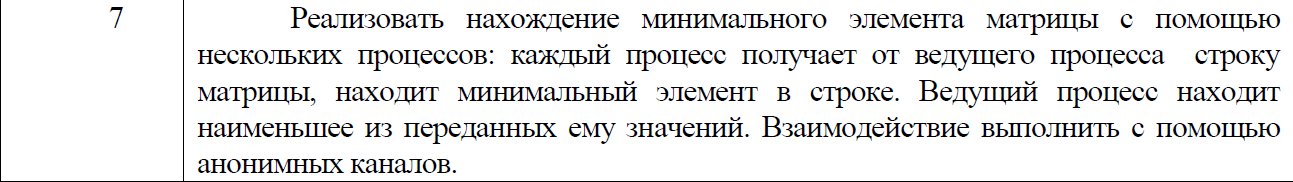
**Лабораторная работа №6**

Межпроцессное взаимодействие на примере анонимных каналов



using System;

using System.IO;

using System.IO.Pipes;

using System.Diagnostics;

using System.Collections.Generic;

using System.Linq;

namespace SPS\_Lab\_6

{

class Program

{

static void Main(string[] args)

{

Console.WriteLine("Enter the dimensions of matrix:");

var n = int.Parse(Console.ReadLine());

var m = int.Parse(Console.ReadLine());

Random rnd = new Random();

int[,] baseArray = new int[n, m];

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

{

for (int j = 0; j < m; j++)

{

baseArray[i, j] = rnd.Next(0, 100);

}

}

var proc = new Process();

proc.StartInfo.FileName = @"..\..\..\pipeClient\bin\Debug\pipeClient.exe";

using (var pipeServer = new AnonymousPipeServerStream(PipeDirection.Out, HandleInheritability.Inheritable))

{

using (var pipeServerIn = new AnonymousPipeServerStream(PipeDirection.In, HandleInheritability.Inheritable))

{

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

{

proc.StartInfo.Arguments = pipeServer.GetClientHandleAsString() + " " + pipeServerIn.GetClientHandleAsString() + " " + i.ToString();

proc.StartInfo.UseShellExecute = false;

proc.Start();

}

pipeServer.DisposeLocalCopyOfClientHandle();

List<string> strokes = new List<string>();

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

{

var stroke = "";

for (int j = 0; j < m; j++)

{

if (j == m - 1)

{

stroke += baseArray[i, j].ToString();

}

else

{

stroke += baseArray[i, j].ToString() + ",";

}

}

strokes.Add(stroke);

}

try

{

using (StreamWriter sw = new StreamWriter(pipeServer))

{

sw.AutoFlush = true;

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

{

sw.WriteLine(i.ToString() + "," + strokes[i]);

pipeServer.WaitForPipeDrain();

}

}

}

catch (IOException e)

{

Console.WriteLine("[SERVER] Error: {0}", e.Message);

}

string[] temp=new string[n];

using (StreamReader sr = new StreamReader(pipeServerIn))

{

lock (sr)

{

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

{

temp[i] = sr.ReadLine();

}

}

}

List<int> numbers = new List<int>();

foreach (string str in temp)

{

numbers.Add(int.Parse(str));

}

Console.WriteLine($"Минимальное число матрицы:{numbers.Min()}");

}

Console.WriteLine("[SERVER] Client quit. Server terminating.");

Console.ReadKey();

}

}

}

}

using System.IO;

using System.IO.Pipes;

namespace pipeClient

{

class Program

{

static void Main(string[] args)

{

if (args.Length > 0)

{

using (PipeStream pipeClient = new AnonymousPipeClientStream(PipeDirection.In, args[0]))

{

using (var pipeClientOut = new AnonymousPipeClientStream(PipeDirection.Out, args[1]))

{

string temp;

using (var sr = new StreamReader(pipeClient))

{

do

{

temp = sr.ReadLine();

}while (!temp.StartsWith(args[2]));

}

string[] stroke;

stroke=temp.Split(',');

int min = int.Parse(stroke[1]);

for(int i=1;i<stroke.Length;i++)

{

if (int.Parse(stroke[i]) < min)

min = int.Parse(stroke[i]);

}

using (var bw = new StreamWriter(pipeClientOut))

{

bw.AutoFlush = true;

bw.WriteLine(min);

pipeClientOut.WaitForPipeDrain();

}

}

}

}

}

}

}

