Simulation of Memory Allocator Module

Posting date: 6/March/14 **Total: 20 Marks**
Submission date: 19/March/14 by 10pm
Demo: 20-24th March/14

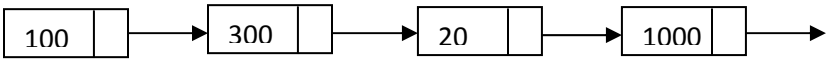
Problem statement:

Write a C/C++ program to allocate memory to the processes from the list of available memory chunks for each of the following three strategies:.

- First-fit (FF),
- Best-fit (BF),
- Worst-fit (WF: This is, in essence, exactly opposite to best fit. Here you look for the available memory chunk for a process-request, which maximises(wherein, you minimise this in case of best fit) the amount of memory left after allocation of memory to a process).

Important Instructions

- A list of available memory chunks which may NOT be of the same size.



- A list of processes with different memory requirements. Note that process with 0 or negative request are not allowed. Take care of such situations.
  (Ex: 40, 200, 300, 20, 50, ...)
- You have to use Singly Linked List for your implementation.
- Use Input and output modules (a C/CPP file provided by instructor)
- Allocate memory to the process **from a single memory chunk only** and after allocation from a given chunk, update its size.
- Remove a memory chunk from the list of available memory chunks when the chunk gets completely allocated.

You need to show an analysis of the methods with the following parameters for the entire allocation procedure of the list of processes:

- Percentage of processes for which memory could be allocated (number of processes got allocated / Total number of processes * 100).

- Number of memory chunks visited by the memory-allocation procedure for allocating memory.
- The amount of memory that is not allocated (Portion of memory chunks which could not be allocated to any process).

Input format:
You need to take inputs from console, using standard way, that has already been conferred to you previously.

- First two inputs are 'n' and 'm'.
  n = number of memory chunks available,
  m= number of processes.
- Next n inputs are n integers – each representing the size of n memory-chunks.
- Further m inputs are m integers – each denoting the amount of memory requested by the processes.

Note:
Please note that n, m and all the next m+n inputs can be very large, may be in order of 10^6. So your code should be able to run against those large inputs. These values have to be read with **cin or scanf** from a file using **input redirection**.

Example:
Input Format:
8 10
100 260 35 100 320 567 111 200
57 124 156 300 230 122 20 34 234 89

Indicative Output format:
Your output should be the following numbers, one at each line:

40.55	(First fit: percentage of processes that could be allocated memory)
100556	(First fit: number of times the memory chunks visited)
20089	(First fit: amount of memory could not be allocated)
50.92	(Best fit: percentage of processes that could be allocated memory)
0320769	(Best fit: number of times the memory chunks visited)
17897	(Best fit: amount of memory could not be allocated)
48.67	(Worst fit: percentage of processes that could be allocated memory)
100556	(Worst fit: number of times the memory chunks visited)
19876	(Worst fit: amount of memory could not be allocated)

Note: All output for which you are asked to give percentage, you need to output only 2 decimal places. There are 9 numbers in total.

Submission:

- Makefile
- Documentation in text format named as **entry_no.txt.**
- Main program with file name as **entry_no.cpp or entry_no.c** containing three functions FF, BF and WF. The order of functions be maintained. Use only one language C or C++.
- Put all files in the folder named as **entry_no**, e.g. 2011ME20778 and **zip** it into **entry_no.zip.**
- Mail the assignment to the following email id: csl201iitd@gmail.com

Marking Scheme:

First Fit :	4
Best Fit:	4
Worst Fit:	4
Analysis:	6
Makefile:	1
Documentation:	1
Total:	**20**