Пояснения к выполнению
лабораторной работы №5

«Компоновка исполняемого модуля формата Portable Executable»

по дисциплине
«Операционные системы»

1) Формирование объектного модуля формата Common Object File:
    a. Смотрим своё задание, и внимательно читаем описание своей функции в Win32SDK или MSDN, чтобы после завершения последнего пункта не пришлось переделывать всё заново.
    b. Пишем на ассемблере программу для своего задания
    c. Запускаем компилятор ассемблера Microsoft версии не ниже 6.11 (файл называется ml.exe) с командной строкой “ml.exe /c /coff myprog.asm”, где вместо myprog.asm подставляем имя файла с исходным кодом своей программы (при необходимости указать полный путь к файлу)
    d. Если компилятор указал на ошибку в исходном коде – исправляем её и возвращаемся в п.(b)
    e. В случае правильно проделанной последовательности действий в текущем каталоге ml сформирует объектный модуль с расширением «obj» и именем совпадающим с именем вашего файла с исходным кодом.
2) Работа с установкой, или формирование исполнительного модуля PE:
    a. Загружаем установку PELink.exe
    b. В главном меню выбираем «Файл»/«Новый проект».(Ctrl+N)
    c. Всплывшим диалогом ищем сформированный объектный файл.
    d. Читаем хелп, много хелпа… хотя хелпа недостаточно...
        i. Узнаём важные детали
            1. алгоритма отображения exe файла, например – адресса всех секций(и не только секций) выравниваются по страницам вирт. пространства.
            2. алгоритма компоновки PE модуля, например – что границы всех секций в файле выравниваются по грануляции физического носителя, т.е. размер секции в файле кратен размеру сектора.
            3. о секциях, которые обязаны или не обязаны присутствовать в итоговом модуле, например секцию .drective не надо включать в итоговый модуль. Но зато в итоговом модуле можно создать новую секцию “.idata”, хотя вместо этого, таблицу импорта можно поместить в секцию “.data”,
            4. о порядке компоновки секций
    e. Поехали:
        i. Нужно определить, какие секции необходимы в итоговом модуле. У нас есть на что опираться – объектный файл, - большая часть того что там лежит должно присутствовать в EXE файле. Для заданий на эту лабораторную работу обычно это секции “.text” (машинный код), “.data”(секция данных) и дополнительно можно создать секцию “.idata” (таблица импорта). Клик на панель слева, «COFF- файл». Изучаем содержимое объектного нашего файла:
            1. Структура COFF-файла
                a. FILE_HEADER – здесь все общие данные этого объектного файла. Обратите внимание, как Masm определил поле Machine, для этого модуля – этоже значение нужно будет использовать в заголовке EXE файла.
                b. SECTION_HEADERS – Смотрим размеры секций (SizeOfRawData) и какими флагами помечена каждая секция - это тоже пригодится.
                c. SECTIONS – Обращаем внимание что секции .Text соответствует таблица RELOCATION – находим в хелпе что это за таблица. Каждый её элемент (Address;SymIndex;Type)- это указатель PDWORD (обычно) от начала секции на

элемент, значение которого следует определить компоновщику. Обычно в это ссылки на переменные, импортируемые функции и т.д. Значение SymIndex ставит в соответствие этому элементу элемент из таблицы SYMBOL_TABLE.

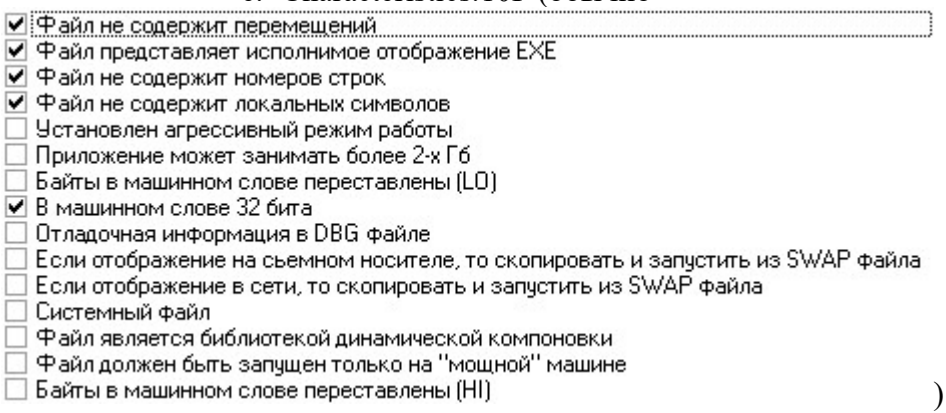
d. SYMBOL_TABLE – ссылочная таблица. В ней каждый элемент – это текстовое значение, которое каким-то образом использовалось в вашей программе, и сопоставленное этому значению местоположение в объектном модуле. Эта информация очень важна. Мы к этой таблице ещё вернёмся.

2. Можно взглянуть на дампы секций объектного модуля и заметить что те адреса секций, которые замечены в таблице RELOCATIONS секции в дампе обозначены нулями.

ii. Теперь посмотрев на устройство объектного модуля, приступаем к производству на его основе исполнительного модуля PE. Клик на панель слева, «заголовок PE-файла»

iii. На раскрывшейся панели выбираем кликом пункты и делаем в них установки

1. DOS-заглушка:
   a. Файловое смещение DOS-заглушки:0 (const)
   b. e_lfanew: 100 (const смещение на новый заголовок)
2. PE-сигнатура:
   a. Файловое смещение PE-сигнатуры: 100 (const)
   b. Значение: NT_SIGNATURE 00004550h (const)
3. Файловый заголовок
   a. Файловое смещение заголовка: 104 (const= DOS_HEADER+PE_SIGNATURE)
   b. Machine: IA-32 (вспоминаем это поле в объектном модуле)
   c. NumberOfSections: (Итоговое количество секций в вашем Exe файле, обычно 3: “.text” и “.data”, “.idata”. После изменения этого значения – создадутся или удалятся секции из вашей модели exe файла, так что поосторожней с этим полем)
   d. TimeDateStamp: 0 (const)
   e. Characteristics:10F (обычно-



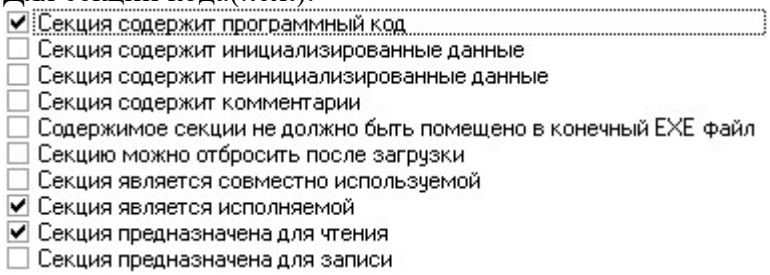
)

4. Дополнительный заголовок
   a. Файловое смещение заголовка: 118 (const)
   b. Magic: 010B (const = 10B - нормальное отображение, 107 – полная копия в физическую память)
   c. Файловое смещение заголовка: 118 (const)
   d. AddressOfEntryPoint: 1000 (Входная точка главного потока = RVA данных секции кода(.text))

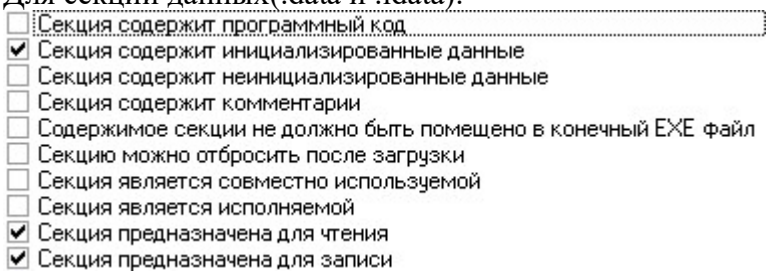
e. ImageBase: 400000 (const . Желательный начальный адрес во flat. Если не хотите проблем оставьте эту величину, долго объяснять почему.)
f. SectionAlignment: 1000 (const Гранулярность физической памяти = размер страницы)
g. FileAlignment: 200 (const Гранулярность физического носителя = размер сектора винчестера)
h. SizeOfImage: 4000 (Суммарный объем от заголовка до конца данных последней секции в вирт. пространстве, следовательно - с учётом того, что границы секций выравниваются по SectionAlignment (грубо говоря))
i. SizeOfHeaders: 400 (const = размер всех заголовков и таблицы секций с учётом гранулярности FileAlignment)
j. SubSystem: windows_gui (const)
k. SizeOfStackReserve: 100000 (const = зарезервированный в Вирт. пространстве объём для стека главного потока)
l. SizeOfStackCommit: 1000 (const = зарезервированный в пространстве физ. Памяти объём для стека главного потока)
m. SizeOfHeapReserve: 100000 (const = зарезервированный объём для главного хипа)
n. SizeOfHeapCommit: 1000 (const = зарезервированный в пространстве физ. памяти объём для главного хипа)
o. ImageDataDirectory – смотри в хелпе «заголовок PE-файла»
    i. VirtualAddress: (пока ничего не трогать, понадобится при настройке таблицы импорта)
    ii. Size: (пока ничего не трогать, понадобится при настройке таблицы импорта)

iv. Теперь вам необходимо заполнить таблицу секций. Вы знаете какие секции из obj файла вам нужны в exe, какие – нет, и какие новые следует добавить. Кроме того вы уже определили количество секций с учётом отброса ненужных. Клик на панель слева, «Таблица секций»
v. На раскрывшейся панели последовательно (порядок последовательности взять из структуры COFF-файла) выбираем кликом каждую секцию и делаем в ней следующие установки:
    1. Файловое смещение таблицы секций: 1F8 (для всех секций одно и тоже постоянное значение)
    2. Name: (Название секции)
    3. VirtualSize: (размер данной секции в виртуальном пространстве, обычно равно SizeOfRawData)
    4. VirtualAddress: (RVA секции)
    5. SizeOfRawData: (Размер дампа секции в файле)
    6. PointerToRawData: (EXE-FileOffset на данную секцию = размер_заголовков_и_таблицы_секций + размер_предстоящ_секций = 400 + размер_предстоящ_секций)
    7. Characteristics:

a. Для секции кода(.text):



b. Для секции данных(.data и .idata):



vi. Теперь заполняем содержимое секций.
   1. Клик на панели слева «Секции».
   2. Выбираем очередную секцию. Открывается окно с её содержимым в Exe файле.
   3. Если это не секция ".idata" то
      a. Клик мышкой на ячейку (0;0)
      b. В нижней части всплывшего окна выбираем вкладку «Вставка из секции COFF»
      c. устанавливаем в поле «Секция COFF» открывшейся панели имя совпадающее с именем этой секции.
      d. устанавливаем в поле «Копировать всю секцию» открывшейся панели галочку.
      e. Нажимаем кнопку «Копировать».
   4. Если это секция ".idata", то в неё необходимо занести таблицу импорта. Смотри в хелпе «Выполнение лабораторной работы\Шаг5 – Создание таблицы импорта» и «Описание форматов COFF и PE\Импортирование в PE файлах»
      a. Заполнить таблицу импорта, первый элемент с адреса 0;0. Таблица состоит из структур IMAGE_IMPORT_DESCRIPTOR. Последняя структура должна быть заполнена «0» (именно так система определяет длину таблицы). Каждая структура соответствует одной библиотеке и содержит в себе ссылки на имя библиотеки, «оригинальный массив» ссылок, «первичный массив» (грубый перевод) и прочее (можно заполнить «0»). Два указанных массива должны копировать друг друга. При загрузке первичный массив перезаписывается реальными ссылками на импортируемые функции. Элемент массива (оригинального или первичного) – ссылка на структуру {WORD; ANSIz;} в первое поле WORD можно заполнять порядковыми номерами или вообще «0», а ANSIz должен содержать имя функции. Последний элемент массива д.б. заполнен «0».
      b. Отметить в Доп. Заголовке в панели IMAGE_DATA_DIRECTORY выбрать из списка «1 Import

Directory» и указать в поле VirtualAddress: RVA секции .idata (3000), а в поле Size: VirtualSize из секции .idata

5. Если этой секции в структуре COFF соответствует массив RELOCATIONS, то необходимо проставить ссылки, которым сопоставлены в SYMBOL_TABLE элементы типа STATIC. Смотри чёткий алгоритм в хелпе «Выполнение лабораторной работы\Шаг4 – Создание секций PE-файла» в самом низу.
    a. Смотрим в таблицу RELOCATION
    b. Берём в рассмотрение очередной элемент (Address;SymIndex;Type). Назначаем выбранному элементу имя currRE
    c. Открываем в SYMBOL_TABLE элемент с индексом SymIndex. Назначаем currSTE=Выбранному элементу
    d. Если currSTE.StorageClass равен STATIC, то выполняем подпункты, иначе переход в п.(e)
        i. В данной секции по адресу currRE.Address пишем в обратном порядке байты абсолютного адреса в виртуальной памяти элемента описываемого в currSTE. Этот адрес = ДопЗаголовок.ImageBase + ТаблицаСекций.СекцияПоНомеру(currSTE.SectionNumber).VirtualAddress+currSTE.Value
    e. Если это был не последний currRE переход в п.(b)
6. Закрываем окно этой секции.
7. Если эта секция была не последней, то возвращаемся в п.2
8. Теперь, когда заполнена секция .idata следует заполнить в секциях с RELOCATION те ссылки, которым сопоставлен элемент из SYMBOL_TABLE с типом EXTERNAL. Это делается аналогично элементам STATIC, но в этот раз нужно вставлять абсолютные адреса элементов «первичного массива», каждый из которых соответствует одной функции.

vii. Выбираем в гл. меню «Компоновщик\Выполнить PE-файл». При удачном результате будет сформирован исполнительный файл

3) Запуск программы в отладчике SoftIce