CS2050 – Lab 12

Summer 2024

Concepts to Practice

- Heaps
- Makefiles

Submission Information

Submit this assignment by the mucsmake command to submit lab12.c:

```
mucsmake 2050 lab12 lab12.c
```

Description

To get started on this lab, type the following while logged in to hellbender.rnet.missouri.edu:

```
cs2050start lab12
cd lab12
make
./a.out
```

For the lab assignment, you are to start with the starter code provided in the lab12 directory. This starter code is an outline of what you need to do. The only file you should change is lab12.c. Notice that you already have tests provided in lab12main.c this time (though, you can write additional tests if you want).

You job is to implement a **min heap** data structure which is accessible through the following functions:

```
PHeap HeapConstruct();
int HeapInsert(PHeap heap, int key, void * inputData);
int HeapDelete(PHeap heap, void ** outputData);
void HeapDestruct(PHeap heap);
```

In addition to these functions, you may want to implement this function to help you understand and debug your code as you work on it:

```
void PrintHeap(PHeap heap);
```

Notice that these functions all rely on a type called “PHeap” which means “Pointer to a Heap”. The provided header file does not define the Heap structure, so you will need to do that in order to implement this lab. Furthermore, you will very likely want to implement other “behind the scenes” data structures (such as a Node structure, for example). You will also almost certainly want to implement “worker” functions in addition to the “public” functions shown above.

For full credit, your heap does not have to utilize “satellite data”. The suggestions from the class notes on implementing a heap do not address satellite data, so you can implement our heap without it [in which case, the inputData parameter and the outputData parameter to HeapInsert() and HeapDelete() respectively can be ignored – just fill in NULL when calling these]. However, you can receive bonus points if you are able to store satellite data correctly in your heap along with the provided key value. If you do use satellite data, note that HeapDelete() will output data into the variable pointed to by the outputData parameter – that is, this is an “output” parameter and points to a variable that will contain the address of the output satellite data after HeapDelete() is successfully called.

Function Specifications:

PHeap HeapConstruct() – Allocates memory and initializes the overall heap data structure. The returned pointer should be used in all other functions.

int HeapInsert(PHeap heap, int key, void * inputData) – Given a valid heap pointer, this function inserts a node into the heap using the provided key. If satellite data is supported, the inputData pointer is stored in the node along with the key. Return -1 if the function fails or 0 if it succeeds.

int HeapDelete(PHeap heap, void ** outputData) – Given a valid heap pointer, this function removes the node with the smallest (min) key value from the heap and returns the key value for the removed node. If satellite data is supported, the output pointer should be filled in with the pointer to the satellite data. Note that when deleting a node, any memory allocated by your implementation should be deallocated. Return -1 if the function fails or the key if it succeeds.

void HeapDestruct(PHeap heap) – Given a valid heap pointer, deallocate all remaining allocated memory that has been allocated by your functions to the heap.

Notice:

1. All of your lab submissions must include documentation in the form of code comments to receive full points. In addition, your program is expected to have **a comment header** at the top that includes your name, pawprint, the course you are taking, and the lab that you solved.
2. All of your lab submissions must compile under GCC using the *–Wall* and *–Werror* flags to be considered for a grade (note that the Makefile should take care of this for you in this case).
3. Do **NOT** change any files other than lab12.c and lab12main.c. Additional struct definitions and helper functions must be placed in your lab12.c file.

Rubric (20 points plus up to 10 bonus points):

- HeapConstruct()
  - Creates a valid and empty heap data structure – 3 points
- HeapInsert()
  - Inserts a node into the heap while maintaining the heap invariant – 5 points
  - Bonus: satellite data is successfully stored in the same node as the given key – 5 points
- HeapDelete()
  - Removes the minimum key-valued node from the heap while maintaining the heap invariant – 5 points
  - Frees allocated memory for the deleted node – 2 points
  - Bonus: satellite data is successfully returned in the output parameter and matches the original data originally inserted – 5 points
- HeapDestruct()
  - Frees all remaining memory allocated to the heap (e.g. any remaining nodes) – 5 points

Sample Output (output from just the starter code)

```
jimr@jimrsurfacepro9:~/CS2050/SS2024/labs/lab12$ ./a.out
-1 deleted
-1 deleted
-1 deleted
-1 deleted
-1 deleted
-1 deleted
-1 deleted
-1 deleted
-1 deleted
-1 deleted
-1 deleted
-1 deleted
-1 deleted
-1 deleted
-1 deleted

Employees should be in reverse order as we delete them:
        NULL pointer!
        NULL pointer!
        NULL pointer!
        NULL pointer!
        NULL pointer!
        NULL pointer!
        NULL pointer!
        NULL pointer!
        NULL pointer!
        NULL pointer!
        NULL pointer!
        NULL pointer!
        NULL pointer!
        NULL pointer!
```

Sample Output (completed code)

```
jimr@jimrsurfacepro9:~/CS2050/SS2024/labs/lab12.solved$ ./a.out
1 deleted
2 deleted
3 deleted
4 deleted
5 deleted
6 deleted
7 deleted
8 deleted
9 deleted
10 deleted
11 deleted
12 deleted
13 deleted
14 deleted
15 deleted
```

Employees should be in reverse order as we delete them:

- William Jefferson Clinton (A)
- William David Moser (X)
- Earl Daniel Kraus (I)
- Alan Jerome Lammers (X)
- Paul David Polly (L)
- Lawrence David Ries (x)
- James EvilJimR Ries (X)
- James AlternateDimension Ries (L)
- Cisco Ries (S)
- Murphy Ries (X)
- Laura Christine Ries (m)
- James Edward Ries (M)
- George W Bush (z)

Sample Output (completed code with Valgrind)

```
jimr@jimrsurfacepro9:~/CS2050/SS2024/labs/lab12.solved$ valgrind ./a.out
==121003== Memcheck, a memory error detector
==121003== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==121003== Using Valgrind-3.18.1 and LibVEX; rerun with -h for copyright info
==121003== Command: ./a.out
==121003==
1 deleted
2 deleted
3 deleted
4 deleted
5 deleted
6 deleted
7 deleted
8 deleted
9 deleted
10 deleted
11 deleted
12 deleted
13 deleted
14 deleted
15 deleted

Employees should be in reverse order as we delete them:
      William Jefferson Clinton (A)
      William David Moser (X)
      Earl Daniel Kraus (l)
      Alan Jerome Lammers (X)
      Paul David Polly (L)
      Lawrence David Ries (x)
      James EvilJimR Ries (X)
      James AlternateDimension Ries (L)
      Cisco  Ries (S)
      Murphy  Ries (X)
      Laura Christine Ries (m)
      James Edward Ries (M)
      George W Bush (z)
==121003==
==121003== HEAP SUMMARY:
==121003==     in use at exit: 0 bytes in 0 blocks
==121003==   total heap usage: 32 allocs, 32 frees, 2,168 bytes allocated
==121003==
==121003== All heap blocks were freed -- no leaks are possible
==121003==
==121003== For lists of detected and suppressed errors, rerun with: -s
==121003== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
```