**НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ УКРАИНЫ**

**«КИЕВСКИЙ ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ»**

**Лабораторная работа №3**

**по дисциплине**

**«Операционные системы»**

**Выполнил:**

**Ст. гр. ИВ-02, ФИВТ**

**Шпурик Владислав**

## Киев-2013

**Priority Queue.**

# 1. *Описание разработанного алгоритма:*

Каждый пакет имеет два параметра:

* Количество тактов для обработки.
* Приоритет( чем меньше приоритет, тем раньше пакет будет обработан).

Очередь имеет фиксированную длину, при появление нового пакета, он упорядочено добавляется в очередь. Соответственно если очередь полная, и пришел пакет с меньшим приоритетом чем у последнего пакета в очереди, он отбрасывается.

Главный недостаток приоритетной очереди – не гарантируется что пакет будет когда-либо обработан.

**Листинг программы**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace os\_lab\_3\_cs

{

class Package

{

public Package(int t, int p)

{

Time = t;

Priority = p;

}

public int Time {get; set;}

public int Priority {get; set;}

}

class PriorityQueue

{

public int size;

public int avt = 0;

public int dt = 0;

public int number = 0;

List<Package> pq = new List<Package>();

public PriorityQueue(int size)

{

this.size = size;

}

public void tick()

{

avt += pq.Count;

if (pq.Count == 0)

{

dt++;

}

else

{

pq[0].Time--;

if (pq[0].Time == 0)

{

pq.RemoveAt(0);

}

}

}

public void add(Package pc)

{

if (pq.Count < size)

{

number++;

int idx = 0;

for (; idx < pq.Count; idx++)

{

if (pq[idx].Priority > pc.Priority)

{

break;

}

}

pq.Insert(idx, pc);

}

}

}

class GenFactory

{

public static IEnumerator<Package> getGenerator(int minTime, int maxTime,

int minPriority, int maxPriority)

{

Random rnd = new Random();

while (true)

{

yield return new Package(rnd.Next(minTime, maxTime),

rnd.Next(minPriority, maxPriority));

}

}

}

}

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Windows.Forms;

namespace os\_lab\_3\_cs

{

public partial class Form1 : Form

{

public Form1()

{

InitializeComponent();

}

private void button1\_Click(object sender, EventArgs e)

{

chart1.Series.Add("");

for (int intensity = 30; intensity > 0 ; intensity--)

{

int delay = intensity;

var gen = GenFactory.getGenerator(3, 10, 0, 10);

var rnd = new Random();

var pq = new PriorityQueue(30);

for (int i = 0; i < 10000; ++i)

{

if (--delay == 0)

{

gen.MoveNext();

pq.add(gen.Current);

delay = intensity;

}

pq.tick();

}

int tavrg = pq.avt / pq.number;

chart1.Series[0].Name = "Среднее время ожидания от интенсивности";

chart1.Series[0].ChartType = System.Windows.Forms.DataVisualization.Charting.SeriesChartType.Line;

chart1.Series[0].Points.AddXY((intensity), tavrg);

chart1.Series[1].Name = "Среднее время простоя от интенсивности";

chart1.Series[1].ChartType = System.Windows.Forms.DataVisualization.Charting.SeriesChartType.Line;

chart1.Series[1].Points.AddXY(( intensity), pq.dt/100.0);

}

}

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Windows.Forms;

namespace os\_lab\_3\_cs

{

static class Program

{

/// <summary>

/// The main entry point for the application.

/// </summary>

[STAThread]

static void Main()

{

Application.EnableVisualStyles();

Application.SetCompatibleTextRenderingDefault(false);

Application.Run(new Form1());

}

}

}

Выводы: когда интенсивность большая, пакеты приходят часто, система не успевает их обрабатывать, и время ожидания очень большое, а система постоянно загружена, чем реже приходят пакеты тем больше шансы что система успеет все обработать, и время простоя увеличивается.