KONGU ENGINEERING COLLEGE
(Autonomous)
Perundurai, Erode – 638 060

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

PAGE REPLACEMENT ALGORITHM

A MINI PROJECT REPORT

for

OPERATING SYSTEMS (22CST43)

Submitted by

Abinaya Sri J 23CSR010

Chandru K 23CSR037

Danush P 23CSR038

FACULTY INCHARGE

Dr.N.SHANTHI

PROFESSOR/CSE

CODE

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="UTF-8" />
<meta name="viewport" content="width=device-width, initial-scale=1.0"/>
<title>Page Fault Simulator</title>
<style>
:root {
--primary: #00d4ff;
--primary-dark: #0097c4;
--accent: #ff6e00;
--text: #e0f7fa;
--dark-bg: #0f1923;
--card-bg: rgba(15, 25, 35, 0.85);
--success: #00e676;
--warning: #ff9100;
}
* {
box-sizing: border-box;
margin: 0;
padding: 0;
}

body {
font-family: 'Segoe UI', system-ui, -apple-system, sans-serif;
background: url('https://images.unsplash.com/photo-1518770660439-4636190af475?ixlib=rb-1.2.1&auto=format&fit=crop&w=1950&q=80') no-repeat center center fixed;
background-size: cover;
color: var(--text);
```

```
line-height: 1.6;
min-height: 100vh;
position: relative;
}
body::before {
content: ";position: absolute;top: 0;
left: 0;width: 100%;
height: 100%;
background: linear-gradient(135deg, rgba(0,0,0,0.8) 0%, rgba(2,20,35,0.85) 100%);
z-index: -1;
}
header {
background: var(--dark-bg);
padding: 0;
position: sticky;
top: 0;
z-index: 100;
box-shadow: 0 4px 12px rgba(0,0,0,0.15);
border-bottom: 1px solid rgba(0, 212, 255, 0.2);
}
nav {
max-width: 1200px;
margin: 0 auto;
padding: 18px 24px;display: flex;
justify-content: space-between;
align-items: center;}
.logo {
display: flex;align-items: center;
gap: 12px;color: var(--primary);
text-decoration: none;}
```

```
.logo-icon {
font-size: 1.8rem;
}
.logo h1 {
margin: 0;font-size: 1.5rem;font-weight: 600;letter-spacing: 0.5px;
}
nav ul {
display: flex;list-style: none;gap: 24px;
}
nav a {
color: var(--text);text-decoration: none;
font-weight: 500;font-size: 1rem;
transition: all 0.2s ease;padding: 8px 12px;
border-radius: 4px;
}
nav a:hover {
color: var(--primary); background: rgba(0, 212, 255, 0.1);
}
nav a.active {
color: var(--primary); border-bottom: 2px solid var(--primary);
}
main {
max-width: 1000px;
margin: 40px auto;
padding: 0 20px;
}
.container {
background: var(--card-bg);
border-radius: 12px;
padding: 32px;
```

```
margin-bottom: 40px;
box-shadow: 0 8px 24px rgba(0, 0, 0, 0.2);
backdrop-filter: blur(10px);
border: 1px solid rgba(255, 255, 255, 0.05);
animation: fadeIn 0.6s ease-out;}
@keyframes fadeIn {
from { opacity: 0; transform: translateY(20px); }
to { opacity: 1; transform: translateY(0); }
}
.section-title {
display: flex;
align-items: center;gap: 12px;
margin-bottom: 24px;
padding-bottom: 16px;
border-bottom: 1px solid rgba(255, 255, 255, 0.1);
}
.section-title h2 {
color: var(--primary);
font-size: 1.8rem;
font-weight: 600;
}
.form-group {
margin-bottom: 20px;
}
label {
display: block;
font-weight: 500;
margin-bottom: 8px;
color: var(--primary);
}
```

```
input[type="text"],
input[type="number"] {
width: 100%;
padding: 12px 16px;
background: rgba(255, 255, 255, 0.05);
border: 1px solid rgba(255, 255, 255, 0.1);
border-radius: 8px;
color: var(--text);
font-size: 1rem;
transition: all 0.3s ease;
}

input[type="text"]:focus,
input[type="number"]:focus {
outline: none;
border-color: var(--primary);
box-shadow: 0 0 0 3px rgba(0, 212, 255, 0.2);
}
input::placeholder {
color: rgba(224, 247, 250, 0.5);
}
.btn-group {
display: flex;
gap: 12px;
margin-top: 24px;
}
button {
flex: 1;
padding: 12px 20px;
border: none;
```

```
border-radius: 8px;
font-size: 1rem;
font-weight: 600;
cursor: pointer;
transition: all 0.2s ease;
display: flex;
justify-content: center;
align-items: center;gap: 8px;
}

.btn-primary {
background: var(--primary);
color: var(--dark-bg);
}
.btn-primary:hover {
background: var(--primary-dark);
transform: translateY(-2px);
box-shadow: 0 4px 12px rgba(0, 212, 255, 0.3);
}
.result-container {
margin-top: 32px;
overflow: hidden;
}

.result-header {
display: flex;
justify-content: space-between;
align-items: center;
margin-bottom: 16px;
}
```

```
.result-title {
font-size: 1.2rem;
color: var(--primary);
font-weight: 600;
}
pre {
background: rgba(0, 0, 0, 0.3);
padding: 20px;
border-radius: 8px;
white-space: pre-wrap;
overflow-x: auto;
color: var(--text);
border-left: 4px solid var(--primary);
font-family: 'Consolas', 'Monaco', monospace;
line-height: 1.6;
max-height: 400px;
overflow-y: auto;
}

.highlight-hit {
color: var(--success);
font-weight: bold;
}
.highlight-fault {
color: var(--warning);
font-weight: bold;
}
.about-section {
display: none;
}
```

```
.about-section h3 {
font-size: 1.5rem;
color: var(--primary);
margin-bottom: 16px;
}
.about-section h4 {
font-size: 1.25rem;
color: var(--primary);
margin-top: 24px;
margin-bottom: 12px;
}
.about-section p {
margin-bottom: 16px;
line-height: 1.7;
}

.about-section pre {
margin: 16px 0;
padding: 16px;
background: rgba(0, 0, 0, 0.3);
border-radius: 8px;
white-space: pre-wrap;
overflow-x: auto;
color: var(--text);
border-left: 4px solid var(--accent);
}
@media (max-width: 768px) {
.btn-group {
flex-direction: column;
}
```

```
.container {
padding: 24px;
}
.footer {
background: var(--dark-bg);
text-align: center;
padding: 20px;
margin-top: 40px;
border-top: 1px solid rgba(0, 212, 255, 0.2);
}
footer p {
color: rgba(224, 247, 250, 0.6);
font-size: 0.9rem;
}
@keyframes pulse {
0% { opacity: 1; }
50% { opacity: 0.6; }
100% { opacity: 1; }
}
.algorithm-info {
background: rgba(0, 212, 255, 0.1);
border-left: 4px solid var(--primary);
padding: 16px;
margin-bottom: 24px;
border-radius: 4px;
}
.algorithm-info h4 {
margin-top: 0;
color: var(--primary);
}
```

```
.algorithm-info p {
margin-bottom: 0;
}
</style>
</head>
<body>
<header>
<nav>
<a href="#" class="logo">
<span class="logo-icon">⚙</span>
<h1>Page Fault Simulator</h1>
</a>
<ul>
<li><a href="#" id="home-nav" class="active" onclick="showHome()">Simulator</a></li>
<li><a href="#" id="about-nav" onclick="showAbout()">About</a></li>
</ul>
</nav>
</header>

<main>
<div class="container" id="home-section">
<div class="section-title">
<h2>Page Fault Calculator</h2>
</div>

<div class="algorithm-info">
<h4>How it works</h4>
<p>Enter a reference string (sequence of page numbers) and select the number of frames.
Then choose an algorithm to see how page faults are handled.</p>
</div>
```

```
<div class="form-group">
<label for="referenceString">Reference String (space-separated):</label>
<input type="text" id="referenceString" placeholder="e.g. 7 0 1 2 0 3 0 4 2 3 0 3 2"/>
</div>

<div class="form-group">
<label for="numFrames">Number of Frames:</label>
<input type="number" id="numFrames" placeholder="e.g. 3" min="1"/>
</div>

<div class="btn-group">
<button class="btn-primary" onclick="calculate('FIFO')">
<span>FIFO Algorithm</span>
</button>
<button class="btn-primary" onclick="calculate('LRU')">
<span>LRU Algorithm</span>
</button>
<button class="btn-primary" onclick="calculate('Optimal')">
<span>Optimal Algorithm</span>
</button>
</div>

<div class="result-container">
<div class="result-header">
<div class="result-title">Simulation Results</div>
</div>
<pre id="result">Results will appear here after running a simulation...</pre>
</div>
</div>
<div class="container about-section" id="about-section">
```

<h2>About Page Fault Algorithms</h2>

Page Fault Algorithms are used by operating systems to decide which page to replace when a page is not found in the main memory during the execution of a program. Here, we discuss three popular algorithms:

<h4>1. FIFO (First In First Out)</h4>

FIFO is the simplest page replacement algorithm. It replaces the oldest page in memory (the page that has been in memory the longest) when a page fault occurs and there is no space available in memory.

The algorithm maintains a queue of pages, and when a page needs to be replaced, the page at the front of the queue is removed and replaced with the new page.

Algorithm:

<pre>

1. Initialize an empty queue.

2. For each page request:

a. If the page is not in memory (a page fault occurs):

i. If there is space in memory, add the page.

ii. If memory is full, remove the page at the front of the queue, and insert the new page.

</pre>

<h4>2. LRU (Least Recently Used)</h4>

LRU replaces the page that has not been used for the longest period of time. The algorithm tracks the order in which pages are accessed, and when a page fault occurs, it replaces the page that has been accessed the least recently.

This algorithm is often more efficient than FIFO because it tends to keep the most recently used pages in memory.

Algorithm:

<pre>

1. Initialize an empty stack.

2. For each page request:

a. If the page is in memory, move it to the top of the stack (most recently used).

b. If the page is not in memory (a page fault occurs):

i. If there is space in memory, add the page.

ii. If memory is full, remove the page at the bottom of the stack (least recently used).

</pre>

<h4>3. Optimal Page Replacement</h4>

The Optimal algorithm is the most efficient in terms of minimizing page faults, but it requires future knowledge of page accesses. The algorithm replaces the page that will not be used for the longest period of time in the future.

Although it cannot be implemented practically due to the need for future knowledge, it serves as a benchmark for comparing other page replacement algorithms.

Algorithm:

<pre>

1. For each page request:

a. If the page is not in memory (a page fault occurs):

i. If there is space in memory, add the page.

ii. If memory is full, find the page that will be used farthest in the future and replace it.

</pre>

</main>

<footer>

```
<p>© 2025 Page Fault Simulator | An Educational Tool for Operating Systems</p>
</footer>
<script>
function calculate(algorithm) {
const refStr =
document.getElementById('referenceString').value.trim().split(/\s+/).map(Number);
const frames = parseInt(document.getElementById('numFrames').value);
let result = '';

if (isNaN(frames) || frames <= 0 || refStr.some(isNaN)) {
result = ' Error: Please enter a valid reference string and frame number greater than 0.';
document.getElementById('result').textContent = result;
return;
}

if (algorithm === 'FIFO') result = calculateFIFO(refStr, frames);
if (algorithm === 'LRU') result = calculateLRU(refStr, frames);
if (algorithm === 'Optimal') result = calculateOptimal(refStr, frames);

document.getElementById('result').innerHTML = formatResult(result);
}

function formatResult(result) {
return result
.replace(/Page Fault/g, '<span class="highlight-fault">Page Fault</span>')
.replace(/Hit/g, '<span class="highlight-hit">Hit</span>');
}

function calculateFIFO(pages, frames) {
const queue = [];
let faults = 0, output = '';
```

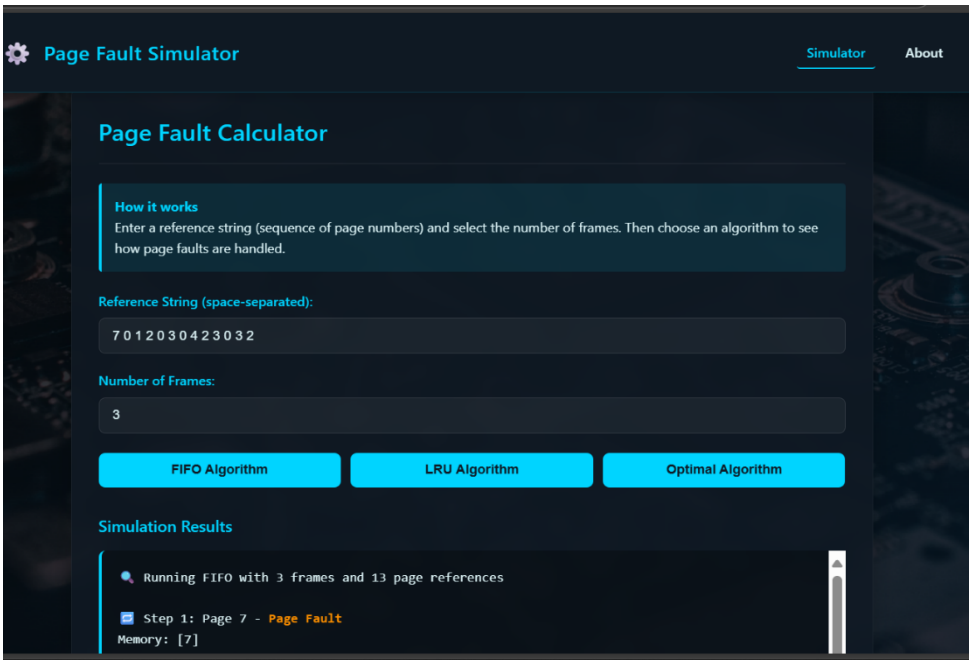
```
output += ` Running FIFO with ${frames} frames and ${pages.length} page references\n\n`;

pages.forEach((page, i) => {
if (!queue.includes(page)) {
if (queue.length === frames) {
const evicted = queue.shift();
output += `Step ${i + 1}: Page ${page} - Page Fault (Replaced ${evicted})\n`;
} else {
output += `Step ${i + 1}: Page ${page} - Page Fault\n`;
}
queue.push(page);
faults++;
} else {
output += ` Step ${i + 1}: Page ${page} - Hit\n`;
}
output += `Memory: [${queue.join(', ')}]\n\n`;
});

output += `Summary: ${faults} faults out of ${pages.length} references
(${Math.round((faults/pages.length)*100)}% fault rate)\n`;

return output;
}

function calculateLRU(pages, frames) {
const stack = [];
let faults = 0, output = '';

output += ` Running LRU with ${frames} frames and ${pages.length} page references\n\n`;
```

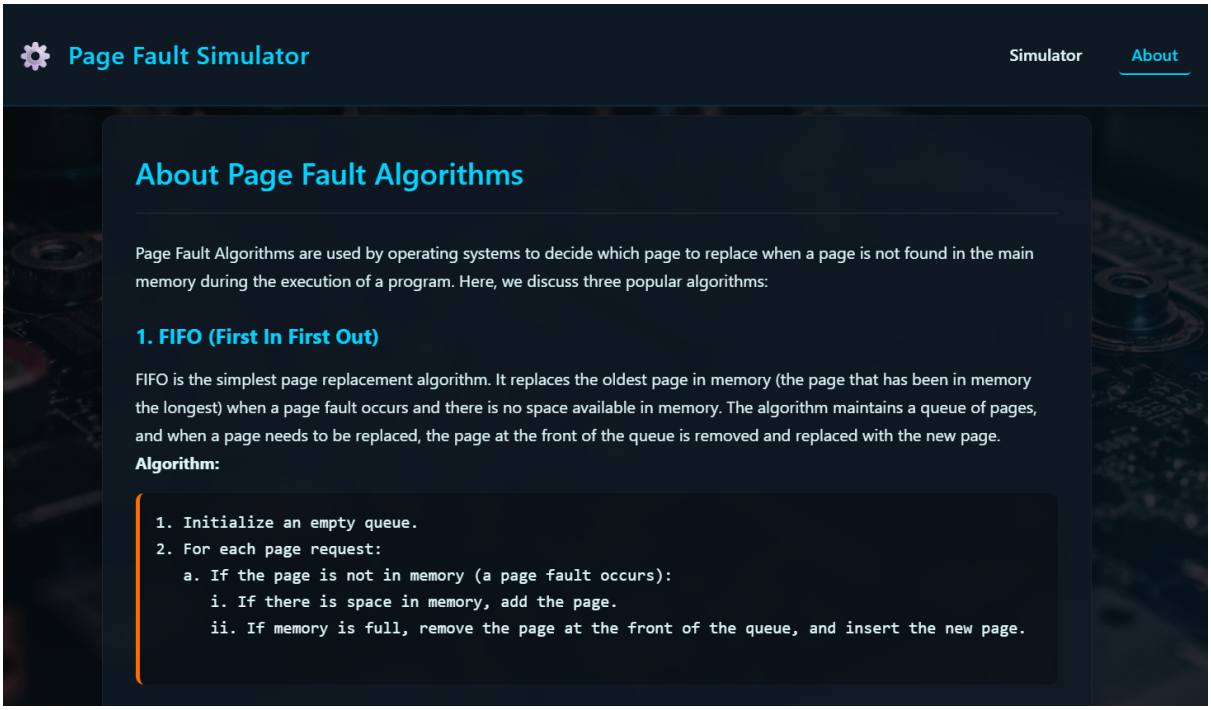
```
pages.forEach((page, i) => {
const idx = stack.indexOf(page);
if (idx === -1) {
if (stack.length === frames) {
const evicted = stack.shift();
output += `Step ${i + 1}: Page ${page} - Page Fault (Replaced ${evicted})\n`;
} else {
output += `Step ${i + 1}: Page ${page} - Page Fault\n`;
}
stack.push(page);
faults++;
} else {
stack.splice(idx, 1);
stack.push(page);
output += `Step ${i + 1}: Page ${page} - Hit\n`;
}
output += `Memory: [${stack.join(', ')}]\n\n`;
});

output += `Summary: ${faults} faults out of ${pages.length} references
(${Math.round((faults/pages.length)*100)}% fault rate)\n`;

return output;
}

function calculateOptimal(pages, frames) {
const memory = [];
let faults = 0, output = '';

output += `Running Optimal with ${frames} frames and ${pages.length} page
references\n\n`;
```

```
pages.forEach((page, i) => {
if (!memory.includes(page)) {
if (memory.length < frames) {
memory.push(page);
output += `Step ${i + 1}: Page ${page} - Page Fault\n`;
} else {
let future = memory.map(p => {
const idx = pages.indexOf(p, i + 1);
return idx === -1 ? Infinity : idx;
});
const replaceIdx = future.indexOf(Math.max(...future));
const evicted = memory[replaceIdx];
memory[replaceIdx] = page;
output += `Step ${i + 1}: Page ${page} - Page Fault (Replaced ${evicted})\n`;
}
faults++;
} else {
output += `Step ${i + 1}: Page ${page} - Hit\n`;
}
output += `Memory: [${memory.join(', ')}]\n\n`;
});
output += ` Summary: ${faults} faults out of ${pages.length} references
(${Math.round((faults/pages.length)*100)}% fault rate)\n`;

return output;
}

function showHome() {
document.getElementById('home-section').style.display = 'block';
document.getElementById('about-section').style.display = 'none';
```

```
document.getElementById('home-nav').classList.add('active');
document.getElementById('about-nav').classList.remove('active');
}
function showAbout() {
document.getElementById('about-section').style.display = 'block';
document.getElementById('home-section').style.display = 'none';
document.getElementById('about-nav').classList.add('active');
document.getElementById('home-nav').classList.remove('active');
}
showHome();
document.getElementById('referenceString').value = "7 0 1 2 0 3 0 4 2 3 0 3 2";
document.getElementById('numFrames').value = "3";
</script>
</body>
</html>
```

OUTPUT SCREEN





GITHUB LINK

https://github.com/CHANDRUK3/OS-pagereplacement

WEBSITE LINK

https://page-replacement-stm.netlify.app/

PRESENTATION PHOTOS