МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение

высшего образования

**«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н.Г.ЧЕРНЫШЕВСКОГО»**

**Практическая работа 1**

Студента 2курса 212 группы

Направления 02.03.02«Фундаментальная информатика и информационные технологии»

Факультет компьютерных наук и информационных технологий

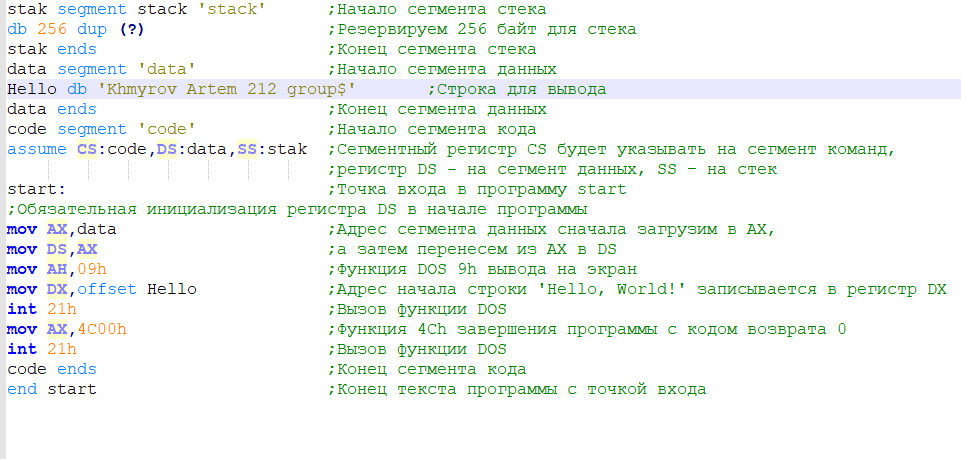
Хмырова Артема Михайловича

Саратов 2019

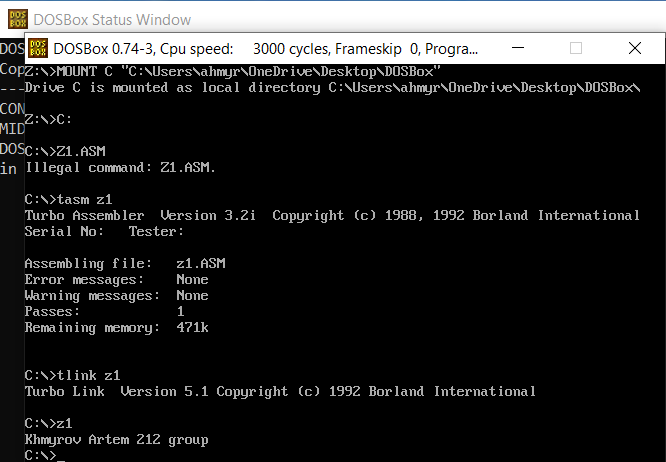
**Задание 1.1.** Измените программы из примеров 1, 2 и 3 так, чтобы они выводили на экран ваши фамилию, имя и номер группы. Используя командные файлы, подготовьте к выполнению и запустите программы из примеров 1, 2 и 3. Убедитесь, что они выводят на экран нужный текст и успешно завершаются.

**Ответ:**

**Пример 1:**



Вывод:



**Листинг:**

**Turbo Assembler Version 3.2 09/24/19 23:52:46 Page 1**

**z1.asm**

**1 0000 stak segment stack 'stack' ;Начало сегмента стека**

**2 0000 0100\*(??) db 256 dup (?) ;Резервируем 256 байт для стека**

**3 0100 stak ends ;Конец сегмента стека**

**4 0000 data segment 'data' ;Начало сегмента данных**

**5 0000 4B 68 6D 79 72 6F 76+ Hello db 'Khmyrov Artem 212 group$' ;Строка для вывода**

**6 20 41 72 74 65 6D 20+**

**7 32 31 32 20 67 72 6F+**

**8 75 70 24**

**9 0018 data ends ;Конец сегмента данных**

**10 0000 code segment 'code' ;Начало сегмента кода**

**11 assume CS:code,DS:data,SS:stak ;Сегментный регистр CS будет +**

**12 указывать на сегмент команд,**

**13 ;регистр DS - на сегмент данных, SS – +**

**14 на стек**

**15 0000 start: ;Точка входа в программу start**

**16 ;Обязательная инициализация регистра DS в начале +**

**17 программы**

**18 0000 B8 0000s mov AX,data ;Адрес сегмента данных сначала +**

**19 загрузим в AX,**

**20 0003 8E D8 mov DS,AX ;а затем перенесем из AX в DS**

**21 0005 B4 09 mov AH,09h ;Функция DOS 9h вывода на экран**

**22 0007 BA 0000r mov DX,offset Hello ;Адрес начала строки 'Hello, World!' +**

**23 записывается в регистр DX**

**24 000A CD 21 int 21h ;Вызов функции DOS**

**25 000C B8 4C00 mov AX,4C00h ;Функция 4Ch завершения программы с +**

**26 кодом возврата 0**

**27 000F CD 21 int 21h ;Вызов функции DOS**

**28 0011 code ends ;Конец сегмента кода**

**29 end start ;Конец текста программы с точкой +**

**30 входа**

**Turbo Assembler Version 3.2 09/24/19 23:52:46 Page 2**

**Symbol Table**

**Symbol Name Type Value**

**??DATE Text "09/24/19"**

**??FILENAME Text "z1 "**

**??TIME Text "23:52:46"**

**??VERSION Number 0314**

**@CPU Text 0101H**

**@CURSEG Text CODE**

**@FILENAME Text Z1**

**@WORDSIZE Text 2**

**HELLO Byte DATA:0000**

**START Near CODE:0000**

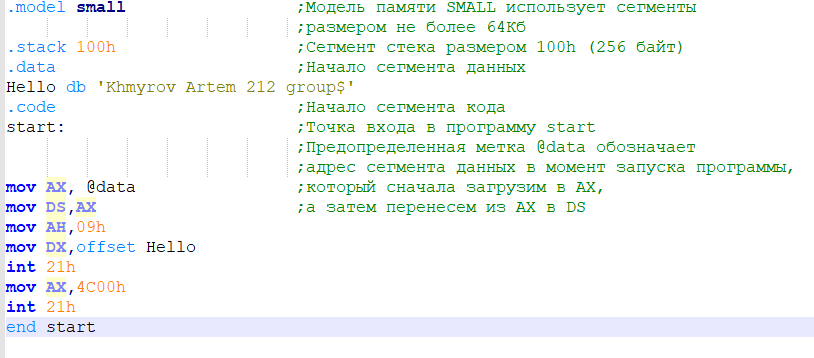
**Groups & Segments Bit Size Align Combine Class**

**CODE 16 0011 Para none CODE**

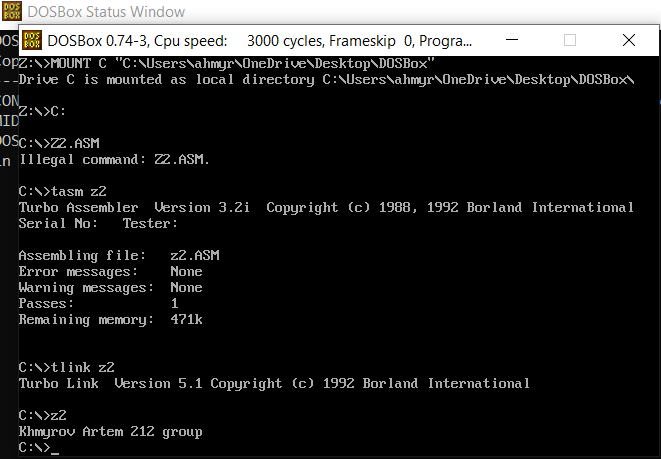
**DATA 16 0018 Para none DATA**

**STAK 16 0100 Para Stack STACK**

**Пример 2:**

****

Вывод:

****

**Листинг:**

**Turbo Assembler Version 3.2 09/24/19 23:53:56 Page 1**

**z2.asm**

**1 0000 .model small ;Модель памяти SMALL использует +**

**2 сегменты**

**3 ;размером не более 64Кб**

**4 0000 .stack 100h ;Сегмент стека размером 100h (256 байт)**

**5 0000 .data ;Начало сегмента данных**

**6 0000 4B 68 6D 79 72 6F 76+ Hello db 'Khmyrov Artem 212 group$'**

**7 20 41 72 74 65 6D 20+**

**8 32 31 32 20 67 72 6F+**

**9 75 70 24**

**10 0018 .code ;Начало сегмента кода**

**11 0000 start: ;Точка входа в программу start**

**12 ;Предопределенная метка @data +**

**13 обозначает**

**14 ;адрес сегмента данных в момент +**

**15 запуска программы,**

**16 0000 B8 0000s mov AX, @data ;который сначала загрузим в AX,**

**17 0003 8E D8 mov DS,AX ;а затем перенесем из AX в DS**

**18 0005 B4 09 mov AH,09h**

**19 0007 BA 0000r mov DX,offset Hello**

**20 000A CD 21 int 21h**

**21 000C B8 4C00 mov AX,4C00h**

**22 000F CD 21 int 21h**

**23 end start**

**Turbo Assembler Version 3.2 09/24/19 23:53:56 Page 2**

**Symbol Table**

**Symbol Name Type Value**

**??DATE Text "09/24/19"**

**??FILENAME Text "z2 "**

**??TIME Text "23:53:56"**

**??VERSION Number 0314**

**@32BIT Text 0**

**@CODE Text \_TEXT**

**@CODESIZE Text 0**

**@CPU Text 0101H**

**@CURSEG Text \_TEXT**

**@DATA Text DGROUP**

**@DATASIZE Text 0**

**@FILENAME Text Z2**

**@INTERFACE Text 00H**

**@MODEL Text 2**

**@STACK Text DGROUP**

**@WORDSIZE Text 2**

**HELLO Byte DGROUP:0000**

**START Near \_TEXT:0000**

**Groups & Segments Bit Size Align Combine Class**

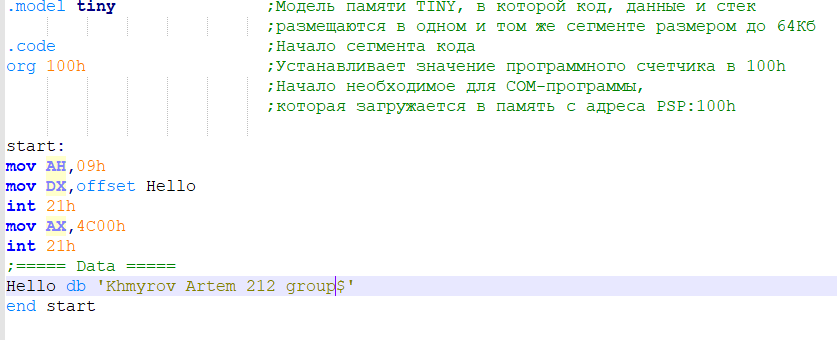
**DGROUP Group**

**STACK 16 0100 Para Stack STACK**

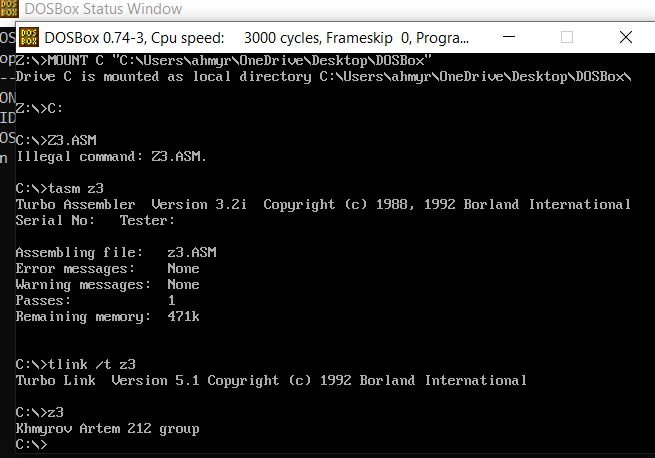
**\_DATA 16 0018 Word Public DATA**

**\_TEXT 16 0011 Word Public CODE**

**Пример 3:**

****

Вывод:

****

**Листинг:**

**Turbo Assembler Version 3.2 09/24/19 23:55:15 Page 1**

**z3.asm**

**1 0000 .model tiny ;Модель памяти TINY, в которой код, +**

**2 данные и стек**

**3 ;размещаются в одном и том же сегменте+**

**4 размером до 64Кб**

**5 0000 .code ;Начало сегмента кода**

**6 org 100h ;Устанавливает значение программного+**

**7 счетчика в 100h**

**8 ;Начало необходимое для COM-программы,**

**9 ;которая загружается в память с +**

**10 адреса PSP:100h**

**11**

**12 0100 start:**

**13 0100 B4 09 mov AH,09h**

**14 0102 BA 010Cr mov DX,offset Hello**

**15 0105 CD 21 int 21h**

**16 0107 B8 4C00 mov AX,4C00h**

**17 010A CD 21 int 21h**

**18 ;===== Data =====**

**19 010C 4B 68 6D 79 72 6F 76+ Hello db 'Khmyrov Artem 212 group$'**

**20 20 41 72 74 65 6D 20+**

**21 32 31 32 20 67 72 6F+**

**22 75 70 24**

**23 end start**

**Turbo Assembler Version 3.2 09/24/19 23:55:15 Page 2**

**Symbol Table**

**Symbol Name Type Value**

**??DATE Text "09/24/19"**

**??FILENAME Text "z3 "**

**??TIME Text "23:55:15"**

**??VERSION Number 0314**

**@32BIT Text 0**

**@CODE Text DGROUP**

**@CODESIZE Text 0**

**@CPU Text 0101H**

**@CURSEG Text \_TEXT**

**@DATA Text DGROUP**

**@DATASIZE Text 0**

**@FILENAME Text Z3**

**@INTERFACE Text 00H**

**@MODEL Text 1**

**@STACK Text DGROUP**

**@WORDSIZE Text 2**

**HELLO Byte DGROUP:010C**

**START Near DGROUP:0100**

**Groups & Segments Bit Size Align Combine Class**

**DGROUP Group**

**\_DATA 16 0000 Word Public DATA**

**\_TEXT 16 0124 Word Public CODE**

**Задание 1.2.** Заполните таблицы трассировки для программ вывода строки на экран из примеров 1, 2 и 3.

**Ответ:**

Таблица трассировки для программы 1:

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Шаг** | **Машинный код** | **Команда** | **Регистры** | | | | | | | | | **Флаги** |
|  |  |  | **AX** | **BX** | **CX** | **DX** | **SP** | **DS** | **SS** | **CS** | **IP** | **CZSOPAID** |
| **1** | **B8BD48** | **mov,**  **ax,48BD** | **0000** | **0000** | **0000** | **0000** | **0100** | **489D** | **48AD** | **48BF** | **0000** | **0000** |
| **2** | **8ED8** | **mov,**  **ds,ax** | **48BD** | **0000** | **0000** | **0000** | **0100** | **489D** | **48AD** | **48BF** | **0003** | **0003** |
| **3** | **B409** | **mov,**  **ah,09** | **48BD** | **0000** | **0000** | **0000** | **0100** | **48BD** | **48AD** | **48BF** | **0005** | **0005** |
| **4** | **BA0000** | **mov,**  **dx,0000** | **09BD** | **0000** | **0000** | **0000** | **0100** | **48BD** | **48AD** | **48BF** | **0007** | **0007** |
| **5** | **CD21** | **int,**  **21** | **09BD** | **0000** | **0000** | **0000** | **0100** | **48BD** | **48AD** | **48BF** | **000A** | **000A** |
| **6** | **B8004C** | **mov,**  **ax,4C00** | **09BD** | **0000** | **0000** | **0000** | **0100** | **48BD** | **48AD** | **48BF** | **000C** | **000C** |
| **7** | **CD21** | **int,**  **21** | **4C00** | **0000** | **0000** | **0000** | **0100** | **48BD** | **48AD** | **48BF** | **000F** | **000F** |

Текст командного файла

tasm Z1

tlink Z1

Z1

td Z1

Таблица трассировки для программы 2:

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Шаг** | **Машинный код** | **Команда** | **Регистры** | | | | | | | | | **Флаги** |
|  |  |  | **AX** | **BX** | **CX** | **DX** | **SP** | **DS** | **SS** | **CS** | **IP** | **CZSOPAID** |
| **1** | **B8AF48** | **mov,**  **ax,48AF** | **0000** | **0000** | **0000** | **0000** | **0100** | **489D** | **48B1** | **48AD** | **0000** | **0000** |
| **2** | **8ED8** | **mov,**  **ds,ax** | **48AF** | **0000** | **0000** | **0000** | **0100** | **489D** | **48B1** | **48AD** | **0003** | **0003** |
| **3** | **B409** | **mov,**  **ah,09** | **48BD** | **0000** | **0000** | **0000** | **0100** | **48AF** | **48B1** | **48AD** | **0005** | **0005** |
| **4** | **BA0000** | **mov,**  **dx,0000** | **09AF** | **0000** | **0000** | **0000** | **0100** | **48AF** | **48B1** | **48AD** | **0007** | **0007** |
| **5** | **CD21** | **int,**  **21** | **09AF** | **0000** | **0000** | **0000** | **0100** | **48AF** | **48B1** | **48AD** | **000A** | **000A** |
| **6** | **B8004C** | **mov,**  **ax,4C00** | **09AF** | **0000** | **0000** | **0000** | **0100** | **48AF** | **48B1** | **48AD** | **000C** | **000C** |
| **7** | **CD21** | **int,**  **21** | **4C00** | **0000** | **0000** | **0000** | **0100** | **48AF** | **48B1** | **48AD** | **000F** | **000F** |

Текст командного файла

tasm Z2

tlink Z2

Z2

td Z2

Таблица трассировки для программы 3:

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Шаг** | **Машинный код** | **Команда** | **Регистры** | | | | | | | | | **Флаги** |
|  |  |  | **AX** | **BX** | **CX** | **DX** | **SP** | **DS** | **SS** | **CS** | **IP** | **CZSOPAID** |
| **1** | **B409** | **mov,**  **ah,09** | **0000** | **0000** | **0000** | **0000** | **FFFE** | **489D** | **489D** | **489D** | **0100** | **0100** |
| **2** | **BA0C01** | **mov,**  **dx,010C** | **0900** | **0000** | **0000** | **0000** | **FFFE** | **489D** | **489D** | **489D** | **0102** | **0102** |
| **3** | **CD21** | **int,**  **21** | **0900** | **0000** | **0000** | **010C** | **FFFE** | **489D** | **489D** | **489D** | **0105** | **0105** |
| **4** | **BA004C** | **mov,**  **ax,4C00** | **0900** | **0000** | **0000** | **010C** | **FFFE** | **489D** | **489D** | **489D** | **0107** | **0107** |
| **5** | **CD21** | **int,**  **21** | **4C00** | **0000** | **0000** | **010C** | **FFFE** | **489D** | **489D** | **489D** | **010A** | **010A** |

Текст командного файла

tasm Z3

tlink /t Z3

Z3

td Z3

**Контрольные вопросы**

1.Запустите под управлением отладчика программу из примера 1. Как располагаются сегменты программы в памяти? Сопоставьте полученные данные с соответствующими размерами сегментов из листинга трансляции.

**Ответ:** 48AD0+100=48BD; 48BD0+001A=48BEA

**SS=48AD0; DS= 48BD0; CS=48BEA**

2.Что такое сегментный (базовый) адрес?

**Ответ:** Чтобы получить 20-битовый адрес, дополнительные 4 бита адресной информации извлекаются из сегментных регистров. Сами сегментные регистры имеют размер в 16 разрядов, а содержащиеся в этих регистрах ( CS , DS , SS или ES ) 16-битовые значения называются **базовым адресом сегмента**. Микропроцессор объединяет 16-битовый исполнительный адрес и 16-битовый базовый адрес следующим образом: он расширяет содержимое сегментного регистра (базовый адрес) 4 нулевыми битами (в младших разрядах), делая его 20-битовым (полный адрес сегмента) и прибавляет смещение (исполнительный адрес).

3.Почему перед началом выполнения программы из примера 1 содержимое регистра DS в точности на 10h меньше содержимого регистра SS?

**Ответ:** Адрес начала сегмента всегда кратен 16-ти (каждый сегмент должен начинаться на границе параграфа).

4.Почему в программе из примера 1 первая команда mov AX,data, начинающаяся с байта 0000 сегмента команд, занимает 3 байта? Какая еще команда занимает 3 байта?

**Ответ:** Так как здесь используется прямая адресация. Ещё 3 байта занимает команда mov dx, 0000.

5.Чему равен сегментный адрес сегмента команд программы из примера 2? Почему?

**Ответ:** 48AD,так как используется модель памяти small.

6.Сравните содержимое регистра SP в программах из примера 2 и примера 3. Объясните, как получены эти значения.

**Ответ:** Во 2-м примере SP = 0100, так как используется модель памяти small и стек располагается в первых 256 байтах сегмента памяти, а в 3-м – FFFE, так как используется модель памяти tiny и сегменты кода, данных и стека находятся в одном сегменте размером 64 Кб**.**

7.Какие операторы называют директивами ассемблера? Приведите примеры директив.

**Ответ:**  Директивы – это параметры, указывающие программе ассемблеру, каким образом следует объединять инструкции для создания модуля, который станет работающей программой. Примеры: ASSUME, SEGMENT.

8.Зачем в последнем предложении end указывают метку, помечающую первую команду программы?

**Ответ:**  Для того, чтобы программа ассемблер поняла какая из подпрограмм закончена.

9.Как числа размером в слово хранятся в памяти и как они заносятся в 2-ух байтовые регистры?

**Ответ:** Они хранятся в памяти как шестнадцатеричное число и записываются справа налево. При занесении их в 2-х байтовые регистры байты меняются местами.

10.Как инициализируются в программе выводимые на экран текстовые строки?

**Ответ:** Сначала строка заносится в память также как и обычное число, а затем строка выводится на экран.

11.Что нужно сделать, чтобы обратиться к DOS для вывода строки на экран? Как DOS определит, где строка закончилась?

**Ответ:** Использовать оператор int 21h. В конец строки всегда добавляется символ $.

12.Программы, которые должны исполняться как .EXE и .COM, имеют существенные различия по:

* размеру,
* сегментной структуре,
* механизму инициализации.

**Ответ:** COM программы занимают меньше места в памяти и быстрее загружаются. В COM программах префикс программного сегмента, код программы, инициализированные данные и стек располагаются в одном сегменте. При запуске такой программы в начале строится PSP, занимающий 256 байт, далее располагается содержимое программы. Указатель стека устанавливается на конец сегмента.

EXE программы содержат сегменты кода, данных и стека. EXE файл загружается, начиная с адреса PSP:0100h. В процессе загрузки считывается информация EXE-заголовка в начале файла, при помощи которого загрузчик настраивает ссылки на сегменты в загруженном модуле, так как программа загружается в произвольный сегмент. После настройки ссылок управление передается загрузочному модулю к адресу CS:IP, извлеченному из заголовка EXE.