Task 3.2P Answer Sheet

Name: Quang Vinh Le
Student ID: 104097488

1. In 2.2P, how many Counter objects were created?

   -Three objects: myCouner[0], myCouner[1], myCouner[2]

2. Variables declared without the “new” keyword are different to the objects created when we call “new”. Referring to the main method in task 2.2P, what is the relationship between the variables initialised with and without the “new” keyword?

- Variables without "new" are references to objects. "new" creates new instances. In the code, myCounters[2] shares object with myCounters[0].

3. In 2.2P, explain why resetting the counter in myCounters[2] also changed the value of the counter in myCounters[0].

   - Both variables refer to the same object; changes to shared object affect all references.

4. The key difference between memory on the heap and memory on the stack is that the heap holds “dynamically allocated memory”. What does this mean? In your answer, focus on the size and lifetime of the allocations.

   - Heap holds dynamically allocated memory. Objects have varying sizes and longer lifetimes. Stack is for method call context, fixed size, short-lived.

5. Are objects allocated on the heap or the stack? What about local variables?

-Objects are allocated on the heap. Local variables reside on the stack, but references to objects are on the stack.

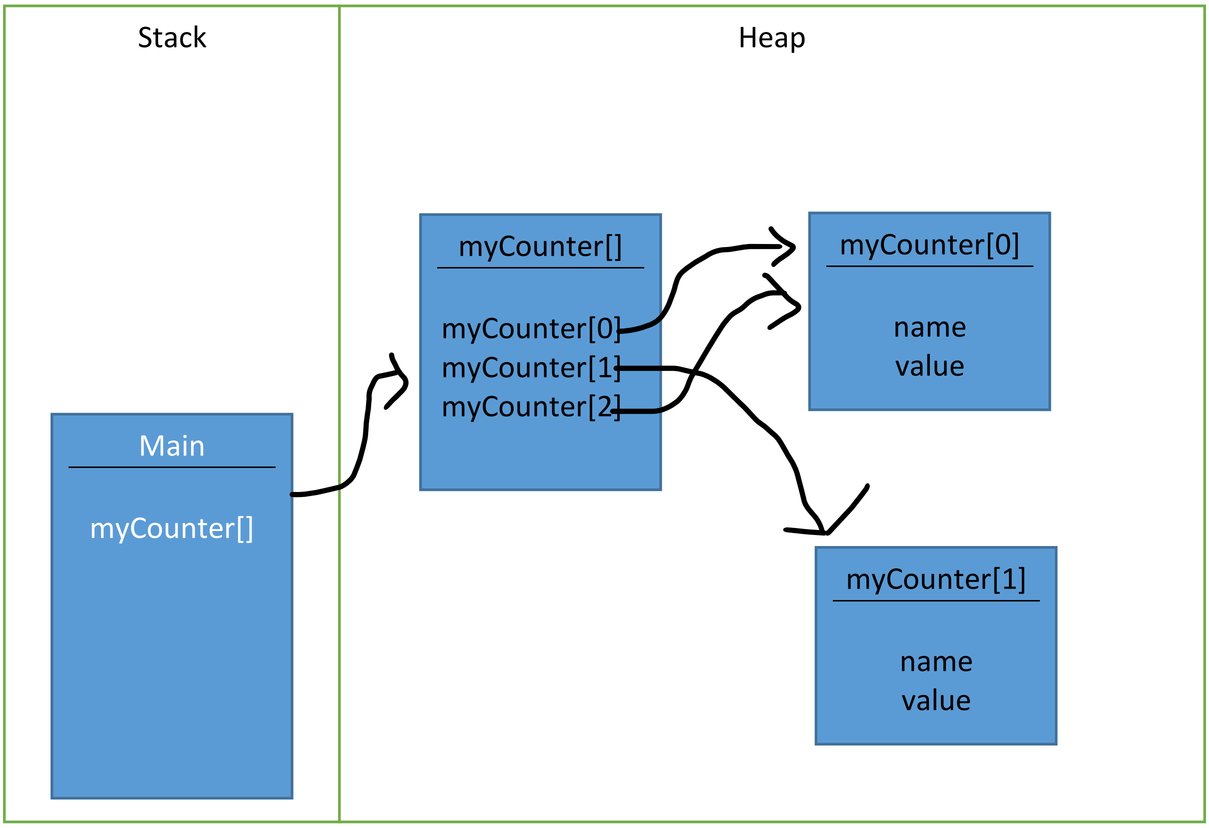
6. What does the new() method do when called for a particular class, and what does it return?

   - "new" keyword creates an instance of a class, allocates memory on heap, initializes fields, returns reference to the new object.

7. Assuming the class Counter exists in my project, if I wrote the code “Counter myCounter;” (note there is no “=”), what value would myCounter have? Why?

-If you write Counter “myCounter”; without an assignment, the variable “myCounter” will have a default value of null. This is because it's a reference variable, and without an explicit assignment to an object, it doesn't point to anything.

8. Based on the code you wrote in task 2.2P, draw a diagram showing the locations of the variables and objects in main and their relationships to one another.


Stack
Heap
Main
myCounter[]
myCounter[]
myCounter[0]
myCounter[1]
myCounter[2]
myCounter[0]
name
value
myCounter[1]
name
value