

Материалы к занятию 18 марта 2020-03-19

Тема: Многомодульные программы (соответствующие программы находятся в папке 20\_03\_04 В подкаталогах COM и EXE).

**Первый вариант:** рассмотрим создание многомодульной (состоящей из трех модулей) программы типа .com (подкаталог COM).

Главный модуль (**main**) начинается с объявления элементов связи с другими модулями. Внешние элементы (**extrn**) – процедуры **prax:near**, **wk:near**, **disp:near** и переменная **num2:word** (про них будет разговор в модулях, в которых они описаны). Публичные элементы (**public** – элементы, описанные здесь, но доступные извне) – переменные **num** (количество печатаемых символов для процедуры в **prn2**) и **text** (адрес текста, печатаемого в той же процедуре).

Затем в главном модуле определены объявленные ранее публичными переменные **num** (равная 9) и **text** (здесь час текст, а публичным объявлен его адрес).

Затем идет инициализация переменной **num2**. Хотя эта переменная и определена во вспомогательном модуле, но инициализация ее должна производиться в главном модуле (из-за особенностей многомодульных программ типа .com). Следом значение переменной **num2** записывается в регистр **CX**, из которого это значение и будет впоследствии печататься.

Далее начинается цикл печати (по **CX**). Прежде всего, **CX** сохраняется в стеке, так как вызываемая ниже процедура **disp** портит этот регистр. Вызываемая процедура печатает слово «осталось», затем из стека восстанавливается **CX**, его значение записывается в **AX** и вызывается процедура печати четырехзначного шестнадцатеричного числа, показывающего, что осталось в **CX**.

Затем вызывается процедура печати «перевод строки + возврат каретки», и, наконец, команда замыкания цикла **loop**. После обнуления **CX** происходит выход из цикла и программа ожидает нажатия любой клавиши (функций 0 клавиатурного прерывания **Int 16h**).

Следует обратить внимание на то, что только в главном модуле после ключевого слова **end** указывается точка входа (**start**).

```
;Главный модуль
Assume CS: Code, DS: Code
extrn prax:near      ;Вн.проц.печати AX
extrn wk:near       ;Вн.проц.выв.ВК+ПС
extrn disp:near     ;Вн.проц.выв.строки
extrn num2:word     ;Внешняя переменная
public num          ;публ.перем. для disp (кол-во)
public text         ;публ.перем. для disp (адрес)
Code SEGMENT
        org 100h
start:  jmp begin
num     dw  9 ;публ.перем.кол.печ.символов
```

```

textdb  'Осталось ';публ.перем.адр.печати
begin:  mov num2,5 ;Иниц, num2 (т.к.COM прогр.)
        mov cx,num2 ;num2 ->CX
b1:     pushcx    ;Сохр.CX в стеке
        calldisp ;Печ.стр.num симв.из text
        pop      cx    ;Восст.CX из стека
        mov      ax,cx;То, что будет в AX
        callprax ;Печать AX
        call     wk    ;ПС+ВК
        loop    b1    ;Цикл* пока CX не 0
        xor     ah,ah  ;0 функ. клав.прерыв.
        int     16h   ;Клав.прерыв.
        int     20h   ;Вых.из прогр.
Code ends ;Конец сегмента
        end start ;Точка входа (только в главн.модуле)

```

Модуль печати 1 (**prn1**) содержит процедуры печати четырехзначного шестнадцатеричного числа **prax** и комбинации «перевод строки + возврат каретки» **wk**. Иак как исполняются они за пределами данного модуля (в главном модуле), они объявляются в этом модуле публичными (**public**). Следует обратить внимание на то, что в конце модуля после ключевого слова **end** не указывается адрес точки входа, так как это не основной модуль.

```

;Модуль печати 1
Assume CS: Code, DS: Code
public prax
public wk
Code SEGMENT
prax proc near;Печать 4-зн.hex числа из AX
        pushbx   ;Сохр.BX в стек
        pushcx   ;Сохр.CX в стек
        mov      cx,4;Кол-во сдвигов по 4 влево
pr1:    rol ax,4;Цикл.сдвиг ст.тетр.на место младшей
        push     ax  ;Сохр.сдвинутого AX
        callprsn;Печать 1 hex цифры
        pop      ax  ;Восст.сдвин.AX из стека
        looppr1 ;Цикл ar CX
        pop      cx  ;Восст.CX из стека
        pop      bx  ;Восст.BX из стека
        ret       ;Возврат из проц.
prax endp      ;Конец процедуры
prsn proc near;Печать 1 hex цифры из мл.тетр.AL
        and      al,0fh ;Выдел.мл.тетрады
        add      al,30h ;Преобр.цифры в ее код символа
        cmp      al,39h ;Проверка диапазона 0-9
        jle     print ;Переход, если AL<="9"
        add      al,7  ;Если нет, переход к буквам A-F

```

```

; Печать 1 ASCII символа
print proc near ;Вложенная процедура (целиком)
    mov     bx,0fh ;Белый цвет (0fh)на черном фоне (0)
    mov     ah,0eh ;Вывод симв.в режиме телетайпа
    int     10h    ;Видеопрерывание
    ret
;Возврат из проц.
print endp      ;Конец процедуры
prsn  endp      ;Конец процедуры
wk:   mov     al,0ah ;Код перевода строки ПС
    call    print ;Печать одного символа
    mov     al,0dh ;Код возврата каретки ВК
    call    print ;Печать одного символа
    ret
;Возврат из проц.
Code ends ;Конец сегмента
end ;Без точки входа (вспом.модуль)

```

Модуль печати 2 (**prn2**) содержит процедуру печати сообщения о том, сколько осталось в переменной **num**. Для печати (вывода на консоль используется функция записи в файл (40h прерывания int 21h). В качестве дескриптора используется предопределенный (заранее назначенный) дескриптор консоли 2. Количество выводимых символов и адрес выводимого текста в функции определены в главном модуле **main** (переменные **num** и **text**, соответственно). Так как они определены за пределами данного модуля, они объявляются в этом модуле внешними (**extrn**).

Процедура **disp** и переменная **num2**, описанные в данном модуле, используются в главном модуле, поэтому здесь они определены как публичные (**public**). Следует обратить внимание на то, что в конце модуля после ключевого слова **end** так же, как и в **prn1** не указывается адрес точки входа, так как это не основной модуль.

```

;Модуль печати 2
Assume CS: Code, DS: Code
public disp
extrn num:word ;Внешняя переменная
extrn text:word ;Внешняя переменная
public num2 ;публ.перем. кол-во выводов
Code SEGMENT
disp proc near;
    mov     cx,num ;Кол-во выводимых символов;
    mov     ah,40h ;Запись в файл
    lea     dx,text;Выводимый текст
    mov     bx,2 ;Дескриптор консоли (выв)
    int     21h
    ret
disp endp
num2 dw ?
Code ends

```

end

Общая картина взаимодействия модулей показана на рисунке 1.

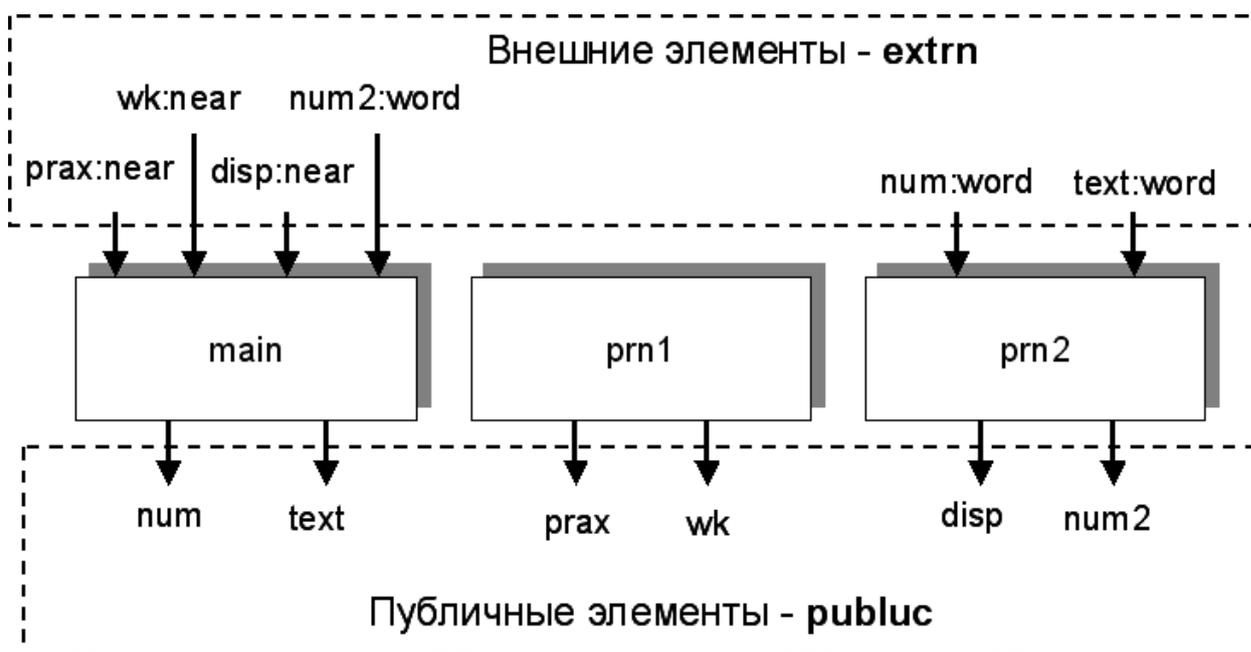


Рисунок 1 – Взаимодействие модулей в многомодульной программе

Для получения исполняемой программы следует скомпилировать каждый модуль самостоятельно (в произвольном порядке), получив при этом три объектных модуля **main.obj**, **prn1.obj**, **prn2.obj**.

Для компоновки программой **tlink** компонуемые модули указываются следующим образом – сначала главный модуль, а затем остальные.

Пакетный файл, используемый для получения исполняемой программы 1.bat, расположенный в подкаталоге COM приведен ниже.

```
del *.obj
tasm main /l
tasm prn1 /l
tasm prn2 /l
tlink main prn2 prn1 /m /l /s /t
```

Первая команда удаляет все ранее созданные объектные файлы (с расширением **.obj**). Три следующие команды – компиляция модулей. Порядок может быть другой, главное, чтобы были скомпилированы все модули. Ключ /l задан в командах компиляции для создания листингов компиляции (соответствующие файлы с расширением **.lst**).

После команды **tlink** указаны компокуемые модули, причем на первом месте стоит главный модуль **main** (порядок перечисления вспомогательных модулей не важен). Ключи, указанные в командной строке имеют следующий смысл:

- /m – создать более полный, чем по умолчанию файл карты загрузки,
- /l – использовать (если необходимо) в карте загрузки номера строк,
- /s – в файл карты включить информацию о сегментах,
- /t – как обычно, нужен для создания файла **.com**.

После запуска программы **main.com** убедиться в том, что программа выводит на экран 5 строк.

Проведите следующие эксперименты:

1. Измените в главном модуле в строке `begin: mov num2, 5` значение на 4. Создайте заново программу, запустив пакетный файл **1.bat**. Запустите **main.com** и убедитесь, что на экран теперь выводится 4 строки.

2. Измените в главном модуле в строке `num dw 9` значение на 7. Создайте заново программу, запустив пакетный файл **1.bat**. Запустите **main.com** и убедитесь, что на экран теперь слово «осталось» теперь выводится без мягкого знака и пробела после него.

3. Закомментируйте в главном модуле в строке `begin: mov num2, 5` команду (строка будет выглядеть так `begin: ;mov num2, 5`). В модуле печати 2 (**prn2**) в строке `num2 dw ?` замените знак вопроса на цифру 5 (`num2 dw 5`). Создайте заново программу, запустив пакетный файл **1.bat**. Посмотрите, что получилось, и попробуйте объяснить это.

4. Если закомментировать всю строку `begin: mov num2, 5` (вместе с меткой) то компилятор выдаст ошибку. Попробуйте, и объясните эффект.

**Второй вариант:** рассмотрим создание многомодульной (состоящей из трех модулей) программы типа .EXE (подкаталог EXE).

Коды всех модулей остаются практически неизменными (за исключением сообщению компилятору о сегментах).

Так в главном модуле `model small` – модель в которой сегмент кода и сегмент данных могут занимать по 64 КБ.

Далее создается стек: `stack 256` определяя при этом и сегмент стека и его размер.

Далее определяется сегмент кода `codeseg`. Обратите внимание на то, что нет замыкающей метки у сегмента.

Получившееся распределение сегментов можно посмотреть в файлах **main.lst** и **main.map**. Попробуйте разобраться в том, что там указано насчет сегментов.

Обратите внимание на то, что в главном модуле нет строки инициализации переменной **num2**. Это не нужно, так как в .exe программе нормально работает инициализация во вспомогательных модулях (см. строку `num2 dw 4`).

Сегменты во вспомогательных модулях определяются аналогично главному модулю. Стек во вспомогательных модулях не создается, а точка входа в конце не указывается (как и программе типа .com).

```
;Главный модуль
model small
stack 256
codeseg
;*****
```

```

extrn prax:near      ;Вн.проц.печати АХ
extrn wk:near        ;Вн.проц.выв.ВК+ПС
extrn disp:near      ;Вн.проц.выв.строки
extrn num2:word      ;Внешняя переменная
public num           ;публ.перем. для disp (кол-во)
public text          ;публ.перем. для disp (адрес)
start:  push cs      ;DS=CS (Здесь обязательно)
        pop ds
        mov     cx,num2 ;Кол-во выводов
b1:  pushcx
     calldisp ;Печ.стр.num симв.из text
     pop cx
     mov ax,cx;То,что будет в АХ
     callprax ;Печать АХ
     call     wk
     loop    b1 ;Цикл* пока СХ не 0
     xor ah,ah ;0 функ. клав.прер.
     int 16h   ;Клав.прерывание
;*****
     mov ah,4ch ;Вых.из прогр.(EHE)
     int 21h
num dw 9          ;Длина сообщ.(публ.)
textdb 'Осталось ';Сообщение (публ.)
     end start ;Точка входа (только в главн.модуле)

```

### Вспомогательный модуль **prn1**.

```

;Модуль печати 1
public prax
public wk
model small
codeseg
praxprocnear;Печать 4-зн.hex числа из АХ
     pushbx  ;Сохр.ВХ в стек
     pushcx  ;Сохр.СХ в стек
     mov cx,4;Кол-во сдвигов по 4 влево
pr1:  rol ax,4;Цикл.сдвиг ст.тетр.на место младшей
     push    ax  ;Сохр.сдвинутого АХ
     callprsn;Печать 1 hex цифры
     pop ax   ;Восст.сдвин.АХ из стека
     looppr1 ;Цикл ar СХ
     pop cx  ;Восст.СХ из стека
     pop bx  ;Восст.ВХ из стека
     ret    ;Возврат из проц.
praxendp          ;Конец процедуры
; Печать одной hex цифры из мл. тетр. al

```

```

prsnprocnear
    and     al,0fh ;Выдел.мл.тетрады
    add     al,30h ;Преобр.цифры в ее код символа
    cmp     al,39h ;Проверка диапазона 0-9
    jle     print ;Переход, если AL<="9"
    add     al,7 ;Если нет, переход к буквам A-F
; Печать 1 ASCII символа
print     procnear ;Вложенная процедура (целиком)
    mov    bx,0fh ;Белый цвет (0fh)на черном фоне (0)
    mov    ah,0eh ;Вывод симв.в режиме телетайпа
    int    10h ;Видеоперерывание
    ret
;Возврат из проц.
print     endp ;Конец процедуры
prsnendp ;Конец процедуры
; ПС+ВК
wk: mov    al,0ah ;Код перевода строки ПС
    call   print ;Печать одного символа
    mov    al,0dh ;Код возврата каретки ВК
    call   print ;Печать одного символа
    ret
;Возврат из проц.
end ;Без точки входа (вспом.модуль)

```

### Вспомогательный модуль **prn2**.

```

;Модуль печати 2
public disp
extrn num:word ;Внешняя переменная
extrn text:word ;Внешняя переменная
public num2 ;публ.перем. кол-во выводов

model small
codeseg
disp: mov    cx,num ;Кол-во выводимых символов;
    mov    ah,40h ;Запись в файл
    lea    dx,text;Выводимый текст
    mov    bx,2 ;Дескриптор консоли (выв)
    int    21h
    ret
num2 dw    4 ;публ.перем. кол. выводов
end

```

### Пакетный файл **2.bat** компиляции **.exe** программы.

```

del *.obj
del *.com
tasm main /1
tasm prn1 /1

```

```
tasm prn2 /l  
tlink main prn1 prn2 /m /l /s
```

Из распечатки файла **2.bat** видно, что все действия похожи на то, что делалось для создания программы типа **.com**. Отличия всего два – первое (главное) при компоновке (**tlink**) не используется ключ **/t**, так как на сей раз создается программа типа **.exe**, второе – в первой строке удаляются все **.com** файлы в каталоге (чтобы не мешались).

Проделайте с **.exe** программой эксперименты, аналогичные тем, что с программой типа **.com**.

Все возникающие вопросы посылайте на почту.