ЛАБОРАТОРНАЯ РАБОТА № 3

# ОТЛАДКА И ДИЗАССЕМБЛИРОВАНИЕ ПРОГРАММ

Цель работы: практическое знакомство с методикой отладки и дизассемблирования программ на языке ассемблера с помощью отладчика OllyDbg

# КРАТКИЕ ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

## 1.1 Средства отладки программ в OllyDbg

**Выполнение.** С помощью отладчика Вы можете выполнять программу пошагово (команду за командой), со входом в подпрограммы или выполнять их за один шаг. Вы можете выполнить программу до следующего возврата или до указанного местоположения, или анимировать выполнение (выполнить команды в автоматическом режиме). Когда приложение выполняется, Вы все еще имеете полный контроль над ним и можете просмотреть содержимое памяти, установить контрольные точки (точки останова). В любое время, Вы можете приостановить или перезапустить отлаживаемую программу.

**Пошаговая отладка.** Пошаговая отладка показывает какие команды или процедуры были выполнены до этого, позволяя Вам проверить все разветвления вашего кода. Пошаговая отладка устанавливает контрольную точку на каждой выбранной команде и удаляет ее, когда команда достигнута.

**Прямая трассировка (Run tracе).** Трассировка выполняет программу пошагово и записывает еѐ выполнение в большой круговой буфер. Этот протокол содержит все регистры (кроме SSE), флаги и ошибки потока, сообщения и декодированные параметры известных функций. Вы можете определить условие прекращения отладки – введя интервал адресов, выражение или команду. Вы можете сохранить информации отладки в файл и сравнить два независимых выполнения программы.

**Профилирование**. Профилировщик вычисляет сколько раз некоторая команда встречается в буфере трассировки. С профилировщиком, Вы узнаете, выполнение какой части кода занимает больше всего времени.

**Внесение исправлений.** Встроенный ассемблер автоматически выбирает самый короткий возможный код. Бинарный редактор показывает данные одновременно в ASCII, UNICODE и шестнадцатеричной формах.

1.1.1 Пошаговая отладка

Пошаговая отладка или пошаговое выполнение программы заключается в проходе по отлаживаемой программе с помощью нажатия клавиши F7 (шаг с вхождением в процедуры) или F8 (шаг с обходом процедур). Главное различие между этими методами - то, что, если текущая команда выполняет вызов (CALL) некоторой функции, F7 войдёт в функцию и остановится на ее первой команде, в то время как F8 будет сразу пробовать выполнить функцию. Если Вы обходите некоторую функцию, любой брейкпоинт (точку останова), или случай отладки, находящийся в пределах функции, приостановит выполнение основной программы, но временный брейкпоинт после вызова процедуры останется активным, и Вы дойдете до него рано или поздно.

Вместо нажатия F7 или F8 несколько сотен раз, Вы можете использовать анимацию (Ctrl+F7 или Ctrl+F8). В этом случае, OllyDbg автоматически повторяет F7 или F8 после того, как предыдущий шаг закончен, и все окна модифицированы. Процесс останавливается когда:

* Вы нажмёте Esc или запустите любую другую команду продвижения, или
* OllyDbg встретит установленный ранее брейкпоинт, или - Отлаживаемая программа произведёт исключение.

Используя Клавиши "+ «и "- «, Вы можете отмотать назад историю выполнения.

Обратите внимание, что OllyDbg прорисовывает большинство окон, каждый раз, когда выполнение приостанавливается. Если анимация, окажется, очень медленной, попробуйте закрыть или, по крайней мере, свернуть все неиспользуемые окна.

Другой, более быстрый путь для обратной трассировки (backtrace) выполнения программы - Run trace. В этом случае OllyDbg создает протокол выполнения и сообщает Вам, когда и сколько раз данная команда была выполнена.

1.1.2. Прямая трассировка

Прямая трассировка - способ обратного выполнения программы, который необходим в некоторых случаях. Вы можете также использовать прямую трассировку для **простого профилирования**. В основном, OllyDbg выполняет отлаживаемую программу пошагово, но не изменяет окна и - что наиболее важно - сохраняет адреса, содержимое регистров, сообщений и известных операндов в буфере трассировки.

Запустите прямую трассировку, нажав Ctrl+F11 (трассировка с входом) или Ctrl+F12 (трассировка с обходом). Остановка трассировки - клавиши F12 или Esc

.

Вы можете определить набор условий, которые проверяются на каждом шаге выполнения прямой трассировки (горячая клавиша: Ctrl+T). Прямая трассировка останавливается, когда встречает какое-либо условие.

Остановиться после прохода указанного числа команд (более точно, после добавления в буфер трассировки). Обратите внимание, что счетчик не делает автоматического перезапуска. То есть, если Вы устанавливаете счет команд до 10, трассировка будет приостановлена один раз после того, как выполнены 10 команд, а не на каждой 10-ой команде.

Остановиться, когда следующая команда соответствует одному из указанных образцов.

Прямая трассировка требует много памяти, в среднем от 16 до 35 байт на команду в зависимости от режима, и выполняется очень медленно. Но во многих случаях, например, когда в программе есть переходы к несуществующему адресу, это единственный способ найти решение проблемы.

В большинстве случаев нет необходимости трассировать системный API код. Опция "Always trace over system DLLs" позволяет трассировать без вхождения в функциональные API при трассировке. OllyDbg предполагает, что модуль системный, если он постоянно находится в системной папке. В окне Modules (Модули) Вы можете отметить любую библиотеку как системную или несистемную.

Чтобы сделать выполнение быстрее, Вы можете ограничить процедуру прямой трассировки выбранными командами или частями кода, устанавливая брейкпоинты на запуск трассировки и запуская программу.

Команды трассировки, упомянутые в начале этого раздела автоматически открывают буфер трассировки. Вы можете определить его размер (до 64 МБ) в Опциях. Этот буфер круговой и, когда он заполняется, переписываются самые старые записи.

Вы можете открыть или очистить буфер прямой трассировки, выбрав «Debug -> Open or clear run trace» (Отладка -> Открыть или очистить буфер трассировки) из главного меню OllyDbg. После того как буфер трассировки открыт, OllyDbg зарегистрирует все паузы в выполнении, даже те, которые не были вызваны прямой трассировкой. Например, Вы можете идти по программе, нажимая F7 или F8, а затем обратно проходить код, используя клавиши Плюс и Минус. Заметьте, что эти клавиши просматривают историю, когда буфер трассировки закрыт. Если Вы проходите программу с помощью трассировки, Регистры, и Информационные области окна становятся серыми, подчеркивая, что данные регистров, которые они отображают - не фактические.

OllyDbg может посчитать, сколько раз каждая команда появляется в буфере прямой трассировки. В окне Дизассемблер выберите «View -> Profile data» (Просмотр -> Профильные данные). Эта команда заменяет столбец Comments (Комментарий) столбцом Profile (Профиль). Или, если отображена панель, щѐлкните еѐ несколько раз, пока она не отобразит Profile information. Заметьте, что отображаемый счетчик - динамический и не подсчитывает старые команды, удалѐнные из буфера трассировки. Вы можете также рассматривать данные профиля для целого модуля, отсортированных несколькими нажатиями, в

отдельном окне Profile

Специальная команда в окне Дизассемблера «Runtrace -> Add entries of all procedures» (Прямая трассировка -> Добавить все процедуры), позволяет проверить, насколько часто вызывается каждая распознанная процедура. Другая команда «Runtrace -> Add branches in procedure» (Прямая трассировка -> Добавить ветки в процедуре) принудительно трассирует все распознанные адресаты переходов в пределах процедуры. В этом случае, профиль позволяет находить наиболее часто выполняемые переходы и оптимизировать их для увеличения скорости.

Опция «Search for -> Last record in run trace» (Найти -> Последнюю запись в буфере трассировки) в контекстном меню окна Дизассемблер находит, когда была выполнена, и была ли выполнена вообще, отмеченная команда в последний раз.

Окно прямой трассировки отображает содержимое буфера трассировки. Для каждой команды существует определённое содержимое регистров, которые были изменены командой (более точно, изменились между записью в исходнике и обновленной). Если Вы сделаете двойной клик на некоторой команде, всплывающее окно выберет все ссылки на данную команду в буфере трассировки, и Вы сможете быстро просмотреть их, нажимая клавиши Плюс или Минус. Если опция «Trace -> Synchronize CPUandRuntrace» (Трассировка -> Синхронизировать окно CPU и окно прямой трассировки) установлена, Дизассемблер перейдѐт к окну прямой трассировки.

Заметьте, что, когда Вы удаляете пошаговую трассировку, Вы одновременно удаляете принудительную прямую трассировку.

## 1.2 Использование меню OllyDbg

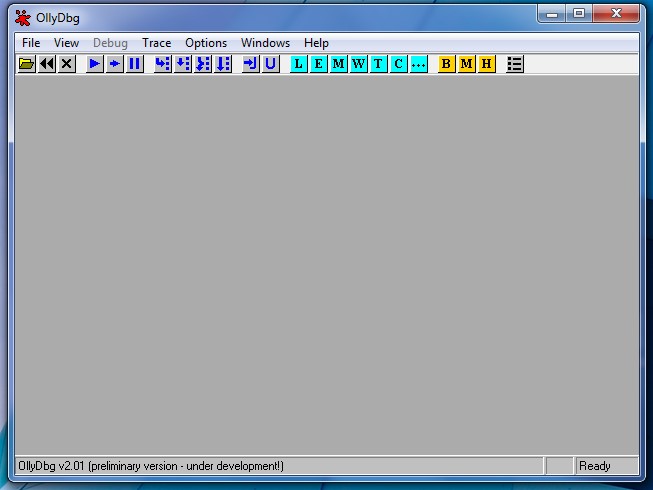


Рис. 1 Главное окно программы OllyDbg

Откроем исследуемый файл (команда Open, рис.2). Файл должен быть исполняемым.

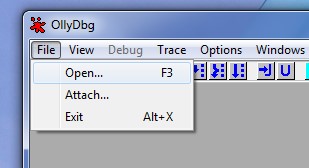


Рис. 2 Меню открытия исследуемого файла

В результате содержимое отлаживаемого файла появится на экране – рис. 3

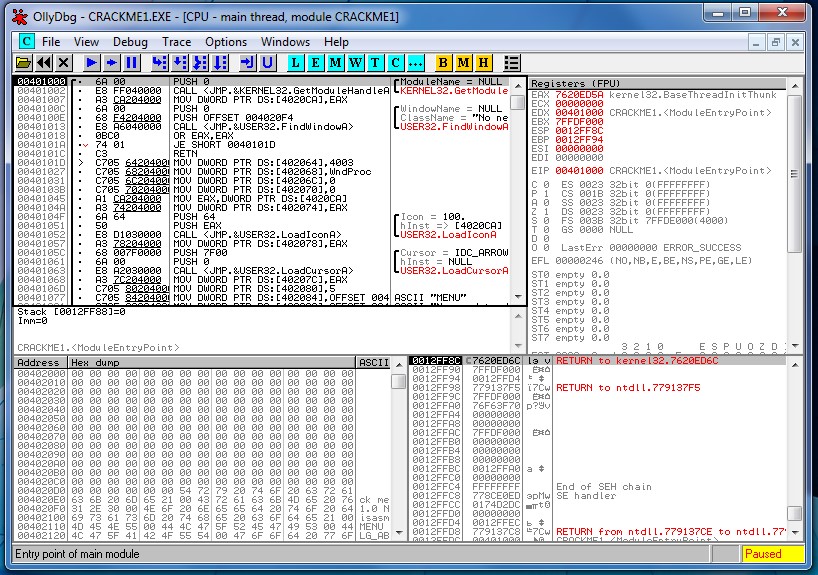


Рис. 3 Главное окно отладчика OllyDbg

Рассмотрим основное рабочее пространство отладчика – рис. 4

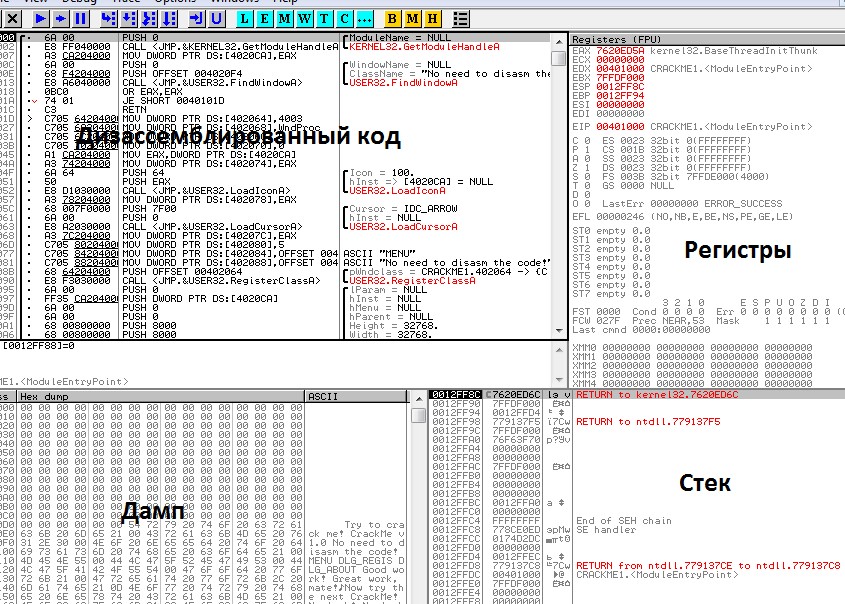


Рис.4 Рабочее пространство отладчика OllyDbg

# 1) Дизассемблированный код (листинг)

Дизассемблированный код отлаживаемой программы. По умолчанию OllyDbg сконфигурирован так, чтобы анализировать программу при ее открытии.

* Колонка адреса команды (**Address**). В данной колонке показан виртуальный адрес команды, который она получает при загрузке модуля в память. Двойной щелчок мышью в данной колонке переводит все адреса в смещения относительно текущего адреса ($, $-2 и т.п.)
* Колонка кода команды (**Hex**). Выделяются код и значение операнда. Кроме этого, в колонке имеются различные значки, которые помогают разобраться в логике программы.
* Колонка команды (**Disassembly**). В этой колонке представлено ассемблерное обозначение команды. Двойной щелчок приводит к появлению окна редактирования, в котором можно исправить команду (рис. 5.

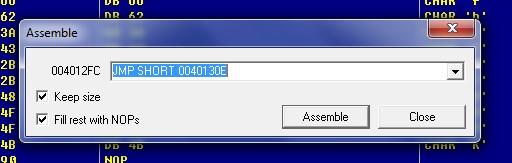


Рис. 5 Окно редактирование ассемблерной команды

Можно записать исправленный текст команды в исполняемый модуль, например, **в первой версии (!)** следующим образом. Правой кнопкой мыши переходим в контекстной меню окна дизассемблера.

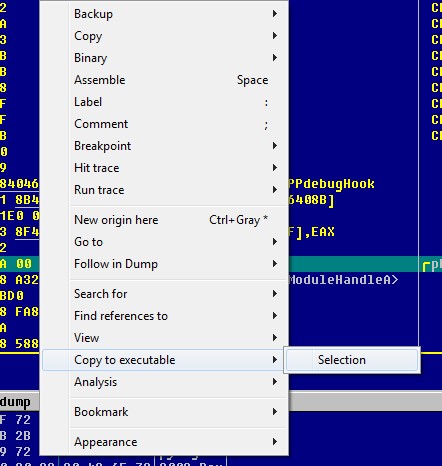


Рис. 6. Выбор из контекстного меню окна дизассемблерного кода пункта копирования кода

В появившемся окне снова переходим в контекстное меню.

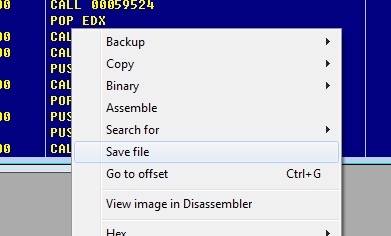


Рис. 7. Выбор из контекстного меню пункта сохранения текущей версии в файл

* Колонка комментария (Comment). Дополнительная информация: имена APIфункций, библиотечных функций и т.д. Двойной щелчок позволяет добавлять свои комментарии (рис. 7).

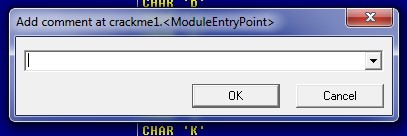


Рис. 7 Окно добавления комментария

Дополнительной опцией можно включить подсветку JUMP (переходов) и

CALL (вызовов). Кликните на листинге правой кнопкой мыши и выберите «APPEARENCE -> HIGHLIGHTING -> JUMPS AND CALLS» (рис. 8).

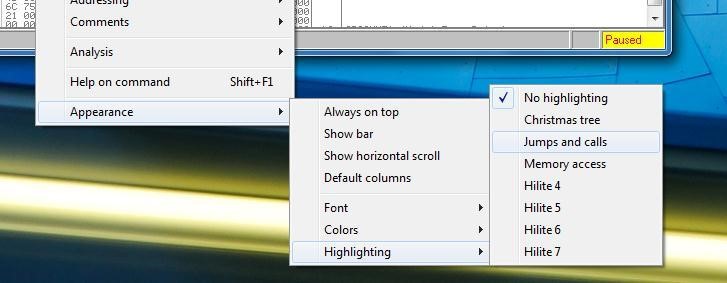


Рис. 8 Выбор дополнительной опции подсветки переходов. Увидим, что call‟ы подсвечены лазурным цветом, а переходы – жёлтым. **2) Регистры** Второе важное окно – это окно отображения содержимого регистров.

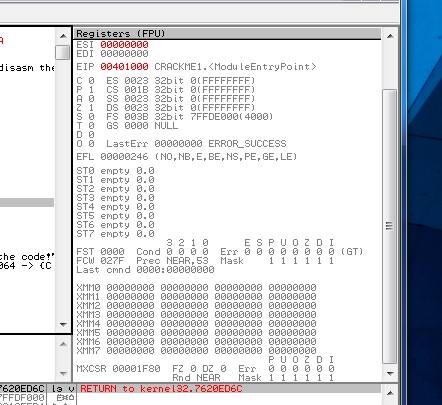


Рис. 9 Окно отображения содержимого регистров в OllyDbg Значения регистров можно изменять «на лету».

# 3) Стек

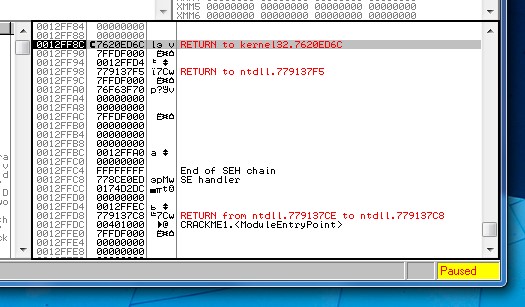


Рис. 10 Окно отображения содержимого стека в OllyDbg

# 4) Дамп

У окна дампа (данных) есть много режимов отображения, которые можно изменять, кликнув правой кнопкой мыши в окне дампа и выбрав тот, который нужен (рис. 11).

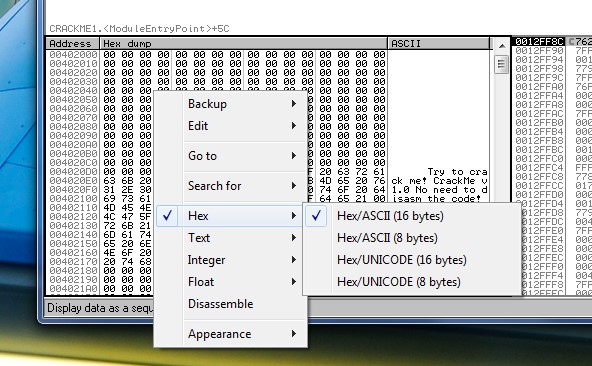


Рис. 11 Окно отображения дампа в OllyDbg

Помимо главного окна, есть окна, недоступные напрямую, но которые могут быть вызваны через кнопки на панели управления.

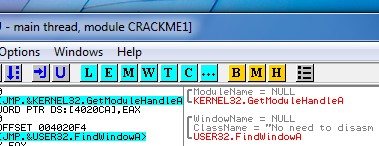


Рис. 12 Дополнительный набор функциональных кнопок в OllyDbg

Приступая к работе с отладчиком, имейте в виду следующее.

 Щелкнув правой кнопкой мыши по любому из окон, вы получите контекстное меню. Оно индивидуально для каждого из четырех окон.  Окна (их содержимое) не являются независимыми. Перечень дополнительных окон представлен в меню View.

**L или** VIEW->LOG показывает нам то, что OllyDbg пишет в окне лога.

Логи можно сконфигурировать на отображение различного рода информации, по умолчанию сохраняется вся информация о запуске, а также информация, связанная с «BREAKPOINTS CONDICIONAL LOGS» (условными логами точек останова). Рассмотрим информацию о запущенном процессе и библиотеках, которые он загружает.

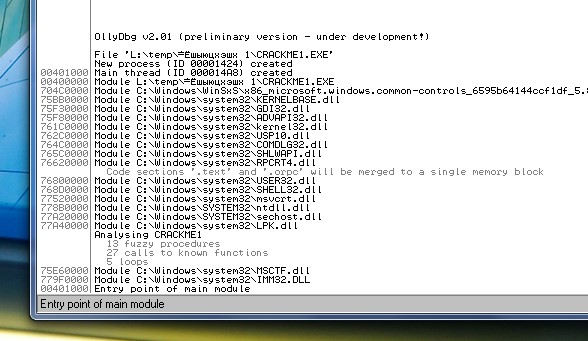


Рис. 13 Окно отображения логов в OllyDbg

Одна из самых главных опций данного окна – ведение лога в файле на тот случай, если хотим сохранять информацию в текстовом файле. Активация опции осуществляется через нажатие правой кнопки мыши и выбора пункта «LOG TO FILE».

**E или** VIEW-> EXECUTABLES показывает список модулей, которые использует программа: exe, dll, ocx и др.

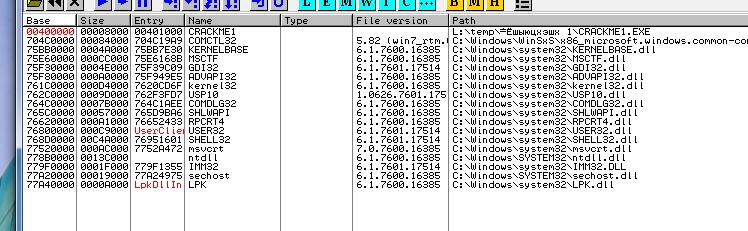


Рис. 14 Окно отображения списка модулей в OllyDbg

**M** или VIEW-> MEMORY отображает память, занятую программой. Здесь видим секции приложения, библиотеки, использующиеся процессом, стек и различные секции, занятые системой.

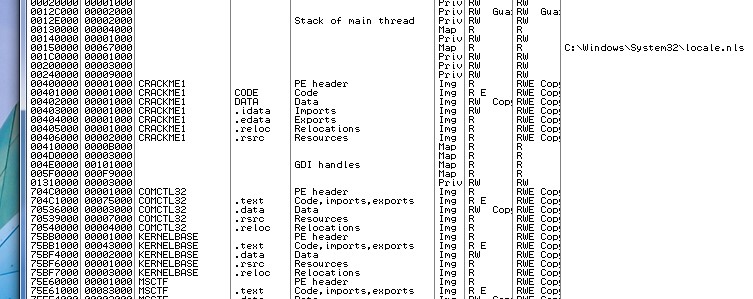


Рис. 15 Окно отображения памяти в OllyDbg

Кликнув правой кнопкой мыши, можем осуществить поиск (SEARCH) в памяти, чтобы найти строки различных видов (текстовые, шестнадцатеричные, юникодовые). Есть возможность подсветить точки останова и изменить права доступа к секциям (SET ACCESS).

**W** или VIEW-> WINDOWS отображает окна программы, но поскольку она еще не выполняется, то и список окон остается пустым.

**T** или VIEW-> THREADS показывает список нитей (потоков) программы.

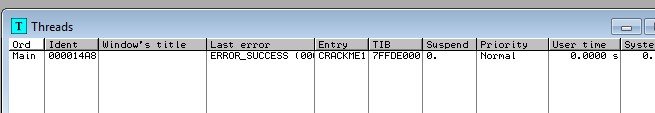


Рис.16 Окно отображения списка потоков в OllyDbg

**C** или VIEW-> CPU возвращает к главному окну OllyDbg.

**«…»** или VIEW-> RUN TRACE отображает результат выполнения команды RUN TRACE. Здесь можно выбрать опцию LOG TO FILE, чтобы сохранить результаты трассировки в текстовом файле.

**B** или VIEW-> BREAKPOINTS вызывает список обычных точек останова, расположенных в программе.

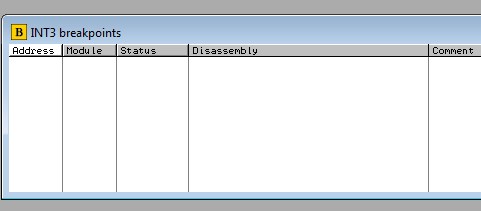


Рис. 17 Окно отображения точек останова в OllyDbg

*Обычные точки останова ставятся* на конкретную команду (клавиша F2 добавляет и удаляет точки останова). В результате адрес команды в первой колонке Address окрашивается по умолчанию в красный цвет. Таким образом, можно проверить состояние регистров, переменных, стека. Остановка осуществляется перед выполнением «помеченной» команды. Обычные точки останова используют стандартный вектор прерывания INT 3.

*Условные точки останова* устанавливаются по нажатию комбинации клавиш Shift **+ F2**. В появившемся окне можно задать условие, при выполнении которого должна быть произведена остановка на данной команде. Например,

EAX == 1

[427070] =1231 – условие определяет остановку, если содержимое ячейки памяти 427070H равно 1231H.

# Просмотр значений переменных

Командой меню можно вызвать окно наблюдения (рис. 18).

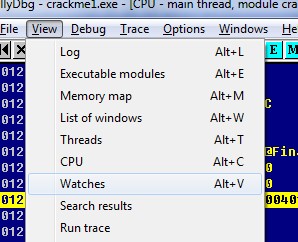


Рис. 18 Выбор из меню View окна наблюдения

В появившемся окне можно добавить наблюдение, т.е. определить выражение, за которым отладчик будет наблюдать (выводить значение этого выражения).

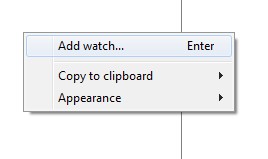


Рис. 19. Добавление выражения для слежения

Ниже на рис. 20 представлено окно наблюдения, содержащее список выражений, значения которых отслеживаются при каждом выполнении команды процессора.

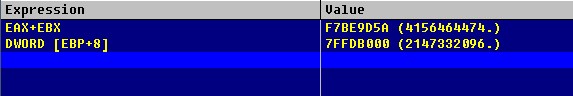


Рис. 20 Пример выражений в окне Watches

При нажатии комбинации **Ctrl + B** появляется окно поиска. Строка для поиска может вводиться в виде последовательности символов, байтов, символов в кодировке Unicode. Для поиска команд используется комбинация **Ctrl + F**.

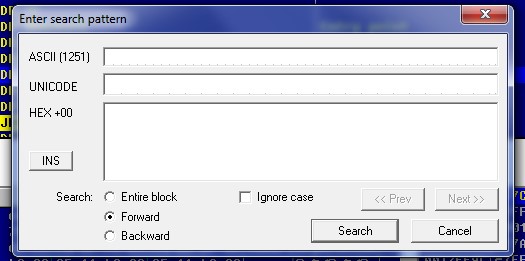


Рис.21 Окно поиска команд

# Горячие клавиши в OllyDbg

F7: выполняет одну строку кода (если находимся на CALL, то переходим внутрь вызванного участка кода).

F8: выполняет одну команду (строку кода) (если находимся на CALL, то просто выполняет вызов без перехода внутрь и переходит на следующую за CALL строку). Выполняя одну команду за другой, мы можем в трех остальных окнах следить за тем, как меняется содержимое регистров, секции данных и секции стека.

F2: устанавливает обычную точку останова на отмеченной строке. Чтобы убрать эту точку останова, надо ещё раз нажать F2.

F9: запускает программу, которая будет выполняться до тех пор, пока не встретит точку останова, не произойдет исключение или же не прекратит работу по каким-либо причинам. Когда программа запущена, в нижнем правом углу OllyDbg отображается слово RUNNING.

Чтобы временно прекратить выполнение программы, нужно нажать F12

или DEBUG-> PAUSE

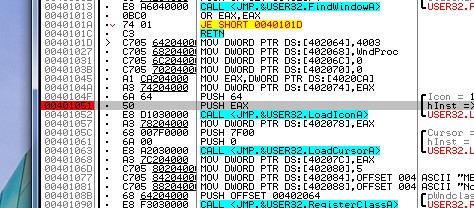


Рис. 22. Установка точек останова в OllyDbg



Рис. 23 Статус выполнения программы в OllyDbg

Видим, что OllyDbg отображает слово PAUSED (пауза). Продолжить выполнение программы можно, нажав F9 или DEBUG-> RUN.

Чтобы закрыть отлаживаемую программу выберите DEBUG-> CLOSE.

2. МЕТОДИКА ВЫПОЛНЕНИЯ

1. В качестве отлаживаемой программы выбрать программу для решения указанной в лаб. работе 3 п.1 задачи.
2. Загрузить отлаживаемую программу в OllyDbg.
3. Ознакомиться с содержимым контекстного меню всех четырех окон отладчика
4. Добавить в дизассемблированный код 3-4 комментария
5. Сохранить в файле дизассемблированный код программы
6. Сохранить в файле содержимое регистров программы.
7. Выполнить программу в пошаговом режиме.
8. Установить 2-3 точки останова и 1-2 точки останова по условию.
9. Сохранить в файле список установленных точек останова.
10. Получить список модулей, используемых отлаживаемой программой
11. Выполнить прямую трассировку программы

* 1. ОТЧЕТ О РАБОТЕ

Готовится в письменном виде один на бригаду. В отчет поместить содержимое файлов, полученных при выполнении пунктов 4, 5, 8, 9, 10 и 11.

* 1. КОНТРОЛЬНЫЕ ВОПРОСЫ

1. Назначение программы OllyDbg. Версии программы.
2. Области применения программы
3. Средства отладки программ в OllyDbg
4. Пошаговая отладка.
5. Прямая трассировка
6. Горячие клавиши OllyDbg
7. Просмотр значений переменных