Geetanjali College of Engineering and Technology

UGC AUTONOMOUS INSTITUTION

### **Permanently Affiliated to JNTUH,Approved by AICTE ,Accredited by NAAC**

### **Cheeryal (V) ,Keesara (M),Medchal Dist.Telangana State-501 301**



### **Department of Computer Science**

### **Operating system Mini Project**

#### **Implementation of Memory allocation algorithms**

**Faculty Incharge:**

**E.Mahender**

**Assistant Professor**

Team:

**K.Madhu**

***index***

|  |  |  |  |
| --- | --- | --- | --- |
| *Topics* |  |  |  |
| **MEMORY MANAGEMENT:**  ***1.0 Best-fit***   * 1. ***explanation***   2. ***algorithm***   3. ***advantages***   4. ***disadvantages*** |  |  |  |
| 1. ***First-fit***    1. ***explanation***    2. ***algorithm***    3. ***advantages***    4. ***disadvantages*** |  |  |  |
| 1. ***Worst-fit***    1. ***explanation***    2. ***algorithm***    3. ***advantages***    4. ***disadvantages***   ***4.0 program using php***  ***(combine of 3 fits)***  ***4.1 mains.php***  ***4.2 f1.php***  ***4.3 f2.php*** |  |  |  |
|  |  |  |  |

# **ABSTRACT**

The technique to control and coordinate the computer memory, to assign blocks to different running programs for the optimization of an entire system performance, is called as memory management. It resides in hardware of the Operating System (OS), and in applications and programs. Memory management is one of the most important parts of an operating system. Next to the CPU, it is one of the most important resources in a computer system. Memory management keeps track of each and every memory location, regardless of either it is allocated to some process or it is free. Dynamic Memory Allocation is an efficient technique of Memory Management and becomes essential part of today’s computer system. It solves problem of sharing memory among different processes. Static memory allocation process is done at compile time; we have to allocate all the memory which is requiring to program, applications or variable in its lifecycle at beginning of execution. This project includes implementation of memory allocation algorithms using HTML, CSS, JavaScript and PHP.

MEMORY MANAGEMENT

Memory Management is one of the services provided by OS which is needed for Optimized memory usage of the available memory in a Computer System.

For this purpose OS uses 3 methods:

1. Best Fit
2. First Fit
3. Worst Fit

***Best-fit:-***

**1.0 *Explanation:-***

Best fit uses the best memory block based on the Process memory request. In best fit implementation the algorithm first selects the smallest block which can adequately fulfill the memory request by the respective process.

Because of this memory is utilized optimally but as it compares the blocks with the requested memory size it increases the time requirement and hence slower than other methods. It suffers from Internal Fragmentation which simply means that the memory block size is greater than the memory requested by the process, then the free space gets wasted.

Once we encounter a process that requests a memory which is higher than block size we stop the algorithm.

**1.1 *Best Fit Algorithm:-***

1. Get no. of Processes and no. of blocks.
2. After that get the size of each block and process requests.
3. Then select the best memory block that can be allocated using the above definition.
4. Display the processes with the blocks that are allocated to a respective process.
5. Value of Fragmentation is optional to display to keep track of wasted memory.
6. Stop.

### 1.2 *advantage:-*

Memory utilization is much better than first fit as it searches the smallest free partition first available.

### 1.3 *Disadvantage:-*

It is slower and may even tend to fill up memory with tiny useless holes.

***First-fit:-***

**2.0 *Explanation:-***

In this scheme we check the blocks in a sequential manner which means we pick the first process then compare it’s size with first block size if it is less than size of block it is allocated otherwise we move to second block and so on.

When first process is allocated we move on to the next process until all processes are allocated.

**2.1 *algorithm:-***

1. Get no. of Processes and no. of blocks.
2. After that get the size of each block and process requests.
3. Now allocate processes  
   if(block size >= process size)  
   //allocate the process  
   else  
   //move on to next block
4. Display the processes with the blocks that are allocated to a respective process.
5. Stop.

### 2.2 *Advantage:-*

Fastest algorithm because it searches as little as possible.

### 2.3 *Disadvantage:-*

The remaining unused memory areas left after allocation become waste if it is too smaller. Thus request for larger memory requirement cannot be accomplished.

***Worst-fit:-***

**3.0 *Explanation:-***

Worst Fit allocates a process to the partition which is largest sufficient among the freely available partitions available in the main memory. If a large process comes at a later stage, then memory will not have space to accommodate it.

**3.1*Algorithm:-***

1. Get no. of Processes and no. of blocks.
2. After that get the size of each block and process requests.
3. Now allocate processes  
   if(block size <= process size)  
   //allocate the process  
   else  
   //move on to next block
4. Display the processes with the blocks that are allocated to a respective process.
5. Stop.

### 3.2 *Advantage:-*

Reduces the rate of production of small gaps.

### 3.3 *Disadvantage:-*

If a process requiring larger memory arrives at a later stage then it cannot be accommodated as the largest hole is already split and occupied.

***4.o Program using php:***

***4.1 Mains.ph***

*<html>*

*<head>*

*<title> </title>*

*<frameset cols="50%,\*">*

*<frame src="f1.php" name="f1">*

*<frame name="f2">*

*</frameset>*

*</head>*

*</html>*

*</html>*

***4.2 F1.php***

*<!DOCTYPE html>*

*<html>*

*<head>*

*<meta charset="ISO-8859-1">*

*<title>Memory allocation Algorithms</title>*

*<style>*

*body {*

*background-color: #E6B0AA;*

*}*

*table, th, td {*

*border: 0px solid black;*

*border-collapse: collapse;*

*text-align:center;*

*}*

*</style>*

*<script type="text/javascript">*

*function array1(){*

*//alert("called");*

*var i = 1;*

*var n1=document.getElementById("n1").value;*

*var n2=document.getElementById("n2").value;*

*my\_div.innerHTML ="";*

*my\_div1.innerHTML ="";*

*if(n1&&n2)*

*{*

*for(i=1;i<=n1;i++)*

*{*

*my\_div1.innerHTML +="<br>P :"+i+" <input type='text' name='l"+ i +"' required> <br>"*

*}*

*for(i=1;i<=n2;i++)*

*{*

*my\_div.innerHTML +="<br>B: "+i+" <input type='text' name='b"+ i +"' required> <br>"*

*}*

*}*

*}*

*</script>*

*</head>*

*<body>*

*<br>*

*<form action="f2.php" method="POST" name="form1" onsubmit="document.form1.target='f2'; return true;">*

*<h2 style="color:red;" align="center" font-family="Arial" > Implementation of memory Allocation Algorithms </h2>*

*&emsp;&emsp;&emsp;<label style="color:blue;"><b> Select the fit:</b></label>*

*<select name="fit">*

*<option>-select fit-</option>*

*<option value="1" selected>First Fit</option>*

*<option value="2" >best Fit</option>*

*<option value="3" >worst Fit</option>*

*</select>*

*&emsp;&emsp;&emsp;&emsp;&emsp;&emsp;*

*<label style="color:blue;"><b>No of process</b></label>*

*<input type="text" Id="n1" name="n1" value="" placeholder="enter the n" oninput="array1()" required><br> <br>*

*&emsp;&emsp;&emsp;&emsp;&emsp;&emsp;&emsp;&emsp;&emsp;&emsp;<label style="color:blue;"><b>No of blocks </b></label>*

*<input type="text" Id="n2" name="n2" value="" placeholder="enter the n" oninput="array1()" required><br> <br>*

*<table style="width:100%">*

*<tr>*

*<th style="color:#7E5109 ">Blocks:</th><th style="color:#6E2C00 ">Process:</th>*

*</tr>*

*<tr>*

*<td>*

*<div id="my\_div">*

*</div>*

*</td>*

*<td>*

*<div id="my\_div1">*

*</div>*

*</td>*

*</tr>*

*</table>*

*</body>*

*<br><br><br>*

*<center>*

*<input type="submit" name="" value="simulate" > &emsp;&emsp;*

*<input type="reset" >*

*</center>*

*</form>*

*</body>*

*</html>*

***4.3 F2.php***

*<!DOCTYPE html>*

*<html>*

*<head>*

*<meta charset="ISO-8859-1">*

*<title>Memory allocation Algorithms</title>*

*<link rel="stylesheet" href="https://maxcdn.bootstrapcdn.com/bootstrap/3.3.7/css/bootstrap.min.css" integrity="sha384-BVYiiSIFeK1dGmJRAkycuHAHRg32OmUcww7on3RYdg4Va+PmSTsz/K68vbdEjh4u" crossorigin="anonymous">*

*<style>*

*#cc{*

*color:MAROON;*

*font:50px verdana;*

*font-weight:"bold";*

*text-align:center;*

*}*

*table, th, td {*

*border: 1px solid black;*

*border-collapse: collapse;*

*text-align:center;*

*}*

*th, td {*

*padding: 5px;*

*}*

*body {*

*background-color: #F7DC6F ;*

*}*

*</style>*

*</head>*

*<body>*

*<?php*

*$p= array();*

*$y= array(1=>"FIRST FIT",2=>"BEST FIT",3=>"WORST FIT");*

*$np=$\_POST["n1"];*

*$fit1=$\_POST["fit"];*

*$nb=$\_POST["n2"];*

*$val=100;*

*for($i=1;$i<=$np;$i++)*

*{*

*$ind="l"."$i";*

*$p[$i]=$\_POST[$ind];*

*}*

*$barray= array();*

*$b= array();*

*$fragment=array();*

*$parray=array();*

*$unal=array();*

*for($i=1;$i<=$nb;$i++)*

*{*

*$ind="b"."$i";*

*$b[$i]=$\_POST[$ind];*

*$val+=100;*

*$barray[$i]=-1;*

*$fragment[$i]=0;*

*$parray[$i]=0;*

*$unal[$i]=0;*

*}*

*$lowest=9999;*

*$temp;*

*$flag=0;*

*$unid=0;*

*for($i=1;$i<=$np;$i++)*

*{*

*for($j=1;$j<=$nb;$j++)*

*{*

*if($barray[$j]!=1)*

*{*

*$temp=$b[$j]-$p[$i];*

*if($temp>=0)*

*if($lowest>$temp)*

*{*

*$parray[$i]=$j;*

*$lowest=$temp;*

*$flag=1;*

*}*

*}*

*}*

*if($flag==1){*

*$fragment[$i]=$lowest;*

*$temp=$parray[$i];*

*$barray[$temp]=1;*

*$lowest=10000;*

*$flag=0;*

*}*

*else{*

*$parray[$i]=0;*

*$unid++;*

*$unal[$unid]=$i;*

*}*

*}*

*if($fit1=='2')*

*{*

*echo("<br><center><input type='text' id='cc' value='Best Fit' text-align:center disabled='disabled' size='10'> </center><br>");*

*echo "<table class='table table-dark' align='center' text-align:center; border='1' cellpadding='5' cellspacing='0'>*

*<thead>*

*<tr bgcolor='#979797'>*

*<th scope='col'>Process no</th>*

*<th scope='col'>Process size</th>*

*<th scope='col'>Block no</th>*

*<th scope='col'>Block size</th>*

*<th scope='col'>Fragment</th>*

*</tr>*

*</thead> ";*

*for($i=1;$i<=$np;$i++)*

*{ echo "<tr>";*

*if( $parray[$i]!=0)*

*{$temp=$parray[$i];*

*echo("<br> <td> $i </td> <td> $p[$i] </td> <td> $parray[$i] </td> <td> $b[$temp] </td> <td> $fragment[$i] </td>");*

*}*

*echo "</tr>";*

*}*

*for($i=1;$i<=$unid;$i++)*

*{ echo "<tr>";*

*$temp6=$unal[$i];*

*echo "<span style='color: red;' /> <td> $unal[$i] </td> <td>$p[$temp6]</td><td style='color: red;'>Not allocated </td> <td>-</td> <td>-</td></span>";*

*echo "</tr>";*

*}*

*echo "</table>";*

*//echo "<br><br><br><br><br>";*

*}*

*if($fit1=='1')*

*{*

*echo("<br><center><input type='text' id='cc' value='First Fit' text-align:center disabled='disabled' size='10'> </center><br>");*

*$bsize=array();*

*$psize= array();*

*$flags=array();*

*$allocation=array();*

*for($i = 1; $i <=$nb; $i++)*

*{*

*$flags[$i] =0;*

*$allocation[$i] =-1;*

*}*

*$bno=$nb;*

*$bsize=$b;*

*$pno=$np;*

*$psize=$p;*

*for($i = 1; $i <= $pno; $i++)*

*for($j = 1; $j <= $bno; $j++)*

*if($flags[$j] == 0 && $bsize[$j] >= $psize[$i])*

*{*

*$allocation[$j] = $i;*

*$flags[$j] = 1;*

*break;*

*}*

*echo " <table class='table table-dark' align='left' border='1' cellpadding='3' cellspacing='0'>*

*<thead>*

*<tr bgcolor='#979797'>*

*<th scope='col'>Block no</th>*

*<th scope='col'>Block size</th>*

*<th scope='col'>Process no</th>*

*<th scope='col'>Process size</th>*

*<th scope='col'>Fragment</th>*

*</tr>*

*</thead> ";*

*for($i = 1; $i <= $bno; $i++)*

*{ echo"<tr>";*

*echo("<td> $i </td> <td> $bsize[$i]</td>");*

*if($flags[$i] == 1)*

*{*

*$temp=$allocation[$i];*

*$frag1=$bsize[$i]-$psize[$temp];*

*echo("<td> $allocation[$i] </td> <td> $psize[$temp] </td> <td> $frag1 </td><br>"); }*

*else{*

*echo "<span style='color: red;'' /><td style='color: red;'>Not allocated </td> <td> -</td> <td>- </td> <br></span>";*

*}*

*echo("</tr>");*

*}*

*echo"</table>";*

*echo "<br><br><br><br><br>";*

*}*

*if($fit1=='3'){*

*echo("<br><center><input type='text' id='cc' value='Worst Fit' text-align:center disabled='disabled' size='10'> </center><br><br><br>");*

*$fragments=array();*

*$blocks=array();*

*$files=array();*

*$flags=array();*

*$top = 0;*

*$block\_arr=array();*

*$file\_arr=array();*

*$po=array();*

*$number\_of\_blocks=$nb;*

*$number\_of\_files=$np;*

*for($m = 1; $m <= $number\_of\_blocks; $m++)*

*{*

*$block\_arr[$m]=0;*

*$file\_arr[$m] = 0;*

*$flags=0;*

*$po[$m]=$m;*

*}*

*for($m = 1; $m <= $number\_of\_files; $m++)*

*{*

*$file\_arr[$m] = 0;*

*}*

*$blocks=$b;*

*for($i=1; $i<=$number\_of\_blocks; $i++)*

*{*

*for($j=$i+1; $j<=$number\_of\_blocks; $j++)*

*{*

*if($blocks[$i] < $blocks[$j])*

*{*

*$tmp = $blocks[$i];*

*$blocks[$i] = $blocks[$j];*

*$blocks[$j] = $tmp;*

*$temp1=$po[$i];*

*$po[$i]=$po[$j];*

*$po[$j]=$temp1;*

*}*

*}*

*}*

*$files=$p;*

*for($m = 1; $m <= $number\_of\_files; $m++)*

*{*

*$flag3=0;*

*for($n = 1; $n <= $number\_of\_blocks; $n++)*

*{*

*if($block\_arr[$n] != 1)*

*{*

*$temp = $blocks[$n] - $files[$m];*

*if($temp >= 0&& $flag3==0)*

*{*

*if($top < $temp)*

*{*

*$file\_arr[$m] =$po[$n];*

*$top = $temp;*

*$block\_arr[$n] = 1;*

*$flag3=1;*

*}*

*}*

*}*

*$fragments[$n] = $top;*

*$temp=$file\_arr[$m];*

*$top = 0;*

*}*

*}*

*echo "<table class='table table-dark' align='left' border='1' cellpadding='3' cellspacing='0'>*

*<thead>*

*<tr bgcolor='#979797'>*

*<th scope='col'>Process no</th>*

*<th scope='col'>Process size</th>*

*<th scope='col'>Block no</th>*

*<th scope='col'>Block size</th>*

*<th scope='col'>Fragment</th>*

*</tr>*

*</thead> ";*

*for($m = 1; $m <= $number\_of\_files; $m++)*

*{*

*$temp=$file\_arr[$m];*

*echo "<tr>";*

*echo ("<td>$m </td> <td> $files[$m] </td> " );*

*if($temp==0){*

*echo"<span style='color: red;' /> <td style='color: red;'>Not allocated</td><td>-</td><td>-</td</span>";*

*$temp=-1;*

*}*

*else {*

*$frag=$blocks[$temp]-$files[$m];*

*echo (" <td> $file\_arr[$m] </td> <td> $blocks[$temp] </td> <td> $frag</td>");*

*}*

*echo"</tr>";*

*}*

*echo"</table>";*

*echo "<br>";*

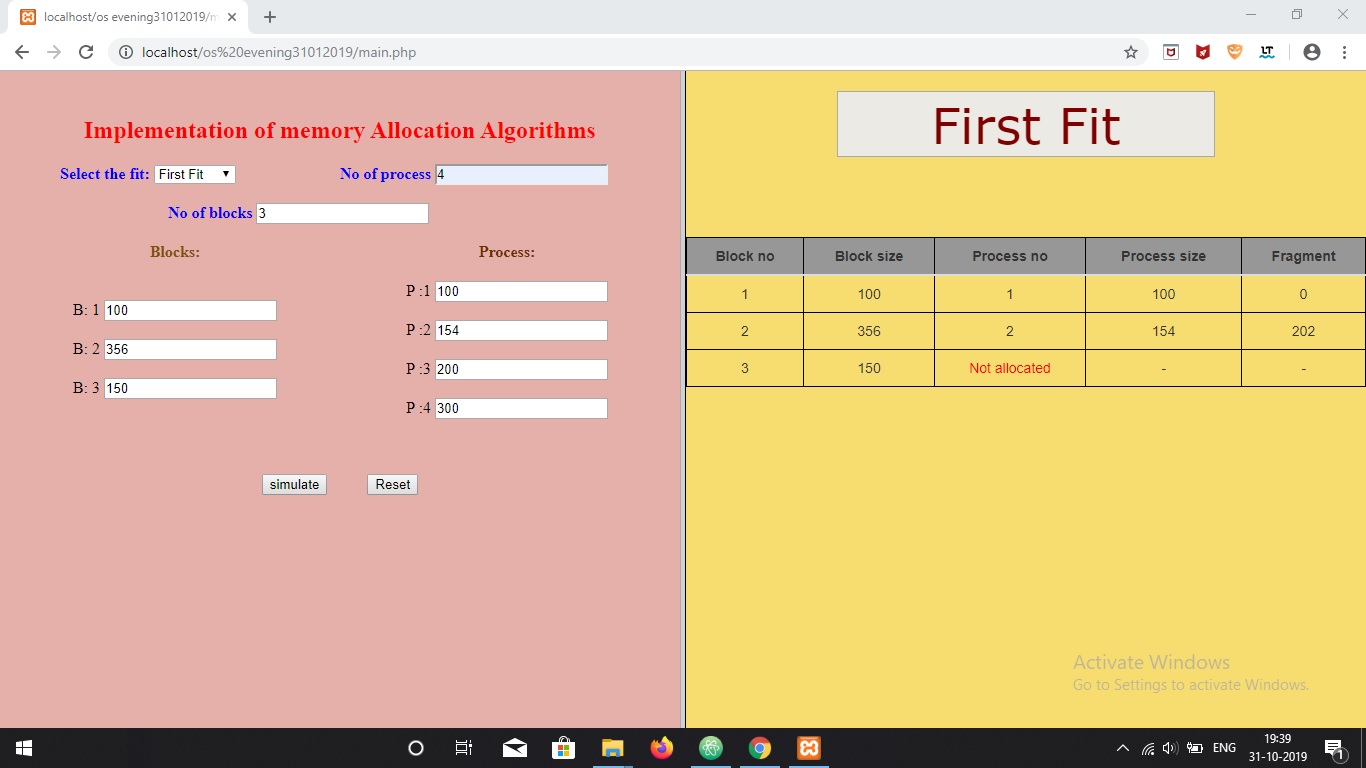
*}*

*?>*

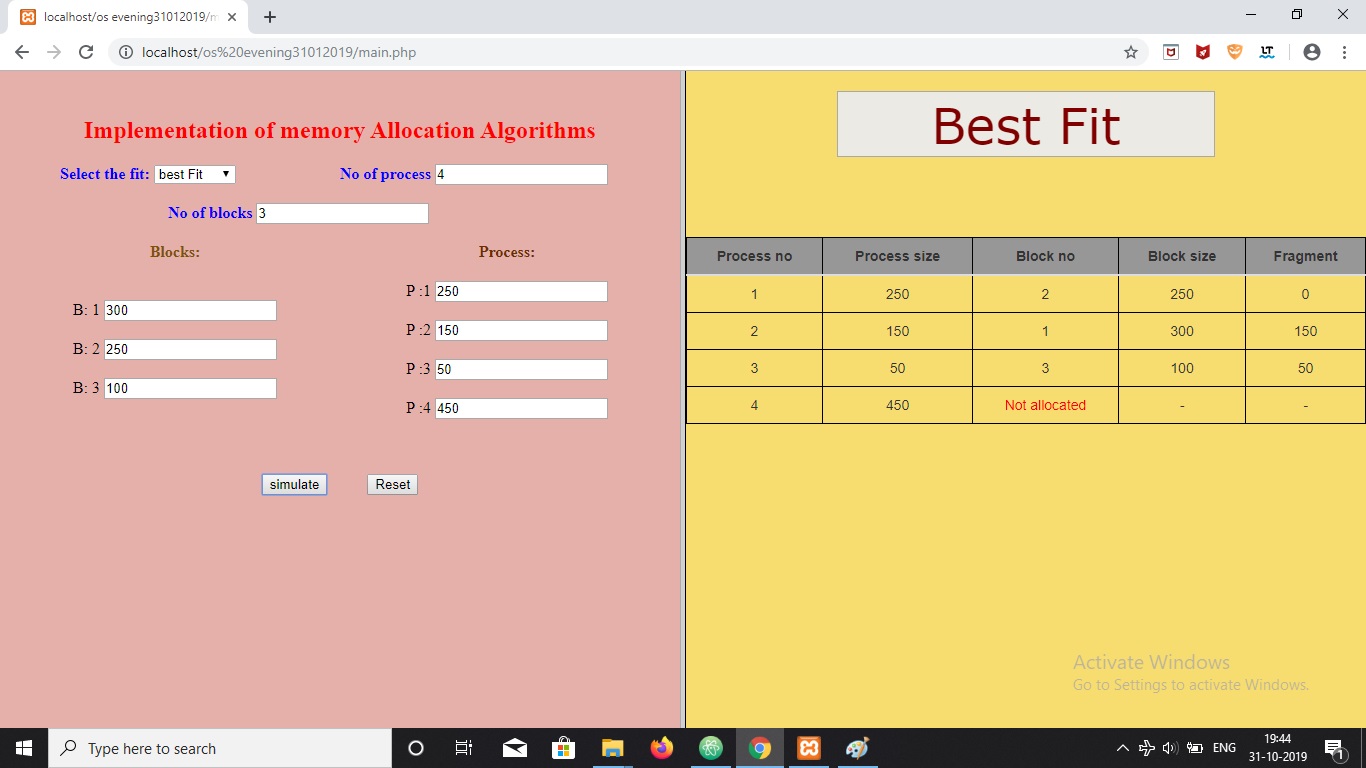
***Output:-***

The input and the output are below:-

1)



2)



3)

