PROJECT 10

Assignment description

Objective The objective of this assignment is to implement the memory management of the Jack operating system.

Contract Complete the implementation of *Memory.jack* and compile it to obtain *Memory.vm*

Resources You will need three tools: the Jack compiler, to translate your program into a set of *.vm* files, the VM emulator, to run and test your translated program, and the Jack Operating System.

For details read section 12.3.7 of the text book.

Details

Read Chapter 12 of the text book. Make sure to understand sections 12.1.2 and 12.3.7 very well. You can start by implementing a memory allocation mechanism that follows the pseudocode in Figure 12.6a. Then you can extend your implementation and design a proper deallocation mechanism.

Your program will be evaluated using the test program (in folder *tester*), which performs memory allocations and deallocations in a loop. The program is designed to check the following:

- The memory segments corresponding to allocated variables are not corrupted
- The deallocated memory segments can be reused, and memory is defragmented
- `Memory.alloc(K)` returns successfully if and only if K+1 consecutive memory words are free.
    - Example: Assume your memory has size 20 and you perform:

```
i = Memory.alloc(10);
j = Memory.alloc(5);
Memory.deAlloc(j);
k = Memory.alloc(7);
```

    - Are you sure the last allocation will succeed in your implementation?

Hints

1. As the Jack OS uses the Memory class to instantiate objects, it will not instantiate an object of the class Memory. Do not write a constructor in the Memory class; that would never be called. If you need class variables within Memory you need to use static variables.
2. *MemoryTest.tst* is provided to test your implementation of *peek* and *poke*. To test your implementation of alloc and deAlloc please use the test program found in the 'tester' directory.
3. To test your *alloc* and *deAlloc* code for common mistakes and to test the efficiency of your implementation compile *Memory.jack* to obtain *Memory.vm*. Then move your *Memory.vm* into the *tester* folder. Finally, use the VM emulator to load all the *tester/*.vm* files and run the program. (Note: if you do not copy your *Memory.vm* into the *tester* folder, the program will use the *Memory.vm* implementation that comes with the Jack OS. You can use the

supplied test program (*tester/Main.jack*) as a reference to assess how well your program performs.)

Submission

Your project must be completed by the deadline. *All projects should be done individually.* One submission per student, in a zipped file named *projectMM-familyname-firstname.zip* including:

- Memory.jack file, which should include the complete implementation of the functions listed within Memory.jack. If you complete both the wasteful and the recycling version of alloc and deAlloc then please provide two separate files: MemoryWasteful.jack for the wasteful implementation and Memory.jack for the recycling implementation.
- Filled in Excel sheet. (Please do not change the file name or extension. Please answer closed (Yes/No) questions with *Yes* or *No*.)

Copying and other forms of plagiarism and cheating will be reported for disciplinary action.