Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ  
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра электронных вычислительных машин

Лабораторная работа № 3

по дисциплине ПнаЯА

Вариант 9

Выполнил студент гр. 150502: Снитко Д.А.

Проверил: Туровец Н.О.

Минск 2022

Теоретические сведения:

1. Арифметические операции над целыми числами.
2. Логические побитовые операции над целыми числами.
3. Процедуры.

Задание:

Ввести массив целых чисел размерностью 30 элементов. Найти отношение разности максимального и минимального значений элементов к максимальному значению.

Код:

.model small

.stack 100h

.data

div\_0\_str db 0Ah, 0Dh, 'division by 0 error$'

ArrayLength db ?

InputArrayLengthMsgStr db 0Dh,'Input the length of array(1<length<31): $'

tmp db ?

InputMsgStr db 0Dh,'Arr['

CurEl1 db 48

CurEl2 db 48

InputMsgStr\_2 db '] - $'

TaskMsg db 0Ah,0Dh,'Result((MaxEl-MinEl)/MaxEl) = $'

ErrorInputMsgStr db 0Ah, 0Dh,'Incorrect input' ,0Ah,'$'

minus\_flag db 0

minus\_flag\_max db 0

minus\_flag\_min db 0

accuracy dw 6 ;кол-во цифр после запятой

num\_10 dw 10

b\_num\_10 db 10

MaxArrayLength equ 30

Array dw MaxArrayLength dup (0)

sk db 0Ah, 0Dh, '$'

MinEl dw ? ;минимальный элемент

MaxEl dw ? ;максимальный элемент

.code

mov ax, @data

mov ds, ax

;---------------------------------------- ввод длины массива

input\_array\_length proc

call arr\_length\_mess

call inp\_length

endp

arr\_length\_mess proc ;Вывод ArrayMsgString

xor bx, bx ;bx = 0

mov ah, 09h

lea dx, InputArrayLengthMsgStr

int 21h

ret

endp

inp\_length proc ;Ввод ArrayLength (al = asci код length)

mov ah,01h

int 21h

cmp al, 0Dh

je push\_to\_arr\_len ;если al = 0Dh

jmp isNum ;если al != 0Dh

next: ;в bx хранится число

sub al,'0' ;вычитаем 30h от ASCII кода цифры и получаем число (Например в al было '3', а станет 3)

xor ah,ah ;ah = 0(расширяем до слова)

mov cx, ax ;cx = ax(в cx цифра числа)

xor ax,ax ;ax = 0

mov ax, bx ;ax = bx(ax становится числом, cx новая цифра)

mul num\_10 ;умножаем ax на 10 для формирования числа

add ax,cx ;ax = ax\*10+cx

mov bx,ax ;bx число

jmp check\_arr\_length

endp

isNum proc ;проверяем является ли введенный символ символом числа(0 <= ASCII код цифры <= 9)

cmp al, '0'

jae nx ;если al >= 0

jmp errorArrLength

nx:

cmp al, '9'

jle next ;если al <= 9

jmp errorArrLength

endp

check\_arr\_length: ;проверка длинны массива

mov cx, MaxArrayLength

cmp bx, cx

jle inp\_length ;если bx(введенное число) <= 30

jmp errorArrLength

push\_to\_arr\_len: ;проверка длины массива(после введенного enter)

cmp bx, 1

jle errorArrLength ;если bx(введенное число) <=1

mov ArrayLength, bl ;записываем длину массива

mov tmp, bl

jmp inp\_array\_of\_shorts

;---------------------------------------- ввод массива

inp\_array\_of\_shorts proc

xor si, si ;si = 0

call inp\_num

endp

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

print\_arr\_index: ;вывод сообщения на экран

mov ah, 09h

lea dx, sk

int 21h

mov ah,09h;

lea dx,InputMsgStr

int 21h

add CurEl2, 1

cmp CurEl2, 58

je set\_counter

z:

jmp exe

set\_counter:

add CurEl1, 1

mov CurEl2, 48

jmp z

inp\_num proc

jmp print\_arr\_index

exe:

xor bx,bx ;bx = 0

xor dx,dx ;dx = 0

loop2: ;вводим число

mov ah,01h

int 21h

cmp al, 0dh ;если введен enter

je mini\_max ;

cmp al, '-' ;если число отрицательное

je minus\_check

jmp isNm

nxt: ;формируем число по схеме Горнера(как в next)

sub al,'0'

xor ah,ah

mov cx, ax

xor ax,ax

mov ax, bx

mul num\_10

jo error

add ax,cx ;число в bx

mov bx, ax

jmp overflow\_check

endp

overflow\_check: ;проверка на переполнение

cmp bx, 32768

ja error

jmp loop2 ;вводим следующую цифру числа

minus\_check proc ;установка -

test bx, bx

jnz error

call set\_minus\_flag

mov minus\_flag, 1

jmp loop2

endp

set\_minus\_flag proc ;установка -

push cx

mov cl, minus\_flag

test cl, cl

pop cx

jnz error

ret

endp

isNm proc ;является ли введенный символ цифрой

cmp al, '0'

jae nex

jmp error

nex:

cmp al, '9'

jle nxt

jmp error

endp

push\_to\_arr proc ;записываем число в массив

add si, 2

sub tmp, 1

cmp tmp, 0

ja inp\_num ; tmp > 0(вводим следующее число)

jmp print\_taskmsg

endp

;---------------------------------------- поиск минимального и максимального элементов

mini\_max proc

mov cl, minus\_flag

test cl, cl ;сбрасываем флаг OF(переполнение) CF(перенос), изменяется ZF

jz pos ;если 0 (ZF = 1)

neg bx ;меняем знак bx

er:

jns error

jmp e\_mm

pos:

cmp bx, 32768 ;если bx = 32768

je error

e\_mm:

cmp si,0

je setMinMaxEl ;Первое число становится мин и макс элементом

cmp bx, MinEl ;Если введенное число меньше минимального

jle setMinEl ;1<2

cmp bx, MaxEl ;Если введенное число больше максимального

jge setMaxEl ;1>2

call push\_to\_arr

endp

setMinMaxEl:

mov cl, minus\_flag

mov minus\_flag\_max, cl

mov minus\_flag\_min, cl

mov MinEl, bx

mov MaxEl, bx

mov minus\_flag,0

call push\_to\_arr

setMinEl: ;устанавливаем минимальный элемент

mov cl, minus\_flag

mov minus\_flag\_min, cl

mov MinEl, bx

mov minus\_flag,0

call push\_to\_arr

setMaxEl: ;устанавливаем максимальный элемент

mov cl, minus\_flag

mov minus\_flag\_max, cl

mov MaxEl, bx

mov minus\_flag,0

call push\_to\_arr

print\_taskmsg: ;Выводим сообщение TaskMsg

mov ah,09h

lea dx,TaskMsg

int 21h

jmp calculate

;---------------------------------------- вывод

push\_num\_to\_stack proc

push dx

push bx

mov bx,ax ;Делимое в bx

mov bp,sp

loop1: .

cmp ax,0

je skip\_actions

div b\_num\_10 ;

xor bx,bx

mov bl,ah

xor ah,ah

skip\_actions:

push bx

cmp al, 0

je print\_num

jmp loop1

print\_num:

loop3:

xor dx,dx

pop bx

add bx,'0'

mov ah,02h

mov dl,bl

int 21h

cmp bp,sp

jne loop3 ;если есть еще цифры

pop bx

pop dx

ret

endp

;---------------------------------------- (макс - мин)/макс

calculate:

mov ax,MaxEl ;ax = MaxEl

cmp ax,0

je err\_div\_0 ;если MaxEl = 0

sub ax,MinEl ;ax = MaxEl - MinEl

call pr\_div

jmp end

pr\_div proc ;Деление

push ax ;Делимое

push dx

push bx

mov bx,MaxEl ;Делитель в bx

mov dl,minus\_flag

cmp dl,minus\_flag\_max

unsing\_div:

div MaxEl ;(Max - Min) / Max (DX:AX/MaxEl в AX целая цасть, DX дробная часть)

call push\_num\_to\_stack

test dx,dx

jz pr\_div\_end

mov ax, MaxEl

call ost\_output

jmp pr\_div\_end

pr\_div\_ost\_unsign:

call ost\_output

jmp pr\_div\_end

pr\_div\_end:

pop bx

pop dx

pop ax

ret

endp

ost\_output proc ;Вывод дробной части

push ax

push dx

push cx

push dx

mov bx,ax

mov ah,2h

mov dl,'.'

int 21h

pop dx

mov cx,accuracy ;Кол-во цифр в дробной части

ost\_cycle:

mov ax, dx

mul num\_10

div bx

push dx

mov dx,ax

mov ah,2h

add dx,'0'

int 21h

sub dx,'0'

pop dx

cmp dx,dx

loopz ost\_cycle

ost\_end:

pop cx

pop dx

pop ax

ret

endp

;---------------------------------------- ошибки

errorArrLength: ;если длина массива неверна

mov ah, 09h

lea dx, ErrorInputMsgStr ;вывод ошибки

int 21h

jmp input\_array\_length ;повторный ввод длины

error:

mov ah, 09h

lea dx, ErrorInputMsgStr

int 21h

sub CurEl2, 1

mov minus\_flag, 0

jmp inp\_num

err\_div\_0: ;если делим 0

mov ah, 09h

lea dx,div\_0\_str

int 21h

jmp end

;---------------------------------------- выход

end:

mov ax, 4c00h

int 21h

Демонстрация работы:

