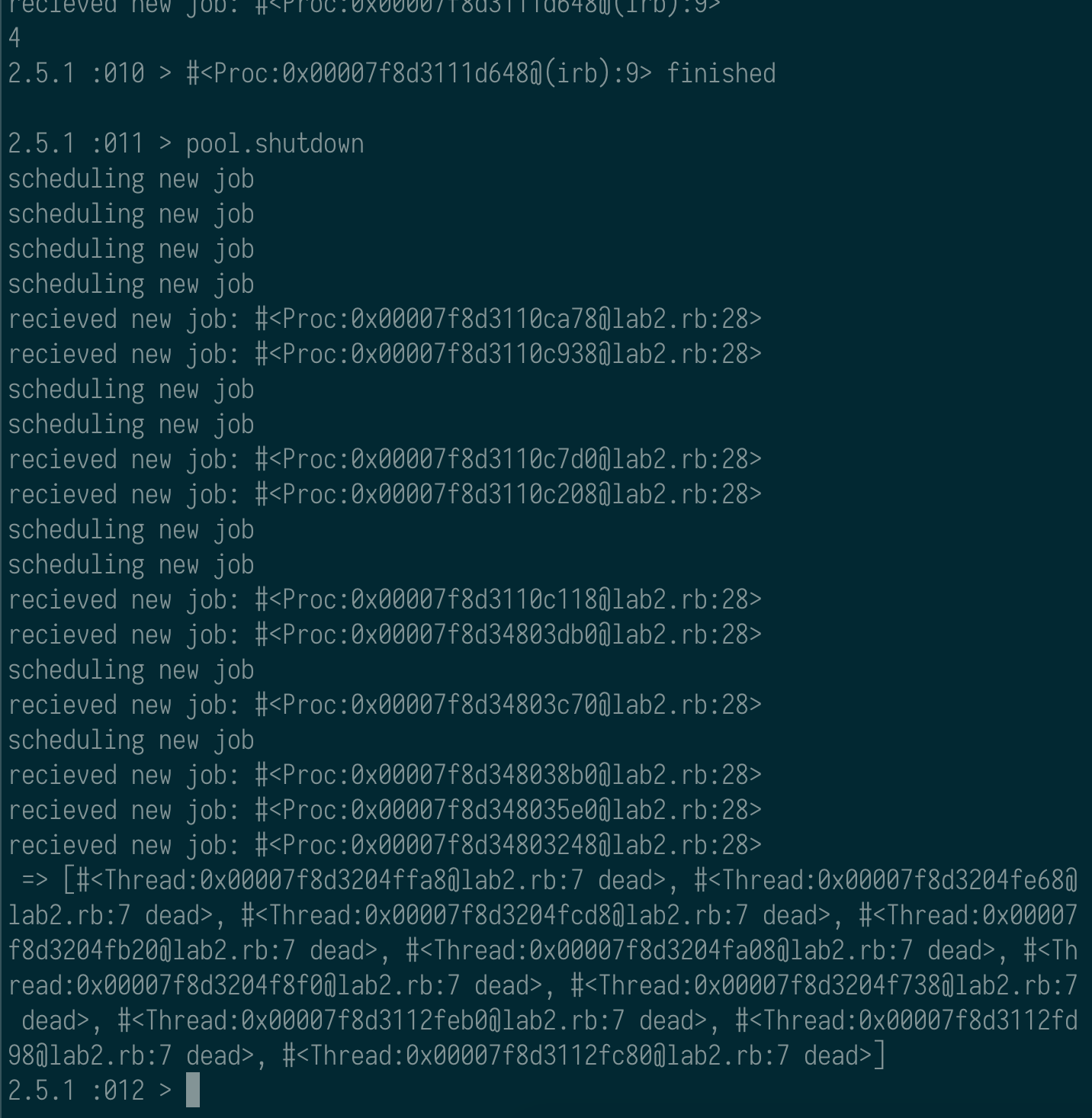
1) создание N потоков при инициализации ThreadPool;

2) сообщение при добавлении новой задачи;

3) сообщения об ошибке выполнения задачи;

4) предупреждения, полученные при работе ПС.

В случае возникновения ошибок при работе пользовательской функции, сообщение об ошибке выводится в файл логов. Поток завершает обработку пользовательской функции и освобождается.



**Код приложения:**

using System;

using System.IO;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading;

namespace L02

{

class Program

{

public const string LogFile = "log.txt";

static void LogClear()

{

File.Delete(LogFile);

}

static void LogWrite(string StrLog)

{

File.AppendAllText(LogFile, StrLog+"\n");

}

static string LogRead()

{

try

{

return File.ReadAllText(LogFile, Encoding.Default);

}

catch (Exception)

{

File.AppendAllText(LogFile, "create log file ["+DateTime.Now.ToString() + "]\n");

return "create log file [" + DateTime.Now.ToString() + "]\n";

}

}

static void ShowLogFile()

{

Console.WriteLine("LOG FILE");

Console.WriteLine(LogRead());

Console.WriteLine("END LOG FILE");

}

static public int MainMenu ()

{

Console.Clear();

Console.WriteLine("MENU");

Console.WriteLine("(1) Show LOG FILE");

Console.WriteLine("(2) Create ThreadPool");

Console.WriteLine("(3) ADD Thread");

Console.WriteLine("(4) Work Thread");

Console.WriteLine("(5) Clear log file");

Console.WriteLine("(6) INFO");

Console.WriteLine("(0) EXIT Program");

Console.Write("Select menu:");

return int.Parse(Console.ReadLine());

}

static int ContinueWork()

{

string pMenu;

do

{

Console.Write("continue work (y/n):");

pMenu = Console.ReadLine();

if (pMenu == "y")

{

return 1;

}

if (pMenu == "n")

{

return 0;

}

} while (true);

return 0;

}

static void Main(string[] args)

{

int CountThreads = 0;

int mMenu = 0;

int nWorkerThreads = 0;

int nCompletionThreads = 0;

do

{

mMenu = MainMenu();

if (mMenu == 1) {

Console.Clear();

Console.WriteLine(" 1. Show LOG FILE");

ShowLogFile();

mMenu = ContinueWork();

}

if (mMenu == 2) {

Console.Clear();

Console.WriteLine(" 2. Create ThreadPool");

Console.Write("Enter count Thread:");

CountThreads = int.Parse(Console.ReadLine());

ThreadPool.SetMaxThreads(CountThreads, CountThreads);

ThreadPool.GetMaxThreads(out nWorkerThreads, out nCompletionThreads);

Console.WriteLine("Максимальное количество потоков: " + nWorkerThreads + "\nПотоков ввода-вывода доступно: " + nCompletionThreads);

LogWrite("Max cont Threads: [" + nWorkerThreads + "] Threads: [" + nCompletionThreads + "]");

mMenu = ContinueWork();

}

if (mMenu == 3)

{

Console.Clear();

Console.WriteLine("3. ADD Thread");

CountThreads++;

ThreadPool.SetMaxThreads(CountThreads, CountThreads);

Console.WriteLine("Максимальное количество потоков: " + CountThreads + "\nПотоков ввода-вывода доступно: " + CountThreads);

LogWrite("Add Threads");

mMenu = ContinueWork();

}

if (mMenu == 4){

Console.Clear();

Console.WriteLine(" 4. Work Thread");

Console.WriteLine("Максимальное количество потоков: " + nWorkerThreads + "\nПотоков ввода-вывода доступно: " + nCompletionThreads);

Console.WriteLine("---------------------------------------");

for (int i = 0; i < CountThreads; i++)

{

try

{

ThreadPool.QueueUserWorkItem(new WaitCallback(JobForAThread), i + 1);

}

catch (Exception)

{

LogWrite("add Error - " + (i + 1).ToString());

}

}

if (CountThreads>0)Thread.Sleep(5000);

}

if (mMenu == 5)

{

LogClear();

Console.WriteLine("Delete data log.");

mMenu = ContinueWork();

}

if (mMenu == 6)

{

Console.Clear();

Console.WriteLine(" 6. INFO");

Console.WriteLine("Максимальное количество потоков: " + nWorkerThreads + "\nПотоков ввода-вывода доступно: " + nCompletionThreads);

mMenu = ContinueWork();

}

}

while (mMenu != 0);

}

static void JobForAThread(object state)

{

Console.WriteLine("Выполняю поток - {0}", state);

Thread.Sleep(5000);

}

}

}