Міністерство освіти і науки України

Одеський національний політехнічний університет

Iнститут компьютерних систем

Кафедра компьютерних інтеллектуальних систем і мереж

Лабораторна робота №3

З дисципліни: “Архiтектура комп’ютерiв”

По темi: “Спрощений опис сегментів. Обчислення цілочисельних арифметичних виразів.”

Зробив

Ст. Баженов М.I.

Групи АМ-182

Перевiрили:

Шапорiн Р.О.

Одеса 2020

**Мета:** Ознайомитися з описом сегментів та арифметичними командами.

**Завдання:**

1. Вибрати свій варіант арифметичного виразу.



2. Написати модуль мовою Асемблера для обчислення значення виразу.

3. Написати на базовій алгоритмічній мові програму коректного вводу вихідних даних (з контролем

припустимого діапазону) і виводу отриманого результату.

4. Вмонтувати виклик модуля мовою Асемблера в програму на базовій алгоритмічній мові.

5. Зробити тестові перевірки, зробити аналіз результатів.



ввод чисел



результат



Перевірка при ділені на нуль. Замість результатату буде виводитись текст “You tried to divide on zero. Choose another A”

6. Занесіть до протоколу:

- складений арифметичний вираз;

- вихідні дані й отримані результати обчислень у шістнадцятирічній формі, а також їхнє місце розташування

в оперативній пам'яті і їхні еквіваленти в десятковій формі (для можливості перевірки обчислень);

- текст програми і її опис.

Код програми

format PE console

entry Start

include 'INCLUDE\win32a.inc'

section '.data' data readable writable

strA db 'Your A is: ', 0 ;text for P

strC db 'Your C is: ', 0 ;text for P

strD db 'Your D is: ', 0 ;text for P

resStr db 'Your answer is: %d', 0

resMod db '/%d' , 0

spaceStr db ' %d', 0

emptyStr db '%d',0

infinity db 'You tried to divide on zero. Choose another A', 0

point db ',', 0

C dd ?

D dd ?

A dd ?

RES dd ?

Fourth dd ?

Fifth dd ?

First dd ?

Second dd ?

Third dd ?

NULL = 0

section '.code' readable executable

Start:

push strA

call [printf]

push A

push spaceStr

call [scanf]

push strC

call [printf]

push C

push spaceStr

call [scanf]

;check adding numbers

push strD

call [printf]

push D

push spaceStr

call [scanf]

push [A]

push spaceStr

push [C]

push [D]

push spaceStr

mov ecx, [C]

imul ecx, 2

mov [First], ecx

push [First]

push resStr

mov ecx, [D]

mov [Second], ecx

push [Second]

push resStr

mov ecx, [First]

sub ecx, [Second]

add ecx, 23

mov [First], ecx

push [First]

push resStr

cmp [A],4

jne Go

push infinity

call [printf]

jmp exit

Go:

mov eax, [A]

mov ecx, 4

mov edx, 0

div ecx

mov [Fourth], eax

push [Fourth]

push resStr

mov ecx, [Fourth]

sub ecx, 1

mov [Fifth], ecx

push [Fifth]

push resStr

; call [printf]

mov eax, [First]

mov ecx, [Fifth]

mov edx, 0

cmp [Fifth],0

jne NotNullDiv1

push infinity

call [printf]

jmp exit

NotNullDiv1:

div ecx

mov [RES], edx

push eax

push resStr

call [printf]

push point

call [printf]

mov ebx, 0

lp:

mov eax, [RES]

mov ecx, [D]

imul eax, 10

mov edx, 0

div ecx

mov [RES], edx

push eax

push emptyStr

call [printf]

add ebx, 1

cmp ebx, 3

jne lp

jmp exit

jmp exit

exit:

call [getch]

push NULL

call [ExitProcess]

section '.idata' import data readable

library kernel, 'kernel32.dll',\

msvcrt, 'msvcrt.dll'

import kernel,\

ExitProcess, 'ExitProcess'

import msvcrt,\

printf, 'printf',\

scanf, 'scanf', \

getch, '\_getch'

**Висновок:** У цій лабораторній роботі я навчився сегментувати код програми та вікористовувати арифметичні команди на ЯП ASSEMBLER. Для більш зручного вводу та виводу інформації додатково навчився користуваватися можливостями бібліотеки “win32a.inc”, яка йде разом з програмою для кодування “Flat Assembler”