Homework 1 Solutions

Question 1

(a) Here are the values of `m`, `n`, and `*p` after each of lines 3 - 8 of the code:

```
5, 7, 7
5, 19, 19
5, 43, 43
5, 43, 5
53, 43, 53
149, 43, 149
```

The output would be `m = 149, n = 43, *p = 149`.

(b) The value of `ptr` would be 3173315908. Since `ptr` is an integer pointer, incrementing it increases the value by the number of bytes in a single integer, which is typically four (32 bits).

(c) This line impliclity calls the Graph constructor. Since the Graph class did not define a default constructor, no appropriate constructor will be found, and the line will not compile.

(d) `ptr` is intially set to the address of the first element in `arr`. Adding two causes it to point to the *third* element of `arr`. Thus, the code will print the value “3”.

Question 2

(a) `leo.Eats()`

(b) `lionPtr->Sleep()`

(c) There are several reasons:

- It helps to differentiate the scope for the variable
- Distinguishes private variables from public
- Avoid name conflicts with functions like getters/setters.

(d) The `Eats()` method of the Animal class will be called, so the output will be “Eats food.” If the `Eats()` function were polymorphic (declared `virtual`), then the `Eats()` method of the Lion class would be called.

Question 3

(a) The begin and end iterators must have access to `m_data`, a private variables of `WideArray`. If they were part of `waIterator`, they would not be able to access this variable.

(b) `void operator++(int dummy) { m_ptr += 2 ; }`

(c) `waIterator waEnd() { return waIterator( m_data + 2*m_size ) ; }`

Question 4

(a) This code would probably result in a *segmentation fault* or other memory error. The address `0xfeedbeef` is almost certainly not one that the user's program is permitted to access, and may not be a valid address at all. Attempting to write to it on the second line will result in a fault.

(b) This code fails to delete the array that `data` points to initially before reassigning `data` to point to the new, larger array. Since the address of the original array will be lost, it will not be possible to delete it, resulting in a memory leak.