

**WIA2004 OPERATING SYSTEM**

**LAB 5 PROJECT REPORT**

**FIRST-FIT**

**GROUP MEMBERS**

| **No.** | **Member Name** | **Matric Number** |
| --- | --- | --- |
| **1.** | **MUHAMMAD IZZAT HAKEEM BIN ROSDI** | **17204856** |
| **2.** | **WAN MUHAMMAD AZIB BIN WAN ABDUL RAHIM** | **17207678** |
| **3.** | **MUHAMMAD AMIRUL AMIN BIN ARMAN** | **17202969** |
| **4.** | **JONE WAN GAN** | **S2031153** |
| **5.** | **LAWRENCE LEROY CHIENG TZE YAO** | **S2018935** |

**OBJECTIVE**

\*Write a C program to simulate the following contiguous memory allocation techniques

a) First Fit

**DESCRIPTION**

For both fixed and dynamic memory allocation schemes, the operating system must keep a list of each memory location noting which are free and which are busy. Then as new jobs come into the system, the free partitions must be allocated.

**FIRST-FIT MEMORY ALLOCATION:**

This method keeps the free/busy list of jobs organized by memory location, low-ordered to high-ordered memory. In this method, the first job claims the first available memory with space more than or equal to its size. The operating system doesn’t search for an appropriate partition but just allocates the job to the nearest memory partition available with sufficient size.

**ADVANTAGES OF FIRST-FIT MEMORY ALLOCATION:**

It is fast in processing. As the processor allocates the nearest available memory partition to the job, it is very fast in execution.

**DISADVANTAGES OF FIRST-FIT MEMORY ALLOCATION:**

It wastes a lot of memory. The processor ignores if the size of the partition allocated to the job is very large as compared to the size of the job or not. It just allocates the memory. As a result, a lot of memory is wasted and many jobs may not get space in the memory, and would have to wait for another job to complete.

**CODE**

#*FIRST-FIT*

#include<stdio.h>

void main()

{

int bsize[10], psize[10], bno, pno, flags[10], allocation[10], i, j;

for(i = 0; i < 10; i++)

{

flags[i] = 0;

allocation[i] = -1;

}

printf("Enter no. of blocks: ");

scanf("%d", &bno);

printf("\nEnter size of each block: ");

for(i = 0; i < bno; i++)

scanf("%d", &bsize[i]);

printf("\nEnter no. of processes: ");

scanf("%d", &pno);

printf("\nEnter size of each process: ");

for(i = 0; i < pno; i++)

scanf("%d", &psize[i]);

for(i = 0; i < pno; i++) //allocation as per first fit

for(j = 0; j < bno; j++)

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

{

allocation[j] = i;

flags[j] = 1;

break;

}

//display allocation details

printf("\nBlock no.\tsize\t\tprocess no.\t\tsize");

for(i = 0; i < bno; i++)

{

printf("\n%d\t\t%d\t\t", i+1, bsize[i]);

if(flags[i] == 1)

printf("%d\t\t\t%d",allocation[i]+1,psize[allocation[i]]);

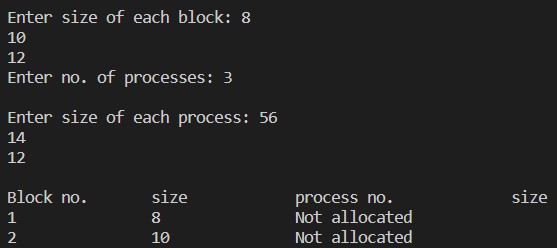
else

printf("Not allocated");

}

}

**OUTPUT**

****