**Memory Challenge**

A memory consumption pattern of 1000 applications is available in allocation.txt.

 Applications are A1 to A1000

 A1: 10 MB malloc, 20 malloc, 9 MB malloc, 30 MB malloc, 10 malloc, 9 MB malloc, 10 MB malloc, 20 malloc, 9 MB malloc, 10 MB malloc, 20 malloc, 9 MB malloc, 20 malloc,

A2:  9 MB malloc, 30 MB malloc, 10 MB malloc, 29 MB malloc, 101 MB malloc, 10 MB malloc, 20 MB malloc,9 MB malloc, 10 MB malloc, 20 MB malloc, 9 MB malloc,  9 MB malloc

          …

A1000: 2 MB malloc, 8 MB malloc, 23 MB malloc, 23 malloc, 10 MB malloc, 23 MB malloc,  8 MB malloc, 23 MB malloc, 23 malloc, 10 MB malloc,  8 MB malloc, 23 MB malloc

**Implement a most efficient memory manager consisting of my\_malloc() and my\_free() routines where MAX memory size is 500MB for applications to allocate.**

Input file to simulate memory requirement sequence: allocation\_seq.txt

Format of file is listed below:

A1:10MB;my\_malloc()

A2:200MB;my\_free()

A1: 10 MB; my\_malloc()    denotes  Application A1 is malloced 10 MB using my\_malloc().

A2 :200 MB; my\_free()        denotes Application A2 freed 200 MB using my\_free() – last allocated 200 MB pointer.

**Write a program to accept input file for your memory manager and display the results for the following:-**

(a) What is the size of the largest block of unfragmented memory available once the input file is run?

(b) What is the time it took to run the complete input file?

(c) How many times the program ran out of memory?

(d) Allow the user to start defragmentation of memory.

(e) Allow the user to issue my\_malloc() and my\_free() for a specific application. E.g.

A1:200MB;my\_malloc()

A1:200MB;my\_free()

(f) Allow the user to query the largest block of **unfragmented** memory.