3.2P: Answer Sheet

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

1. How many *Counter* objects were created?

There were two new Counter objects created(2 new Counter objects and 1 reference to the Counter at the first index)

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?

Variables initialized with "new" keyword in the Main function will contain a reference to the object while variable initialized without "new" keyword will conatin 'null' or uninitialized.

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

Because myCounters[2] and myCounters[0] have the same reference due to the command line myCounters[2] = myCounters[0] (the pointer of myCounters[2] points at the location of myCounters[0])

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.

The heap memory is used for objects where it's size and lifetime are unknown until you run the program while the stack is use for fixed-size data type such as local variables.

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

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

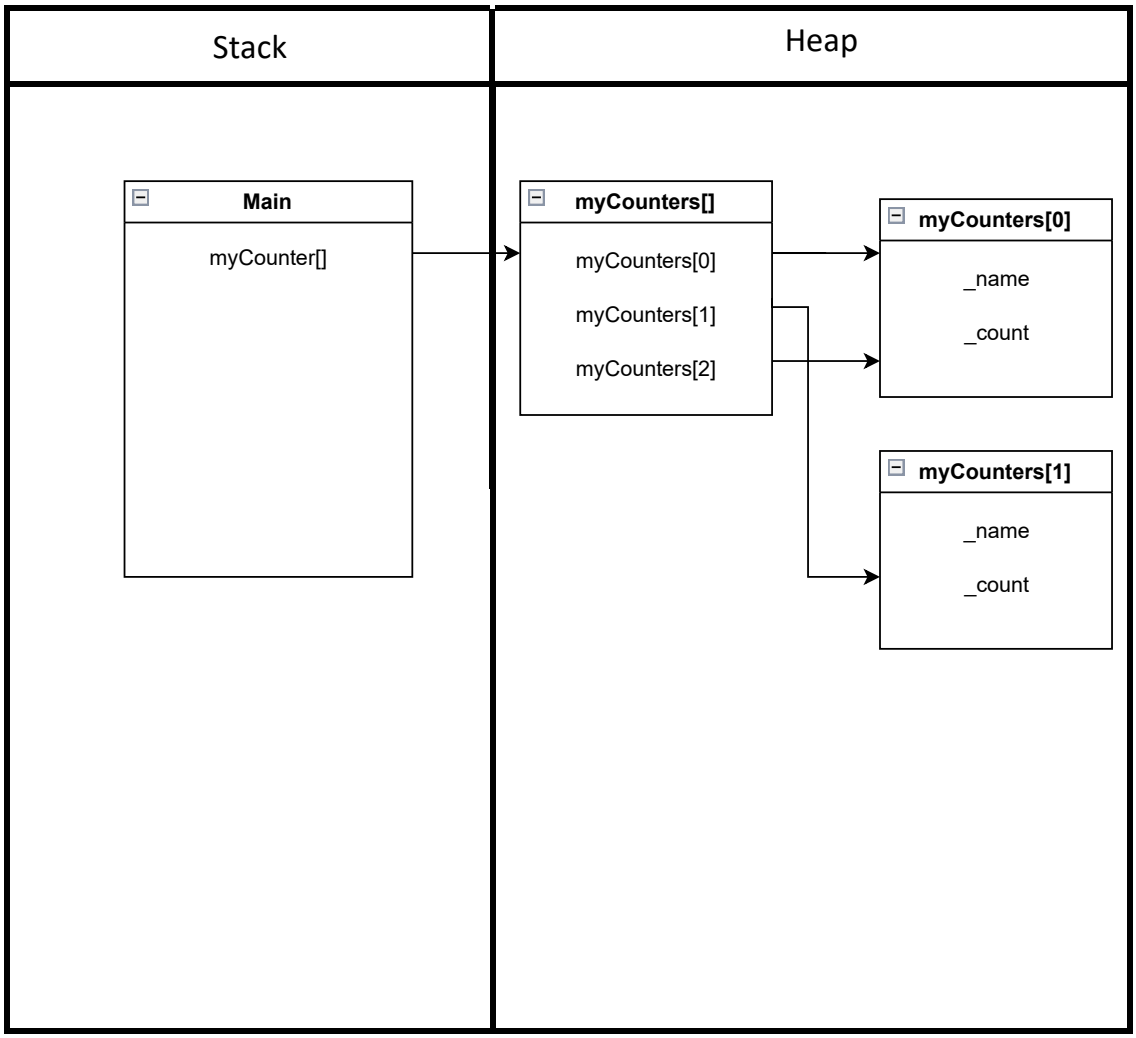
6. What is the meaning of the expression ***new*** *ClassName*(), where *ClassName* refers a class in your application? What is the value of this expression?

When you declare an expression "new ClassName()", the program will allocate the memory needed for that class in the heap memory, then it will call the constructor class and return the reference of the object

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

The value of myCounter after this statement will be null, because myCounter doesn't allocated to any object in the memory.

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.



9. **If the variable myCounters is assigned to null, then you want to change the value of myCounters[X], where X is the last digit of your student ID, what will happen? Please provide your observation with screenshots and explaination.**

Hint. You may want to read this material for this task
https://learn.microsoft.com/en-us/dotnet/csharp/language-reference/keywords/null

For further reading at your own.

- Null pointer CrowdStrike Bug, https://www.thestack.technology/crowstrike-null-pointer-blamed-rca/
- CrowdStrike Blog, https://www.crowdstrike.com/blog/tech-analysis-channel-file-may-contain-null-bytes/

-Because the null keyword does not refer to any object so when you want to access and change an element in myCounter variable then there will appear a bug when you run the program.