БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Кафедра программного обеспечения информационных технологий

Факультет ИНО

Специальность ПОИТ

Контрольная работа

по дисциплине «Языки программирования»

часть 1

Вариант № 15

Выполнила: Н. Г. Карпеко

Обучение по сертификатам

Минск 2020

***Тема работы.*  *ПРОЦЕДУРЫ.***

Для рассчитанного варианта индивидуального задания написать на языке Ассемблер и отладить три программы, содержащие процедуру. Программы различаются между собой способом передачи параметров в процедуру и типом процедуры:

**Программа 1.** Передача параметров через регистр. Тип процедуры – дальний.

**Программа 2**.Передача параметров через глобальные переменные. Тип процедуры – дальний.

**Программа 3**. Передача параметров через стек. Тип процедуры – ближний.

**Задание:** Ввести три 16-битовых целых числа Х, Y и Z. Вычислить результат циклического сдвига влево значения выражения 2\*X2 + Y – 5\*Z на количество битов, равное числу Х.

### Описание программы:

Файл contr.asm содержит исходный код программы. Компиляция, сборка и запуск программы выполнены в среде DOSBox Ver. 0.74-3.

В программе имеются пять процедур:

1. Процедура **RegisterProc** — передача параметров (X, Y, Z) происходит через регистры ax, bx, cx. Тип процедуры — дальний.
2. Процедура **GlobProc**— передача параметров (X, Y, Z) происходит через глобальные переменные GlobX, GlobY, GlobZ. Тип процедуры — дальний.
3. Процедура **StackProc** — передача параметров (X, Y, Z) осуществляется через стек. Тип процедуры — ближний.

В данных процедурах происходит вычисление заданного выражения и сохранение результата в переменной Rzlt. Далее результат заносится в регистр данных dx для вывода на экран.

1. Процедура **ShowRzlt** – выводит результат на экран, который предварительно загружается в регистр данных dx, к его содержимому добавляется смещение 30h для вывода на экран числа.
2. Процедура **NewLine** — осуществляет перевод на новую строку.

Листинг программы:

.model small ; модель памяти small

.386 ; режим работы микропроцессора

.stack 256 ; сегмент стека (занимает 256 байтов)

.data ; сегмент данных

X equ 3 ; начальное значение числа X

Y equ 14 ; начальное значение числа Y

Z equ 6 ; начальное значение числа Z

Rzlt dw 0 ; переменная для хранения результата

GlobX dw ? ; глобальная переменная X

GlobY dw ? ; глобальная переменная Y

GlobZ dw ? ; глобальная переменная Z

BR db 10, 13, '$' ; переменная для перевода строки

;---------------------------------------------------------------------------------------------------

.code ; сегмент кода

main: ; метка начала программы

mov ax, @data ; занесение адреса сегмента данных в регистр ax

mov ds, ax ; копирование содержимого регистра ax в регистр ds

**xor** ax, ax ; обнуление регистра ax

mov ax, X ; занесение в регистры ax, bx, cx значений

mov bx, Y ; чисел X, Y, Z соответственно

mov cx, Z

call far ptr RegisterProc ; дальний вызов процедуры передачи параметров

; (X, Y, Z) через регистры

call ShowRzlt ; вызов процедуры вывода результата вычисления

call NewLine ; вызов процедуры перевода на новую строку

mov GlobX, X ; занесение в глобальные переменные GlobX, GlobY,

mov GlobY, Y ; GlobZ значений чисел X, Y, Z соответственно

mov GlobZ, Z

call far ptr GlobProc ; дальний вызов процедуры передачи параметров

; через глобальные переменные

call ShowRzlt ; вызов процедуры вывода результата вычисления

call NewLine ; вызов процедуры перевода на новую строку

push X ; занесение в стек значений чисел X, Y, Z

push Y

push Z

call near ptr StackProc ; ближний вызов процедуры передачи параметров

; через стек

call ShowRzlt ; вызов процедуры вывода результата вычисления

mov ah, 4ch ; завершение работы программы, возврат в DOS

int 21h ; вызов прерывания

; -------- процедура передачи параметров (X, Y, Z) RegisterProc через регистры

; ------------- тип процедуры – дальний --------------------------------------------------

RegisterProc proc far

mov Rzlt, 0 ; обнуление переменной результата

imul ax, ax ; умножение содержимого ax на это же значение (X\*X)

imul ax, 2 ; умножение содержимого ax на число 2 (2\*X^2)

add ax, bx ; добавление в ax содержимого bx – Y (2\*X^2+Y)

mov Rzlt, ax ; занесение в Rzlt промежуточного результата (2\*X^2+Y)

imul cx, 5 ; умножение содержимого cx на число 5 (Z\*5) (в cx – Z)

sub Rzlt, cx ; вычитание из Rzlt содержимого cx, в переменной Rzlt

; хранится результат вычисления 2\*X^2+Y-5\*Z

rol Rzlt, offset X ; циклический сдвиг влево Rzlt на X битов

mov dx, Rzlt ; занесение в dx значения переменной Rzlt

**xor** ax, ax ; обнуление используемых в процедуре

**xor** bx, bx ; регистров (ax, bx, cx)

**xor** cx, cx

retf ; дальний возврат из процедуры

RegisterProc endp ; --- конец процедуры RegisterProc

; --- процедура передачи параметров UseGlobals через глобальные переменные

; ------------- тип процедуры – дальний ---------------------------------------------------

GlobProc proc far

mov Rzlt, 0 ; обнуление переменной результата

mov ax, GlobX ; занесение в ax значения GlobX

imul ax, ax ; умножение содержимого ax на это же значение (X\*X)

imul ax, 2 ; умножение содержимого ax на число 2 (2\*X^2)

add ax, GlobY ; добавление в ax значения GlobY (2\*X^2+Y)

mov Rzlt, ax ; занесение в Rzlt промежуточного результата (2\*X^2+Y)

mov ax, GlobZ ; занесение в ax значения GlobZ

imul ax, 5 ; умножение содержимого ax на число 5 (Z\*5)

sub Rzlt, ax ; вычитание из Rzlt содержимого ax, в переменной Rzlt

; хранится результат вычисления 2\*X^2+Y-5\*Z

rol Rzlt, offset X ; циклический сдвиг влево Rzlt на X битов

mov dx, Rzlt ; занесение в dx значения переменной Rzlt

**xor** ax, ax ; обнуление используемого в процедуре регистра ax

retf ; дальний возврат из процедуры

GlobProc endp ; --- конец процедуры UseGlobals

; ---------- процедура передачи параметров StackProc через стек -------------------

; ---------- тип процедуры – ближний -----------------------------------------------------

StackProc proc near

mov Rzlt, 0 ; обнуление переменной результата

mov bp, sp ; занесение в регистр bp указателя вершины стека sp

; для косвенной адресации

mov ax, [bp+6] ; занесение в ax переданного через стек числа X

imul ax, ax ; умножение содержимого ax на это же значение (X\*X)

imul ax, 2 ; умножение содержимого ax на число 2 (2\*X^2)

add ax, [bp+4] ; добавление в ax значения переданного через стек

; числа Y (2\*X^2+Y)

mov Rzlt, ax ; занесение в Rzlt промежуточного результата (2\*X^2+Y)

mov ax, [bp+2] ; занесение в ax переданного через стек числа Z

imul ax, 5 ; умножение содержимого ax на число 5 (Z\*5)

sub Rzlt, ax ; вычитание из Rzlt содержимого ax, в переменной Rzlt

; хранится результат выражения 2\*X^2+Y-5\*Z

rol Rzlt, offset X ; циклический сдвиг влево Rzlt на X бит

mov dx, Rzlt ; занесение в dx значения переменной Rzlt

**xor** ax, ax ; обнуление используемого в процедуре регистра ax

retn 6 ; ближний возврат из процедуры, удаление из стека

; переданных параметров

StackProc endp ; --- конец процедуры StackProc

; ------- процедура вывода результата на экран ShowRzlt -----------------------

ShowRzlt proc

add dx, 30h ; занесение в dx числа 30 для вывода числа

mov ah, 02h ; вывод на консоль имеет свой адрес (02h), на него с

; помощью mov отправляется байт для отображения

int 21h ; вызов прерывания 21h, вывод результата (Rzlt)

**xor** dx, dx ; обнуление используемых в процедуре регистров (dx, ah)

**xor** ah, ah

ret ; возврат из процедуры

ShowRzlt endp ; --- конец процедуры DisplayRzlt

; ------ процедура перевода на новую строку NewLine -------------------

NewLine proc

mov dl, offset BR ; занесение в dl смещения переменной перевода строки

mov ah, 09h ; в регистре ah хранят функцию вывода строки (09h)

int 21h ; вызов прерывания 21h, перевод строки

**xor** dl, dl ; обнуление используемых в процедуре регистров (dl, ah)

**xor** ah, ah

ret ; возврат из процедуры

NewLine endp ; --- конец процедуры NewLine

**end** main ; конец программы с точкой входа main.

Результаты запуска программы в DOSBox:

Тестовый набор данных:

X = 2, Y = 13, Z = 4.

Подставим данные значение в заданное выражение:

2\*X\*X + Y – 5\*Z = 2\*2^2 + 13 – 5\*4 = 1

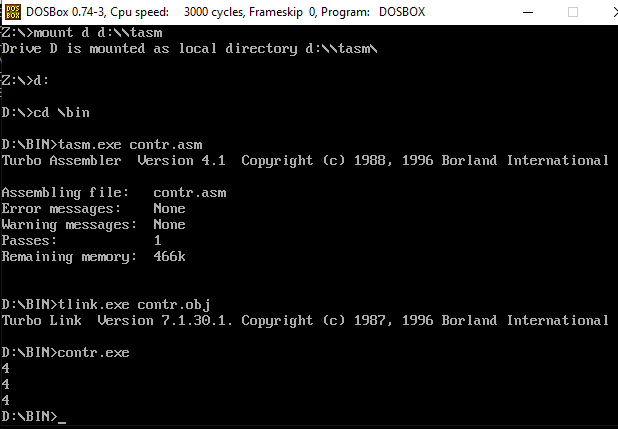
Для вычисления результата циклического сдвига 16-битового числа 1 на 2 бита (число X), представим его в двоичной системе счисления:

1d = 0000 0000 0000 0001b

Выполним операцию циклического сдвига данного двоичного числа на 2 бита, получим число:

0000 0000 0000 0100b = 4d

Аналогичный результат получен при запуске программы:



В данном результате вычисления (скриншот):

в первой строке — результат в процедуре RegisterProc,

во второй строке — результат в процедуре GlobProc,

в третьей строке — результат в процедуре StackProc.

Ещё один тестовый набор данных (для контроля):

X = 1, Y = 16, Z = 3.

2\*X\*X + Y – 5\*Z = 2\*1^2 + 16 – 5\*3 = 3

3d = 0000 0000 0000 0011b

После циклического сдвига на 1 бит, получим число:

0000 0000 0000 0110b = 6d

Аналогичный результат – при запуске программы:

