Programming Assignment # 2
Due: 03/27/2023

Basic Base/bounds memory management with garbage collection (in C++)

This exercise is to give you some experience with old fashioned memory management.

Problem specification: You are to mimic process management within physical memory. To do this, you will create an event driven **simulation** in which a process is **"created"**, memory references for that process are made and **processes die**. You are to use **base and limit** registers to make sure that a process stays within bounds and whenever a **memory reference** is made, you give the **EXACT physical address** of the location after following the base register. You are to handle external memory fragmentation, by **"defragmenting"** the physical memory whenever a process wants to start up and there is sufficient memory to support the request.

A process can ask to **start at some time T**, it can then make memory reference thereafter, and then it can **stop at some other future T**. There will also be a directive **DUMP**, which tells you to state the base and bounds of all ACTIVE processes.
You will output the time step whenever you garbage collect. You can **ONLY garbage collect** if you do not have contiguous memory to support the new process, but you have enough for the process if you collect the pieces.
You must connect chunks of unused memory without performing garbage collection.

If there is no memory available for a new process (even with defragmentation and garbage collection), the request should be rejected.

The input will start up with a simple configuration line:

RAM 8000

Which means that the physical memory has 8000 bytes of RAM
Then the following events are examples of the following input

```
1 START  P1  2000       -- at time 1,start P1 with size 2000
2 P1 LOC 17             -- at time 2, give the physical location of P1 VA 17
3 START P2   2000
4 P2 LOC 17
5 START P3   2000
6 P3 LOC 17
7 DUMP                  -- give the base/bounds of all active processes
8 START P4 2000
9 P4 LOC 17
10 STOP P2
11 STOP P3
```

```
12 START P5 4000        -- should NOT cause garbage collection
13 P1 LOC 17
14 P4 LOC 17
15 P5 LOC 17
17 STOP                 -- STOP the SIMULATION
```

A different scenario is

```
1 START  P1  2000       -- at time 1,start P1 with size 2000
2 P1 LOC 17             -- at time 2, give the physical location of P1 VA 17
3 START P2   2000
4 P2 LOC 17
5 START P3   2000
6 P3 LOC 17
7 DUMP              -- give the base/bounds of all active processes
8 START P4 2000
9 P4 LOC 17
10 STOP P1
11 STOP P3
12 START P5 4000        -- WILL cause garbage collection
13 P1 LOC 17
14 P4 LOC 17
15 P5 LOC 17
17 STOP                 -- STOP the SIMULATION
```

As part of your submission, please create a ReadMe.txt file that includes the following:

- Instructions on how to compile and run your program.
- An explanation of your garbage collection and defragmentation strategy, including details on the structure(s) used to represent memory. Did you use arrays, linked lists, or other structures?
- Examples of program output.

In addition, please make sure to include comments in your code.

References: https: // www.cs.nmsu.edu/ ~scooper/C S474/ program5 / index.html