МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

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

Кафедра дискретной математики и информатики

ЛАБОРАТОРНАЯ РАБОТА №2

студента 2 курса 221 группы

направления 09.03.01 Информатика и вычислительная техника

факультета компьютерных наук и информационных технологий

Карасева Вадима Дмитриевича

Саратов 2025

**Текст задания**

Сначала программа должна печатать фамилию, имя и номер группы студента и переходить на новую строку. Затем аналогично рассмотренному упражнению выполните следующие задания:

**Задание 2.1.** Первая цифра задана в АХ, вторая цифра задана в ВХ. Написать программу, которая выводит в одну строку первую цифру, пробел, вторую цифру.

**Задание 2.2.** Первая цифра задана в АХ, вторая цифра задана в ВХ. Написать программу, которая выводит в одну строку первую цифру (АХ), пробел, вторую цифру (ВХ). Далее совершает обмен значений регистров АХ и ВХ и снова в новой строке на экране выводит в одну строку первую цифру (АХ), пробел, вторую цифру (ВХ). Обмен совершить без использования дополнительной памяти, регистров. Структура программы должна обязательно содержать одну или более вспомогательных процедур.

Выполнение заданий осуществлялось с помощью .COM файлов, так как это наиболее желаемый вариант для «маленьких» программ.

**Задание 2.1.**

.model tiny ;Модель памяти TINY, в которой код, данные и стек

;размещаются в одном и том же сегменте размером до 64Кб

.code ;Начало сегмента кода

org 100h ;Устанавливает значение программного счетчика в 100h

;Начало необходимое для COM-программы,

;которая загружается в память с адреса PSP:100h

start:

;===== Main Code =====

mov AH,09h

mov DX,offset Info

int 21h ;вывод ФИ + номер группы

call first ;вызов процедуры first

call space ;вызов процедуры space

call second ;вызов процедуры second

mov AX,4C00h

int 21h ;заканчиваем

;===== Procedures =====

first Proc NEAR ;процедура для вывода первой цифры

mov AL, 3h

add AL, 30h ;согласно таблице ASCII

mov DL, AL ;для вывода

mov AH, 02h ;команда вывода символа

int 21h ;DOS

Ret ;возврат вызываемой процедуры

first endp ;конец процедуры

space Proc NEAR ;процедура для вывода пробела

mov BL, 0h ;по таблице, 0h - пробел

mov DL, BL ;для вывода

mov AH, 02h ;команда вывода символа

int 21h ;DOS

Ret ;возврат выдаваемой процедуры

space endp ;конец процедуры

second Proc NEAR ;процедура для вывода второй цифры

mov BL, 4h

add BL, 30h ;согласно таблице ASCII

mov DL, BL

mov AH, 02h ;команда вывода символа

int 21h ;DOS

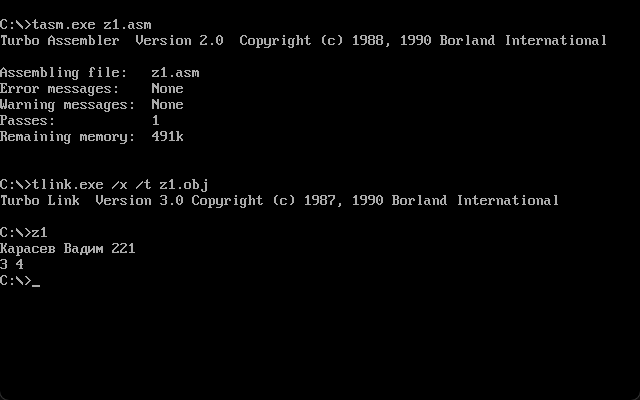
Ret ;возврат вызываемой процедуры

second endp ;конец процедуры

;===== Data =====

Info db 'Карасев Вадим 221', 13, 10, '$' ;строка с переходом на новую

end start



|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Шаг | Машинный код | Команда | Регистры | | | | | | | | | Флаги | |
| AX | BX | CX | DX | SP | DS | SS | CS | IP | CZSOPAID |
|  |  |  | 0000 | 0000 | 0000 | 0000 | FFFE | 489D | 489D | 489D | 0100 | 00000010 |
| 1 | BA3501 | MOV DX, 0135 | 0900 | 0000 | 0000 | 0000 | FFFE | 489D | 489D | 489D | 0102 | 00000010 |
| 2 | CD21 | INT 21 | 0900 | 0000 | 0000 | 0135 | FFFE | 489D | 489D | 489D | 0105 | 00000010 |
| 3 (вход в first) | E80B00 | CALL 0115 | 0900 | 0000 | 0000 | 0135 | FFFE | 489D | 489D | 489D | 0107 | 00000010 |
| 3.1 | B003 | MOV AL, 03 | 0900 | 0000 | 0000 | 0135 | FFFC | 489D | 489D | 489D | 0115 | 00000010 |
| 3.2 | 0430 | ADD AL, 30 | 0903 | 0000 | 0000 | 0135 | FFFC | 489D | 489D | 489D | 0117 | 00000010 |
| 3.3 | 8AD0 | MOV DL, AL | 0933 | 0000 | 0000 | 0135 | FFFC | 489D | 489D | 489D | 0119 | 00001010 |
| 3.4 | B402 | MOV AH, 02 | 0933 | 0000 | 0000 | 0133 | FFFC | 489D | 489D | 489D | 011B | 00001010 |
| 3.5 | CD21 | INT 21 | 0233 | 0000 | 0000 | 0133 | FFFC | 489D | 489D | 489D | 011D | 00001010 |
| 3.6 | C3 | RET | 0233 | 0000 | 0000 | 0133 | FFFC | 489D | 489D | 489D | 011F | 00001010 |
| 4 (вход в space) | E81300 | CALL 0120 | 0233 | 0000 | 0000 | 0133 | FFFE | 489D | 489D | 489D | 010A | 00001010 |
| 5 (вход в second) | E81900 | CALL 0129 | 0200 | 0000 | 0000 | 0100 | FFFE | 489D | 489D | 489D | 010D | 00001010 |
| 6 | B8004C | MOV AX, 4C00 | 0234 | 0034 | 0000 | 0134 | FFFE | 489D | 489D | 489D | 0110 | 00000010 |
| 7 | CD21 | INT 21 | 4C00 | 0034 | 0000 | 0134 | FFFE | 489D | 489D | 489D | 0113 | 00000010 |

**Задание 2.2.**

.model tiny ;модель памяти tiny

;(под всю программу выделяется 1 сегмент памяти)

.code

org 100h ;значение программного счётчика в 100h

start:

; ======Основной код======

mov AX, 3 ;сразу записываем наши цифры в регистры

mov BX, 4

mov AH, 09h

mov DX, offset Info

int 21h ;вывод строки

call print\_digits ;вызов процедуры print\_digits

XCHG AX, BX ;меняем значения AX и BX местами

call new\_line ;вызов процедуры new\_line

call print\_digits ;вызов процедуры print\_digits

mov AX, 4C00h

int 21h ;заканчиваем

; ======Процедуры======

proc print\_digits NEAR ;процедура для вывода двух цифр через

;пробел

; ------Первая цифра------

push AX ;пушим в стек

push BX ;чтобы сохранить их старые значения

add AL, 30h ;согласно таблице ASCII

mov DL, AL ;для вывода

mov AH, 02h ;команда вывода символа

int 21h ;DOS

call space ;вызов процедуры space

; ------Вторая цифра------

add BL, 30h ;согласно таблице ASCII

mov DL, BL ;для вывода

mov AH, 02h ;команда вывода символа

int 21h ;DOS

pop BX ;возвращаем значения

pop AX ;в "норму"

Ret ;возврат вызываемой процедуры

print\_digits endp ;конец процедуры

Proc new\_line NEAR ;процедура для перехода на новую

;строку

; ------Новая строка------

push AX ;пушим в стек

push BX ;чтобы сохранить их старые значения

mov AH, 02h ;команда вывода символа

mov DL, 13 ;для вывода

int 21h ;DOS

mov DL, 10 ;для перехода на новую строку нужно

;использовать 13, 10 или 0Dh, 0Ah

int 21h

pop BX ;возвращаем значения

pop AX ;в "норму"

Ret ;возврат вызываемой процедуры

new\_line endp ;конец процедуры

Proc space NEAR ;процедура для вывода пробела

; ------Пробел------

push AX ;пушим в стек

push BX ;чтобы сохранить их старые значения

mov BL, 0h ;это пробел (или пустой символ,

mov DL, BL ;но работает как пробел)

mov AH, 02h ;команда вывода символа

int 21h ;DOS

pop BX ;возвращаем значения

pop AX ;в "норму"

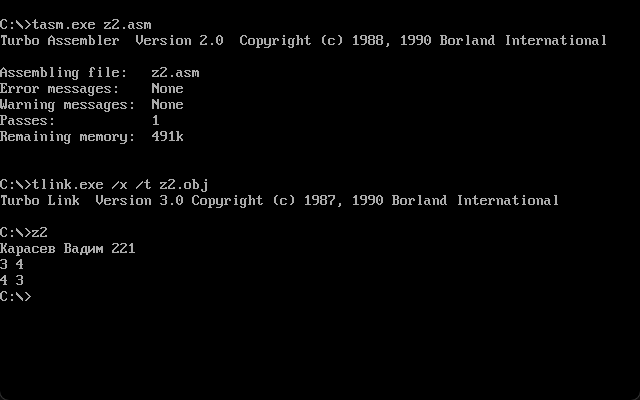
Ret ;возврат вызываемой процедуры

space endp ;конец процедуры

; ======Данные======

Info db 'Карасев Вадим 221', 13, 10, '$' ;строка с переходом на новую

end start



|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Шаг | Машинный код | Команда | Регистры | | | | | | | | | Флаги | |
| AX | BX | CX | DX | SP | DS | SS | CS | IP | CZSOPAID |
|  |  |  | 0000 | 0000 | 0000 | 0000 | FFFE | 489D | 489D | 489D | 0100 | 00000010 |
| 1 | BB0400 | MOV AX, 0004 | 0003 | 0000 | 0000 | 0000 | FFFE | 489D | 489D | 489D | 0103 | 00000010 |
| 2 | B409 | MOV AH, 09 | 0003 | 0004 | 0000 | 0000 | FFFE | 489D | 489D | 489D | 0106 | 00000010 |
| 3 | BA5101 | MOV DX, 0151 | 0903 | 0004 | 0000 | 0000 | FFFE | 489D | 489D | 489D | 0108 | 00000010 |
| 4 | CD21 | INT 21 | 0903 | 0004 | 0000 | 0151 | FFFE | 489D | 489D | 489D | 010B | 00000010 |
| 5 (процедура print\_digits) | E80C00 | CALL 011C | 0903 | 0004 | 0000 | 0151 | FFFE | 489D | 489D | 489D | 010D | 00000010 |
| 5.1 | 50 | PUSH AX | 0903 | 0004 | 0000 | 0151 | FFFC | 489D | 489D | 489D | 011C | 00000010 |
| 5.2 | 53 | PUSH BX | 0903 | 0004 | 0000 | 0151 | FFFA | 489D | 489D | 489D | 011D | 00000010 |
| 5.3 | 0430 | ADD AL, 30 | 0903 | 0004 | 0000 | 0151 | FFF8 | 489D | 489D | 489D | 011E | 00000010 |
| 5.4 | 8AD0 | MOV DL, AL | 0933 | 0004 | 0000 | 0151 | FFF8 | 489D | 489D | 489D | 0120 | 00001010 |
| 5.5 | B402 | MOV AH, 02 | 0933 | 0004 | 0000 | 0133 | FFF8 | 489D | 489D | 489D | 0122 | 00001010 |
| 5.6 | CD21 | INT 21 | 0233 | 0004 | 0000 | 0133 | FFF8 | 489D | 489D | 489D | 0124 | 00001010 |
| 5.7 (процедура space) | E81B00 | CALL 0144 | 0233 | 0004 | 0000 | 0133 | FFF8 | 489D | 489D | 489D | 0126 | 00001010 |
| 5.7.1 | 50 | PUSH AX | 0233 | 0004 | 0000 | 0133 | FFF6 | 489D | 489D | 489D | 0144 | 00001010 |
| 5.7.2 | 53 | PUSH BX | 0233 | 0004 | 0000 | 0133 | FFF4 | 489D | 489D | 489D | 0145 | 00001010 |
| 5.7.3 | B300 | MOV BL, 00 | 0233 | 0004 | 0000 | 0133 | FFF2 | 489D | 489D | 489D | 0146 | 00001010 |
| 5.7.4 | 8AD3 | MOV DL, BL | 0233 | 0000 | 0000 | 0133 | FFF2 | 489D | 489D | 489D | 0148 | 00001010 |
| 5.7.5 | B402 | MOV AH, 02 | 0233 | 0000 | 0000 | 0100 | FFF2 | 489D | 489D | 489D | 014A | 00001010 |
| 5.7.6 | CD21 | INT 21 | 0233 | 0000 | 0000 | 0100 | FFF2 | 489D | 489D | 489D | 014C | 00001010 |
| 5.7.7 | 5B | POP BX | 0200 | 0000 | 0000 | 0100 | FFF2 | 489D | 489D | 489D | 014E | 00001010 |
| 5.7.8 | 58 | POP AX | 0200 | 0004 | 0000 | 0100 | FFF4 | 489D | 489D | 489D | 014F | 00001010 |
| 5.7.9 | C3 | RET | 0233 | 0004 | 0000 | 0100 | FFF6 | 489D | 489D | 489D | 0150 | 00001010 |
| 5.8 | 80C330 | ADD BL, 30 | 0233 | 0004 | 0000 | 0100 | FFF8 | 489D | 489D | 489D | 0129 | 00001010 |
| 5.9 | 8AD3 | MOV DL, BL | 0233 | 0034 | 0000 | 0100 | FFF8 | 489D | 489D | 489D | 012C | 00000010 |
| 5.10 | B402 | MOV AH, 02 | 0233 | 0034 | 0000 | 0134 | FFF8 | 489D | 489D | 489D | 012E | 00000010 |
| 5.11 | CD21 | INT 21 | 0233 | 0034 | 0000 | 0134 | FFF8 | 489D | 489D | 489D | 0130 | 00000010 |
| 5.12 | 5B | POP BX | 0234 | 0034 | 0000 | 0134 | FFF8 | 489D | 489D | 489D | 0132 | 00000010 |
| 5.13 | 58 | POP AX | 0234 | 0004 | 0000 | 0134 | FFFA | 489D | 489D | 489D | 0133 | 00000010 |
| 5.14 | C3 | RET | 0903 | 0004 | 0000 | 0134 | FFFC | 489D | 489D | 489D | 0134 | 00000010 |
| 6 | 93 | XCHG BX, AX | 0903 | 0004 | 0000 | 0134 | FFFE | 489D | 489D | 489D | 0110 | 00000010 |
| 7 (процедура new\_line) | E82100 | CALL 0135 | 0004 | 0903 | 0000 | 0134 | FFFE | 489D | 489D | 489D | 0111 | 00000010 |
| 8 (процедура print\_digits) | E80500 | CALL 011C | 0004 | 0903 | 0000 | 010A | FFFE | 489D | 489D | 489D | 0114 | 00000010 |
| 9 | B8004C | MOV AX, 4C00 | 0004 | 0903 | 0000 | 0133 | FFFE | 489D | 489D | 489D | 0117 | 00001010 |
| 10 | CD21 | INT 21 | 4C00 | 0903 | 0000 | 0133 | FFFE | 489D | 489D | 489D | 011A | 00001010 |

**Контрольные вопросы**

**В какой регистр надо поместить код выводимого символа? Какой код Dos-функции используется для вывода отдельного символа на экран?**

Код выводимого символа нужно поместить в регистр DL, при этом код DOS-функции для вывода символа - int 02h.

**Какая операция позволяет получить для цифры её код в кодовой таблице?**

Получение кода в кодовой таблице ASCII для цифры можно, прибавив к числу 30h.

**Объясните назначение процедуры. Как определяются начало и конец процедуры?**

Процедура даёт возможность писать более объёмные программы, и их можно скопировать в другую программу, чтобы избежать переписывание кода.

Начало процедуры:

НАЗВАНИЕПРОЦЕНДУРЫ Proc ТИП

где ТИП = FAR или NEAR (по умолчанию)

Конец процедры:

ret

НАЗВАНИЕПРОЦЕДУРЫ endp

**Ваша программа состоит из главной процедуры и процедур-подпрограмм. Каким может быть взаимное расположение главной процедуры и подпрограмм?**

В общем случае, подпрограммы можно размещать где угодно, однако они должны выполняться только тогда, когда к ним идёт обращение (call). Самыми распространенными вариантами расположения являются:

— посреди программы в сегменте кода;

— В конце, после завершения основной программы;

— В начале, до точки входа в программу.

**Как процессор использует стек при работе с любой процедурой?**

Первый байт PSP содержит код команды INT 20h, которая используется для выхода из программы и возврата в DOS. Поэтому для выполнения возврата в операционную систему с помощью команды процедур RETF, в начале процедуры MAIN необходимо поместить в стек адрес PSP. RETF передаст управление на нулевой адрес текущего сегмента, т.е. на первый байт PSP.

**С помощью какой команды вызывается процедура? Как меняется значение регистра SP после вызова процедуры? Приведите пример из вашей таблицы трассировки.**

Процедура вызывается с помощью:

CALL ИМЯПРОЦЕДУРЫ

После чего в стек помещается адрес вызова (с которого продолжится программа после выполнения процедуры). В качестве примера рассмотрим таблицу трассировки для 2-й таблицы: начиная с шага 5, программа «входит» в процедуру вывода цифр. На последующих шагах видно, что значение SP меняется с FFFE на FFFC. Это говорит о том, что была произведена запись в стек точки, с которой следует продолжать выполнение кода после выхода из процедуры.

**После какой команды процедуры из стека извлекается адрес возврата?**

После завершения работы процедуры, с помощью команды RET происходит извлечение адреса из стека, после чего управление передаётся на этот адрес, т.е. продолжается выполнение программы с момента, следующим за точкой вызова процедуры.