Лабораторное занятие №5 от 04.10.2022

Доскоч Роман ФПМИ 4 курс 13 группа

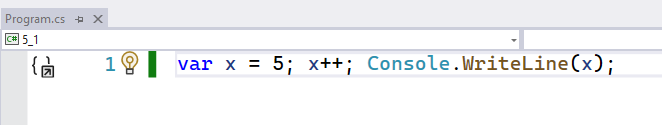
Условие задач ДО КОНЦА НЕДЕЛИ 09.10.2022

ЗАДАНИЕ. 5.1. Получить самую маленькую программу (где под программой понимается совокупность команд или инструкций, которые после дешифровки самой ЭВМ заставляют эту ЭВМ выполнить некоторую строго определенную последовательность действий, приводящую к изменению состояния ЭВМ (а не просто осуществление выхода из программы). 32 балла.

                    ЗАДАНИЕ 5.2. Получить текст на каком угодно языке записи алгоритмов, который после компиляции, загрузки исполнимого модуля в оперативную память превращается  в программу, которая уже в свою очередь заставляет ЭВМ вывести на экран ИСХОДНЫЙ ТЕКСТ БЕЗ ОБРАЩЕНИЯ к этому исходному тексту на внешнем устройстве!!!! Это олимпиадная задача из 80-х годов прошлого столетия и она формулировалась тогда так: «Составить программу, которая печатает сама себя». Но так формулировать нельзя, так как Вы же знаете, что такое программа и что её «составить» нельзя — можно только «получить» в памяти с помощью средств операционной системы. 16 баллов.

                    ЗАДАНИЕ 5.3. Получить программу, которая ИЗМЕНЯЕТ САМА СЕБЯ. 32 балла.

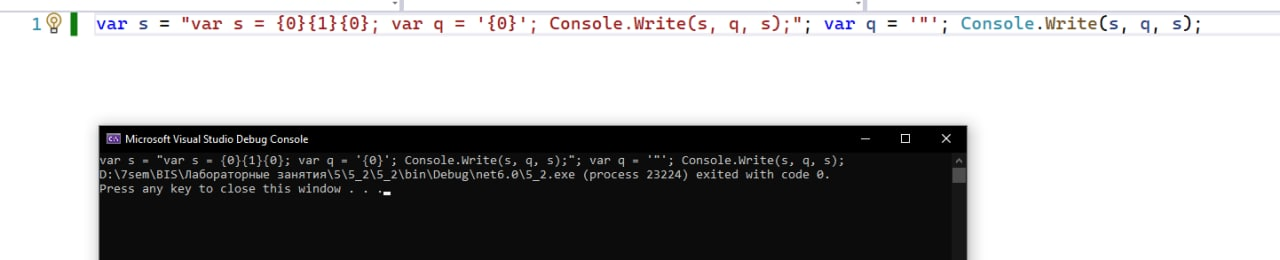
**Задание 5.1**



Завел переменную изменил ее значение и вывел на консоль тем самым изменив состояние эвм

**Задание 5.2**

Такие программы называются квайны (quine)



Завел переменную s в которой описал шаблон строки которую хочу вывести.

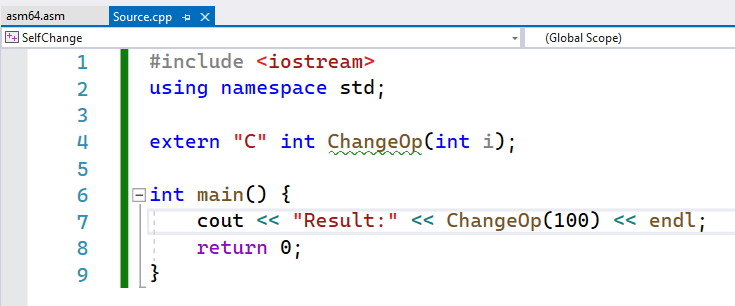
Функция Console.Write() первым аргументом принимает шаблон строки а последующими все что угодно что может быть подставлено в этот шаблон.

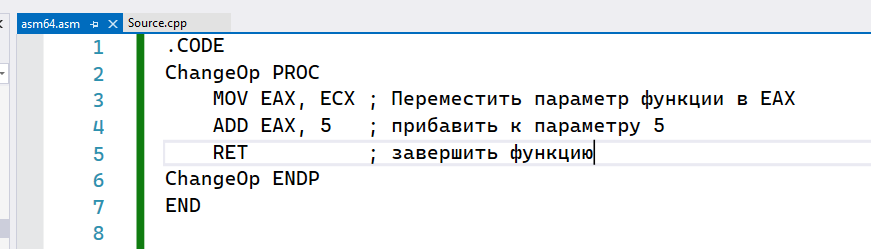
В {0} будет подставляться та или иная строка, так как я вызываю функцию **Console.WriteLine** таким образом что использую сам шаблон для подстановки что и решает мою задачу

**Задание 5.3**

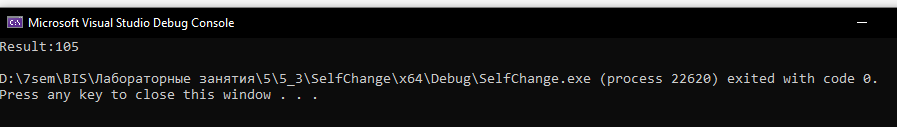
Сначала я написал программу (где под программой понимается совокупность команд или инструкций, которые после дешифровки самой ЭВМ заставляют эту ЭВМ выполнить некоторую строго определенную последовательность действий, приводящую к изменению состояния ЭВМ)

Которая должна повышать данное ей число на 5





Результат

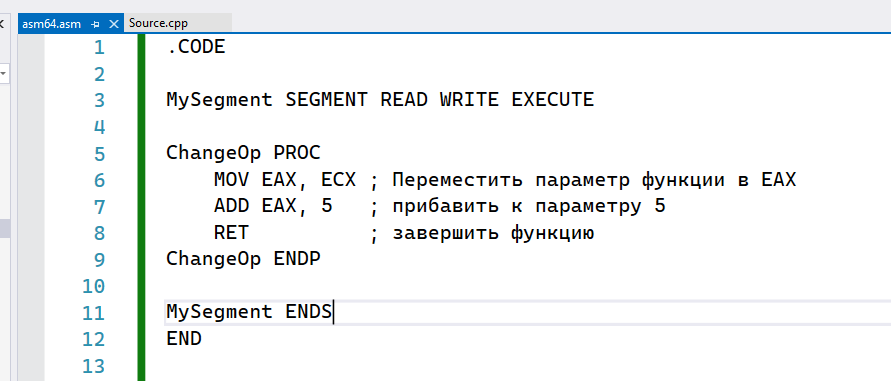


Первый шаг к написанию самоизменяющейся программы — получение возможности изменять код во время выполнения. Когда-то давно программисты поняли, что это плохая идея и с тех пор добавили защиту на изменение выполняемого кода.

Для начала разберемся, где хранятся инструкции, когда программа выполняется. Перед выполнением загрузчик помещает всю программу целиком в память. Затем программа запускается в виртуальном адресном пространстве. Адресное пространство разбито на различные области, нас интересует только область инструкций (text segment) Именно туда загружаются команды процессора

По умолчанию область инструкций можно читать и выполнять оттуда команды. Писать в нее нельзя. Для того, чтобы изменять инструкции в этой области памяти, мы должны изменить права доступа.

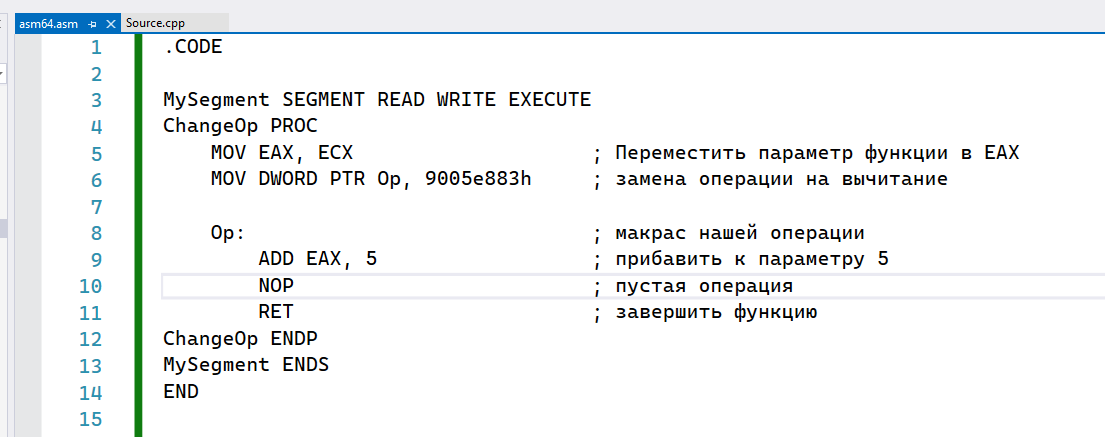
Напишем это следующим образом



Добавим сегмент с правами на чтение запись и выполнение

Потом нам нужно каким-то образом заменить место прибавление на 5 на другую операцию, например на отнятие от 5

Для этого просто обозначим наш код маркером и переназначим ему другую операцию

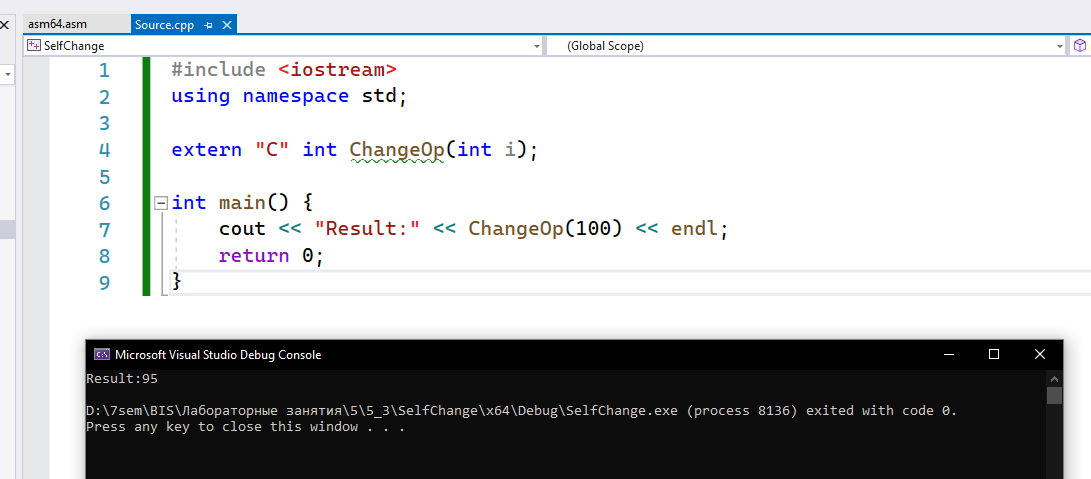


Тут на 6 строчке мы переписываем команду ADD на SUB через команду 9005e883h

Но вы можете спросить, зачем я написал nop ведь это операция которая ничего не делает, да но так как команда sub eax,5 является 3 битной и если мы начнем ее перезаписывать dword’ом то в итоге перезапишем команду ret

Поэтому я добавил тут NOP

Результат:



Теперь выводиться 95!