Учреждение образования Республики Беларусь

«Гомельский государственный технический университет им. П. О. Сухого»

Факультет автоматизированных и информационных систем

Кафедра «Информационные технологии»

Отчет по лабораторной работе №10

по дисциплине «Операционные системы»

Выполнил: студент группы ИТП-11

Половцев М. С.

Принял: преподаватель-стажер

Карась О. В.

Гомель, 2022

**Цель:** изучение алгоритмов замещения страниц, разработка программы

замещения страниц.

**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;

Вариант задания:

|  |  |
| --- | --- |
| Вариант | Задание |
| 6 | Локальное размещение. Алгоритм замещения – Random.  Замещается случайная страница. |

Если нам нужно иметь стратегию выталкивания страниц, которая характеризовалась бы малыми издержками и не являлась бы дискриминационной по отношению к каким-либо конкретным пользователям, то можно пойти по очень простому пути – выбирать случайную страницу (алгоритм Random). В этом случае все страницы, находящиеся в основной памяти, могут быть выбраны для выталкивания с равной вероятностью, в том числе даже следующая страница, к которой будет производиться обращение (и которую, естественно, удалять из памяти наиболее нецелесообразно). Поскольку подобная стратегия по сути, как бы рассчитана на “слепое” везение, в реальных системах она применяется редко.

На рисунке 1 представлен результат работы алгоритма замещения Random:

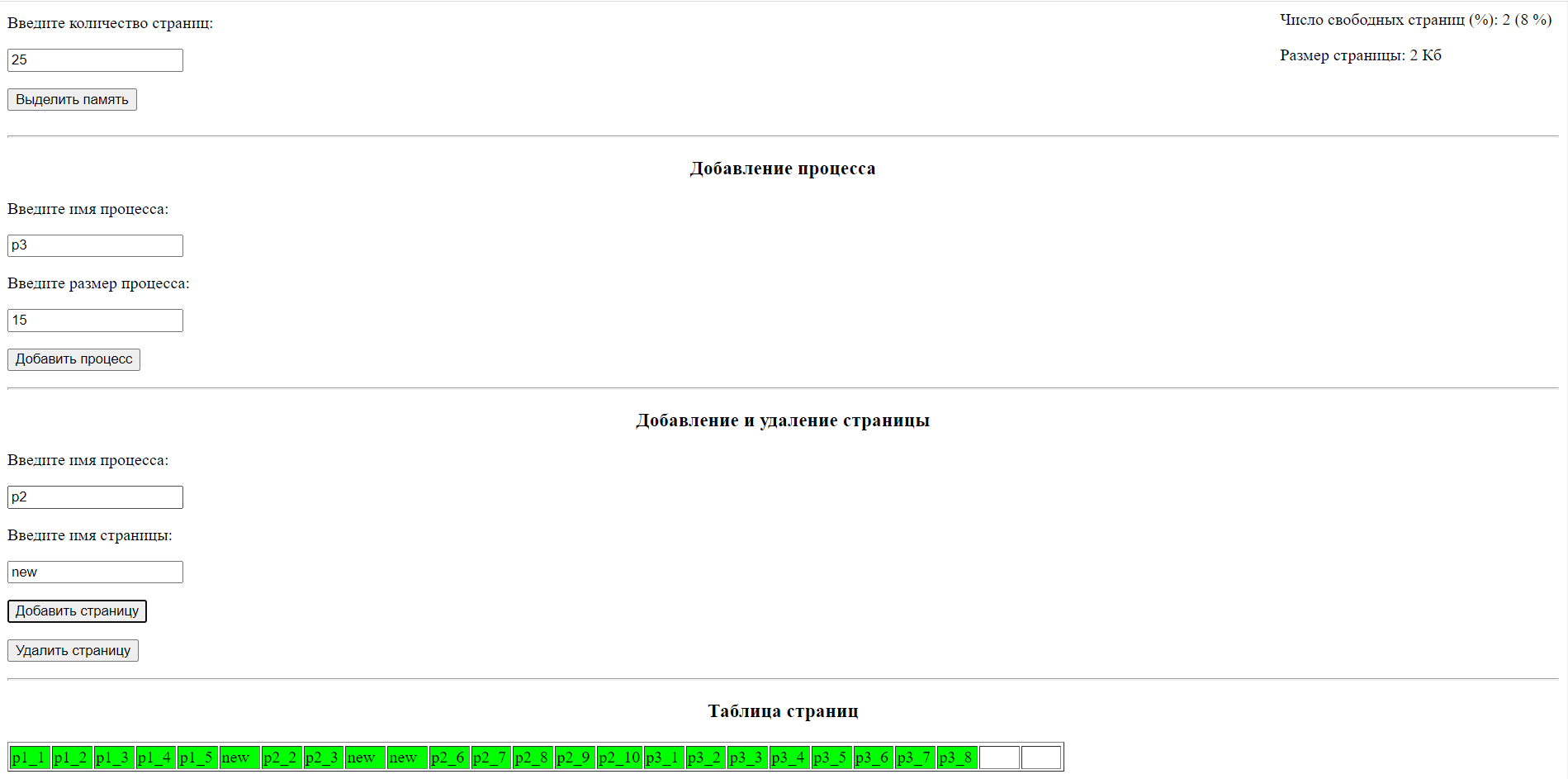


Рисунок 1 – Результат работы алгоритма замещения Random

**Вывод:** изучил алгоритмы замещения страниц, разработал программу

замещения страниц.

Приложение А

Листинг программы

let pageSize;

let freeSize;

function allocMemory() {

let size = parseInt(document.getElementById('inpSizeOfMemory').value);

let number = parseInt(document.getElementById('inpNumOfPages').value);

if (size % number === 0) {

pageSize = size / number;

freeSize = number;

let per = freeSize / number \* 100;

arrInput();

document.getElementById('span1').innerText = ` ${size} Кб`;

document.getElementById('span2').innerText = ` ${number}`;

document.getElementById('span3').innerText = ` ${freeSize} (${per} %)`;

document.getElementById('span4').innerText = ` ${pageSize} Кб`;

}

else {

alert('Размер памяти должен делиться нацело на количество страниц');

}

}

let allocButton = document.getElementById('allocButton');

allocButton.addEventListener('click', allocMemory);

let pages = [];

function arrInput() {

pages.length = parseInt(document.getElementById('inpNumOfPages').value);

for (let i = 0; i < pages.length; i++) {

pages[i] = 0;

}

}

class page {

name;

size;

start;

}

let processes = [];

function addProcess() {

let nameOfProc = document.getElementById('addNameOfProc').value;

let sizeInKbs = parseInt(document.getElementById('addSizeOfProc').value);

let sizeInPages = Math.floor(sizeInKbs / pageSize);

if (sizeInKbs % pageSize !== 0) {

sizeInPages++;

}

if (sizeInPages <= freeSize) {

processes.push(new page);

processes[processes.length - 1].name = nameOfProc;

processes[processes.length - 1].size = sizeInPages;

for (let i = 0; i < pages.length; i++) {

if (pages[i] === 0) {

processes[processes.length - 1].start = i;

let k = 0;

for (let j = i; j < i + sizeInPages; j++) {

k++;

pages[j] = `${nameOfProc}\_${k}`;

}

break;

}

}

freeSize -= sizeInPages;

let per = freeSize / document.getElementById('inpNumOfPages').value \* 100;

document.getElementById('span3').innerText = ` ${freeSize} (${per} %)`;

createTable();

}

else {

alert('Недостаточно памяти для добавления процесса');

}

}

let addProcButton = document.getElementById('addProc');

addProcButton.addEventListener('click', addProcess);

function addPage() {

let nameOfProc = document.getElementById('NameOfProc').value;

let nameOfPage = document.getElementById('NameOfPage').value;

let num;

for (let i = 0; i < processes.length; i++) {

if (nameOfProc === processes[i].name) {

num = i;

break;

}

}

let f = 0;

for (let i = processes[num].start; i < processes[num].start + processes[num].size; i++) {

if (pages[i] === 0) {

pages[i] = nameOfPage;

f = 1;

freeSize--;

let per = freeSize / document.getElementById('inpNumOfPages').value \* 100;

document.getElementById('span3').innerText = ` ${freeSize} (${per} %)`;

break;

}

}

if (f === 0) {

let k = 0;

while (k !== processes[num].size) {

let rand = randomInt(processes[num].start, processes[num].start + processes[num].size - 1);

if (pages[rand] !== nameOfPage) {

pages[rand] = nameOfPage;

break;

}

k++;

}

}

createTable();

}

let addPageButton = document.getElementById('addPage');

addPageButton.addEventListener('click', addPage);

function randomInt(min, max) {

let rand = min - 0.5 + Math.random() \* (max - min + 1);

return Math.round(rand);

}

function deletePage() {

let nameOfProc = document.getElementById('NameOfProc').value;

let nameOfPage = document.getElementById('NameOfPage').value;

let f = 0;

for (let i = 0; i < processes.length; i++) {

if (processes[i].name === nameOfProc) {

freeSize++;

let per = freeSize / document.getElementById('inpNumOfPages').value \* 100;

document.getElementById('span3').innerText = ` ${freeSize} (${per} %)`;

for (let j = processes[i].start; j < processes[i].start + processes[i].size; j++) {

if (pages[j] === nameOfPage) {

pages[j] = 0;

break;

}

}

f = 1;

createTable();

break;

}

}

if (f === 0) {

alert('Страницы с таким именем не существует');

}

}

let delPageButton = document.getElementById('delPage');

delPageButton.addEventListener('click', deletePage);

function createTable() {

document.getElementById('table').innerHTML = '';

let table = document.createElement('table');

table.setAttribute('border', '1');

table.innerHTML = '<tr>';

for (let i = 0; i < pages.length; i++)

{

let td = document.createElement('td');

td.setAttribute('width', `35px`);

if (pages[i] !== 0)

{

td.setAttribute('bgcolor', `#00FF00`);

td.innerText = pages[i];

}

table.append(td);

}

table.innerHTML += '</tr>';

document.getElementById('table').append(table);

}