3.2P: Answer Sheet

Recall task *2.2P Counter Class* and answer the following questions.

1. How many *Counter* objects were created?

2 objects were created

2. Variables declared without the ***new*** keyword are different to the objects created using ***new***. In the ***Main*** function, what is the relationship between the variables initialized with and without the ***new*** keyword?

the value WITHOUT the new value will be store within the stack and those references WITH the new will be stored in the heap.

the value without the new function are just variable to hold direct values meanwhile, the function with the new function is to create instance of that class

3. In the ***Main*** function, explain why the statement ***myCounters***[2].***Reset***(); also changes the value of ***myCounter***[0].

Because counter[2] is refering to counter[0] which has the same value and memory location

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

focussing on the size and lifetime of the allocations
- Size, the stack is smaller compared to the heap because its usually used to store small and temporary objects meanwhile the heap is used to store larger objects
-lifetime memory, the stack is smaller because the its limited to its block of code (functions) while the heap can depends on the application run time

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

references type like instances,classes,array,etc are so allocated on the stack but the actual object is stored in the heap meanwhile the key value like bool,string,int,float all this is store directly at the stack

6. What is the meaning of the expression ***new*** *ClassName*(), where *ClassName* refers a class in your application? What is the value of this expression?

the new ClassName() is used to create a new objects from the ClassName. New ClassName is a references to a newly created object in the heap while the references assigns to its coresponding types which is ussually stored in the stack

7. Consider the statement "*Counter* ***myCounter***;". What is the value of ***myCounter*** after this statement? Why?

myCounter is counted as null value, just a reference variable with no objects called into it yet

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