**МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ**

**УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ**

**ГОМЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ П. О. СУХОГО**

Факультет автоматизированных и информационных систем

Кафедра «Информационные технологии»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №10

по дисциплине **«**Операционные системы»

на тему: «Управление виртуальной памятью.

Алгоритмы замещения страниц. Управление виртуальной памятью.

Алгоритмы замещения страниц»

Выполнил: студент гр. ИТП-11

Косиченко Д.А.

Принял: преподаватель-стажёр

Карась О.В.

Гомель 2022

**Цель работы:** изучить основные алгоритмы замещение страниц и работой с виртуальной памятью.

**Задание**

Разработать программу, реализующую заданный алгоритм замещения страниц в памяти.

Менеджер памяти должен:

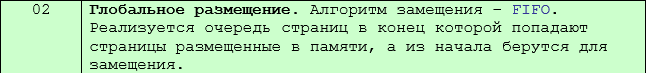
1. Разбивать память заданного размера на указанное количество страниц. На экран должна выводиться следующая информация о состоянии памяти: объем памяти, число страниц, число свободных страниц (%), размер страницы;

2. Размещать в памяти страницу заданного процесса, с замещением занятой по заданному алгоритму (по нажатию кнопки «ДОБАВИТЬ»). Для размещения страницы в памяти, указывается имя процесса и ее номер (вводятся отдельно). Например: Pro 3. После нажатия на кнопку «ДОБАВИТЬ» страница размещается в свободной странице памяти. Если задано глобальное размещение (см. вариант задания), то выбирается любая не занятая страница. При локальном размещении страница размещается только среди виртуальных страниц, выделенных этому процессу. Выделение страниц в памяти выполняется при первом ее занесении процесса в память. Алгоритм замещения выполняется только при отсутствии свободных страниц под процесс;

3. Удалять из памяти заданную страницу или все страницы заданного процесса (по нажатию кнопки «УДАЛИТЬ»). Указывается номер удаляемой страницы в памяти;

4. Организовывать циклическое обращение к страницам, размещенным в памяти по нажатию на кнопку. При этом случайным образом задается количество обращений к страницам (диапазон 1...10). Для каждого обращения генерируется, случайным образом, номер страницы из диапазона [0; количество страниц памяти]. При обращении к странице в зависимости, от варианта, увеличивается ее внутренний счетчик обращений или устанавливается флаг обращения.





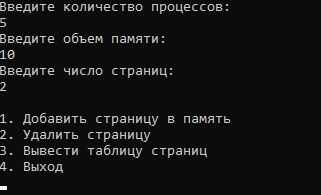


Рисунок 1– Файловый менеджер

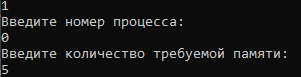


Рисунок 2– Добавление процесса

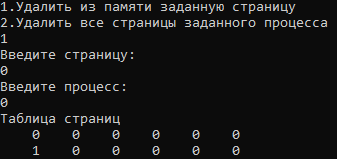


Рисунок 3– Удаление процесса

**Вывод:** в ходе лабораторной работы были изучены основные алгоритмы замещение страниц и работа с виртуальной памятью.

**Приложение 1**

**Листинг программы**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

namespace Laba10

{

class Program

{

static void Main(string[] args)

{

/\*

Разбивать память заданного размера на указанное количество страниц. На экран должна выводиться следующая информация о состоянии памяти:

объем памяти, число страниц, число свободных страниц (%), размер страницы;

\*/

int pm, kol, k = 0, del = 0, j = 0, flag = -1, v = 0, proc, f, pust = 0;

double obem;

int ind, ch = 0;

Console.WriteLine("Введите количество процессов:");

proc = Convert.ToInt32(Console.ReadLine());

Console.WriteLine("Введите объем памяти:");

pm = Convert.ToInt32(Console.ReadLine());

Console.WriteLine("Введите число страниц:");

kol = Convert.ToInt32(Console.ReadLine());

obem = pm / kol;

Console.WriteLine();

double[,] A = new double[proc,kol];

double[,] B = new double[2,100];

int[] C = new int[proc];

while (k != 4)

{

Console.WriteLine("1. Добавить страницу в память");

Console.WriteLine("2. Удалить страницу");

Console.WriteLine("3. Вывести таблицу страниц");

Console.WriteLine("4. Выход");

try

{

k = Convert.ToInt32(Console.ReadLine());

}

catch (FormatException)

{

k = 3;

}

switch (k)

{

case 1:

{

Console.WriteLine("Введите номер процесса:");

B[0, j] = Convert.ToDouble(Console.ReadLine());

Console.WriteLine("Введите количество требуемой памяти:");

B[1, j] = Convert.ToDouble(Console.ReadLine());

j++;

break;

}

case 2:

{

Console.WriteLine("1.Удалить из памяти заданную страницу");

Console.WriteLine("2.Удалить все страницы заданного процесса");

int p = Convert.ToInt32(Console.ReadLine());

try

{

if (p == 1)

{

Console.WriteLine("Введите страницу:");

ind = Convert.ToInt32(Console.ReadLine());

Console.WriteLine("Введите процесс:");

del = Convert.ToInt32(Console.ReadLine());

A[del, ind] = 0;

for (f = 0; f < kol; f++)

{

Console.Write("{0,5}", f);

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

{

Console.Write("{0,5}", A[i, f]);

}

Console.WriteLine();

}

Console.WriteLine("Общий объем памяти:{0}", pm \* proc);

Console.WriteLine("Объем памяти на каждый процесс:{0}", pm);

Console.WriteLine("Общее число страниц:{0}", kol \* proc);

Console.WriteLine("Число страниц для каждого процесса:{0}", kol);

Console.WriteLine("Число свободных страниц во всех стрннах:{0}", pust);

Console.WriteLine("Размер страници:{0}", obem);

}

else if (p == 2)

{

Console.WriteLine("Введите процесс:");

del = Convert.ToInt32(Console.ReadLine());

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

{

A[del,i] = 0;

}

C[del] = 0;

}

}

catch (IndexOutOfRangeException)

{

Console.Write("Нет такой страници");

}

break;

}

case 3:

{

Console.Clear();

ch = 0;

while (ch < proc)

{

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

{

if (ch == B[0, h])

{

flag = h;

break;

}

}

for(f=0;f< kol; f++)

{

if (A[ch, f] == 0 && flag >= 0)

{

A[ch, f] = B[1, flag];

B[1, flag] = -1;

B[0, flag] = -1;

v = 1;

break;

}

}

for (f = 0; f < kol; f++)

{

if (flag >= 0 && v == 0)

{

A[ch, C[ch]] = B[1, flag];

B[1, flag] = -1;

B[0, flag] = -1;

C[ch]++;

if (C[ch] == kol)

{

C[ch] = 0;

}

break;

}

}

flag = -1;

ch++;

v = 0;

}

for (f = 0; f < kol; f++)

{

Console.Write("{0,5}", f);

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

{

Console.Write("{0,5}",A[i, f]);

}

Console.WriteLine();

}

pust = 0;

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

{

for (f = 0; f < kol; f++)

{

if (A[i, f] == 0) pust++;

}

}

Console.WriteLine("Общий объем памяти:{0}", pm\*proc);

Console.WriteLine("Объем памяти на каждый процесс:{0}",pm);

Console.WriteLine("Общее число страниц:{0}",kol\*proc);

Console.WriteLine("Число страниц для каждого процесса:{0}",kol);

Console.WriteLine("Число свободных страниц во всех стрннах:{0}",pust);

Console.WriteLine("Размер страници:{0}",obem);

break;

}

case 4:

{

break;

}

}

}

}

}

}