# Лабораторная 2 – Управление виртуальной памятью.

## Задание 1:

Напишите программу, которая, используя функцию GlobalMemoryStatus, выводит на экран информацию (dwMemoryLoad, dwTotalPhys, dwAvailPhys) о текущем состоянии памяти. Информация должна выводиться с периодичностью в одну секунду до тех пор, пока пользователь не нажмет клавишу с буквой «q». Продемонстрируйте, как меняются параметры состояния памяти при запуске нескольких различных приложений.

## Задание 2:

Напишите программу, демонстрирующую некоторые возможности работы с виртуальной памятью. Последовательность действий в программе:

1. Запросите у пользователя количество страниц виртуальной памяти, которое необходимо зарезервировать. C помощью VirtualAlloc зарезервируйте (MEM\_RESERVE) соответствующий объем памяти с атрибутом PAGE\_READWRITE. (Предварительно необходимо определить размер страницы в данной системе с помощью функции GetSystemInfo). Выведите на экран начальный адрес зарезервированной области.
2. Запросите у пользователя количество страниц виртуальной памяти, которым необходимо передать (commit) физическую память, и номер начальной страницы, с которой будет передана память. C помощью VirtualAlloc выделите (MEM\_COMMIT) запрошенный объем памяти, начиная с указанной страницы, с атрибутом PAGE\_READWRITE. Выведите на экран адрес выделенной области. Пример: предположим, что в первом пункте вы попросили зарезервировать 10 страниц памяти. Здесь вы можете «привязать» физическую память к 2-м страницам из зарезервированной области, начиная, например, с 3-й страницы. Тогда, страницы с номерами 1, 2, 5, 6, 7, 8, 9, 10 по-прежнему останутся в зарезервированном состоянии, а страницами с номерами 3, 4 вы сможете пользоваться.
3. В ту область памяти, которую вы выделили на втором шаге, скопируйте (memcpy\_s) заранее заданный числовой массив небольшого размера (менее 10 элементов). Выведите на экран скопированный массив (не исходный, а из вашей области памяти!).
4. С помощью функции VirtualQuery выведите информацию (State, Protect) о выделенной на втором шаге области памяти. Затем такую же информацию об области памяти, расположенной сразу за выделенной, и адрес этой области. Например, если вы выделяли 2 страницы, начиная с третьей, то сначала вы выводите информацию об области памяти, начиная с 3 страницы, а затем, начиная с 5.
5. C помощью VirtualAlloc выделите (MEM\_COMMIT) еще одну страницу сразу за выделенной на втором шаге областью, с атрибутом PAGE\_READONLY. Выведите на экран адрес выделенной страницы и еще раз с помощью VirtualQuery информацию о ней (State, Protect). Очевидно, что ее состояние и атрибуты защиты должны измениться по сравнению с предыдущим пунктом.
6. Запросите у пользователя количество страниц виртуальной памяти, которые необходимо вернуть, и номер начальной страницы, с которой будет отсчитываться количество. С помощью VirtualFree верните (MEM\_DECOMMIT) нужный объем физической памяти, начиная с указанной страницы. Выведите на экран начальный адрес указанной области и с помощью VirtualQuery выведите информацию (State, Protect) о ней.
7. С помощью VirtualFree полностью освободите (MEM\_RELEASE) зарезервированную на первом шаге область.

**Обязательна** обработка возможных ошибок (например, неуспешное выделение памяти, неуспешное освобождение памяти, некорректные значения для номеров страниц и их количества и т.п.). Без обработки ошибок программа засчитана не будет.