Lab 5 - IO and Dynamic Memory

Part 1: What’s Your Jedi Name

# Part A

* + What is your jedi name? To find out, follow these steps:
    - For your Name:
      * Take the first 3 letters of your last name;
      * Add the first 2 letters of your first name.
  + Write a function that takes 3 parameters, char \* first name, char \* last name, and a string buffer, then copies your jedi name in the buffer.

# Part B

* + - * Create a file with 10 people's names in it (1 name per line).
      * Ask user to input one more name from terminal, add it to the end of your file.
      * Read in the first and last name from your file and print out the first, last, and the jedi name for each person (use your function from Part A to generate jedi names).
  + Some people’s names may be not long enough. Since each name is on its own line, you can determine how best to handle this.

Part 2: Check for Memory Leaks

# Part A: Allocating your memory

* + Define a function that takes an unsigned integer, size, allocates the requested amount of memory on the heap, then returns a void pointer.
    - The function interface should be: void \* allocate(unsigned int size);
    - Be sure to check for null on the pointer returned by malloc within the function.
  + Define a second function that takes a void \* and the size of memory to be freed, frees the memory parameter, then returns a null pointer.
    - The function signature should be: void \* deallocate(void \*, int size);
    - Make sure to keep track of the amount of memory you are freeing;
    - Be sure to use the return value to null the pointer.
  + Test both your functions to make sure they work.

# Part B: Keep track of your usage

* + Add a global variable, int heap\_usage = 0.
  + Alter your allocate() and deallocate() functions to add to or subtract from (respectively) heap\_usage every time you allocate or free memory.
  + Add a print statement to both functions that states how much memory (number of bytes) was allocated or freed, and how much memory (number of bytes) is currently in use.

# Part C: Check for memory leaks

* + Declare a struct, Names, that has 3 char \* attributes, char \* first\_name, char \* last\_name, and char \* jedi\_name.
    - You can store first and last name in a single char buffer if you wish.
  + You should just modify your existing code from Part 1, which should already print out all people’s names.
    - Alter your code from Part 1 to read in the name, allocate space on the heap, and store it in the Names struct;
    - Alter your Jedi Name function to take the Names struct as a parameter, and store the jedi name in the Names struct;
    - Change your code so it uses the allocate function from part 2 instead of a direct call to malloc.
  + Make sure you free all memory using your deallocate function before exiting the program.
    - Also be sure to set the pointers to NULL using the return value of deallocate.
  + When you run your program, it should print 0 bytes used before exiting.

Part 3 - Submission

Create a tar archive with the command “tar -czvf lab5.tar.gz .”, and then email your archive to [bu580u@gmail.com](mailto:bu580u@gmail.com) and cc your TA [ssopark1@binghamton.edu](mailto:ssopark1@binghamton.edu) before the submission deadline. Make sure you do not include the executable in your archive (make clean before creating the archive). Late assignments will not be accepted under any circumstances. Plan to turn in your assignments early.

Demo your lab before the demo deadline (after the submission deadline) by downloading your submission from class Gmail and extracting your archive with the command “tar -xvf lab5.tar.gz”. Then compile (with your makefile), and run your code, show your source to your TA, and answer any questions your TA may have.

Grading Guidelines

## Part 1:

* + Part A: 2 point
  + Part B: 2 point
* **Part 2:**
  + Part A: 2 points
  + Part B: 1 points
  + Part C: 2 points

## Style Guidelines and Memory Leaks

* + Follows Style Guidelines: 1 point
  + Do Not Know How to Use Valgrind or Valgrind Shows Memory Leak: -2 points

Submission Deadline: 11:59pm 11/15/2016 EST

Demo Deadline: 11:10am 11/23/2016 EST