Tyler Andersen

**Lab03 – Guess whos fault report**

* I injected three errors into this code. Find the three errors and fix them.​
  + First two errors are commented out. Third error was corrected by moving the delete mysteryAnimal to inside the while loop.
* Add validation code, to check if the user enters something other than 0, 1, 2, 3 or 4.​
* Answer the questions:

**#0** – What are the three errors?​

* + 1) Line 38; printing out AnimalUtil is being initialized to the pointer mysteryAnimal.The error is that the program is attempting to dereference an uninitialized pointer. The pointer mysteryAnimal is declared but not assigned a value, so dereferencing it with \*mysteryAnimal causes undefined behavior.
  + 2) Line 41-43; mysteryAnimal is set to nullptr, trying to dereference it with \*mysteryAnimal causes an error because it is trying to access memory at address().
  + 3) Line 46; In the while loop, the mysteryAnimal is allocated with new on each iteration, but the previous allocation is never deleted. This causes a memory leak. We should add delete mysteryAnimal; at the end of each loop iteration.

**#1** – When mysteryAnimal is declared, the next instruction prints out: “The animal is initialized to: …” It is never initialized to a variable. Why does this not crash?? What is it being initialized to??​

A computer screen with white text

AI-generated content may be incorrect.It doesn’t crash because the line just prints out what the mysteryAnimal is generated. It contains whatever undefined value was previously in that memory location on the stack. Its essentially pointing to some random memory address. The random address might contain readable memory so it doesn’t crash every time so it is unpredictable. After multiple times compiling the program, the program prints “the animal is initialized to: Cat”.

**#2** – Why is your program crashing on this line: “std::cout << "The animal should initally be nothing:…”?​

A screen shot of a computer

AI-generated content may be incorrect.The program crashses because you’re attempting to deference a mullptr. When mysteryAnimal = nullptr, the pointer explicitly points to null. Dereferencing it with \*mysteryAnimal attempts to read memory at address 0, which causes a segmentation fault. Any further user input is treated as an invalid command and crashes.

**#3** – Run the program for 3 different guesses. On each guess, print out:​

- The address of the mysteryAnimal pointer​

- The address where the pointer is pointing​

- The value located at the address where the pointer is pointing.​

A computer screen with white text

AI-generated content may be incorrect.Three printout locations added to each iteration.

A close-up of a paper

AI-generated content may be incorrect.- Draw a picture diagraming each of these memory locations with their values.​

The stack pointer address stays constant (0xc0e41ff8d8), each new allocation creates a new heap address. Each address for each guess is a separate heap address. staticWelcomeMessageis a global variable, so it’s stored in the data segment, which is separate from the stack and heap.

- Is the stack growing from the bottom (0x0000….) or the top (0xFFFFF…) of memory?​

The stack typically grows from top (0xFFFFF…) toward the bottom (0x0000….). Looking at the separate guess values, it is growing towards the bottom.

A computer screen with white text

AI-generated content may be incorrect.**#4 -** Relative to the stack and heap, where is the staticWelcomeMessage located? What is its address? Add it to the diagram.​

staticWelcomeMessageis a global variable, so it’s stored in the data segment, which is separate from the stack and heap. Address for the staticWelcomeMessage for this compilation of the program is 0x7ff60ebd8040.

**Link to github directory:** [**AmbientWarfare117/CS3022\_labs: Labs for CS3022**](https://github.com/AmbientWarfare117/CS3022_labs/tree/main)