МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «РЯЗАНСКИЙ ГОСУДАРСТВЕННЫЙ РАДИОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ ИМЕНИ В.Ф. УТКИНА»

КАФЕДРА «ЭВМ»

Отчет по лабораторной работе №2

по дисциплине

«Машинно-зависимые языки программирования»

Выполнили:

Студенты группы 945

Комлева Екатерина

Соболева Елизавета

Горбачева Дарья

Проверили:

доц., к.т.н. Муратов Е.Р.

асс. каф. ЭВМ Тарасов А.С.

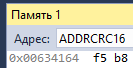
Рязань 2021

**Цель работы:** ознакомиться с размещением массивов данных в памяти и выполнением арифметических действий над разными типами данных.

**ПРАКТИЧЕСКАЯ ЧАСТЬ**

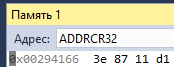
**Задание 1:** Рассчитайте контрольную сумму для первых 256 слов (word) прочитанных из файла w\_512.dat. Контрольная сумма рассчитывается по следующим образом: - суммируются все 256 слов. Результат берется по модулю 0x10000h, затем из 0 вычитается полученный результат. Полученное после вычитания значения будем считать контрольной суммой (CRC16).

Переменная CRC16 хранит контрольную сумму, а переменная ADDRCRC16 – адрес, по котором хранится контрольная сумма. При переходе по этому адресу получим контрольную сумму – B8F5h.

  
Рисунок 1 – Результат выполнения задания 1

**Задание 2:** Рассчитайте контрольную сумму для первых 256 двойных слов (dword) прочитанных из файла w\_512.dat. В этом случае сумма берется по модулю 0x100000000h, затем из 0 вычитается полученный результат. Полученное после вычитания значения будем считать контрольной суммой (CRC32).

Переменная CR32 хранит контрольную сумму, а переменная ADDRCR32 – адрес, по котором хранится контрольная сумма. При переходе по этому адресу получим контрольную сумму – D111873Eh.

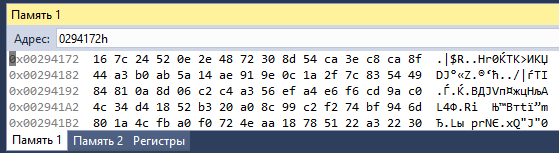
   
Рисунок 2 – Результат выполнения задания 2

**Задание 3:** Для 512 значений типа word, разбейте на 2 отдельных массива четные слова с нечетными.

В переменной EvenWordsC содержится количество чётных слов, а в OddWordsC – нечётных слов.

C:\Users\komle\AppData\Local\Temp\lu9348cgznox.tmp\lu9348cgznpi_tmp_4aa4a973662495af.png  
Рисунок 3 – Количество чётных слов

C:\Users\komle\AppData\Local\Temp\lu9348cgznox.tmp\lu9348cgznpi_tmp_72c453b6f5ec77bb.png  
Рисунок 4 – Количество нечётных слов

  
Рисунок 5 – Массив в памяти

**Задание 4:** Создайте массив в который сохраните все нечетные значения из первых 1024 элементов типа dword прочитанных из файла. Выведите количество полученных элементов.

Переменная Ex4\_counter содержит количество нечётных двойных слов, записанных в массив.

C:\Users\komle\AppData\Local\Temp\lu9348cgznox.tmp\lu9348cgznpi_tmp_9b58318972e2f3cf.png  
Рисунок 6 – Результат выполнения задания 4

**Задание 5:** Выведите последние два байта прочитанные из файла.

Результат выполнения подпрограммы для задания 5 представлен на рисунке 5. Последние 2 байта из прочитанного файла были загружены в регистр BX.

C:\Users\komle\AppData\Local\Temp\lu9348cgznox.tmp\lu9348cgznpi_tmp_797dde8aa56ea3f0.png  
Рисунок 7 – Результат выполнения задания 5

**Задание 6:** Выведите первый байт сдвинутый на бит влево и первое слово сдвинутое на 3 бита вправо.

Результат выполнения подпрограммы для задания 6 представлен на рисунке 6. Результат операций сдвига помещён в регистр CX.

C:\Users\komle\AppData\Local\Temp\lu9348cgznox.tmp\lu9348cgznpi_tmp_11b4082c1e3b26e2.png  
Рисунок 8 – Результат выполнения задания 6

**Вывод:** в данной лабораторной работе мы ознакомились с размещением массивов данных в памяти и выполнением арифметических действий над разными типами данных.

**ПРИЛОЖЕНИЕ**

.686P

.MODEL FLAT, STDCALL

.STACK 4096

option casemap:none

include c:/masm32/include/windows.inc

include c:/masm32/include/user32.inc

includelib c:/masm32/lib/user32.lib

include c:/masm32/include/kernel32.inc

includelib c:/masm32/lib/kernel32.lib

.DATA

FileName db "F:\w\_512.dat",0

BadText db "File dont open!",0

GoodText db "File is open!", 0

EvenWordsC dd 0

OddWordsC dd 0

Ex4\_counter dd 0

.DATA?

hFile HANDLE ?

hMemory DWORD ?

pMemory DWORD ?

dwBytesRead dd ?

dwFileSize dd ?

CRC16 dw ?

CR32 dd ?

ADDRCRC16 dd ?

ADDRCR32 dd ?

EvenWordsAcc dw 512 dup(?)

OddWordsAcc dw 512 dup(?)

Ex4\_Arr dd 1024 dup(?)

.CODE

START:

invoke CreateFile, addr FileName, GENERIC\_READ, 0, NULL, OPEN\_EXISTING, FILE\_ATTRIBUTE\_NORMAL, NULL

mov hFile, eax

cmp hFile, INVALID\_HANDLE\_VALUE

jz ErrorMsg

invoke GetFileSize, hFile, NULL

mov dwFileSize, eax

invoke GlobalAlloc, GMEM\_FIXED or GMEM\_ZEROINIT, dwFileSize

mov hMemory, eax

invoke GlobalLock, hMemory

mov pMemory, eax

invoke ReadFile, hFile, pMemory, 4096, addr dwBytesRead, NULL

or eax, eax

jz ErrorMsg

mov eax, pMemory

ex1:

mov ecx, 256

cycle1:

mov ebx, 256

sub ebx, ecx

sal ebx, 1

add eax, ebx

add dx, [eax]

sub eax, ebx

dec ecx

jz ex1\_end

jmp cycle1

ex1\_end:

xor dx, 1000h

mov CRC16, dx

mov ADDRCRC16, offset(CRC16)

ex2:

mov ecx, 256

cycle2:

mov ebx, 256

sub ebx, ecx

sal ebx, 2

add eax, ebx

add edx, [eax]

sub eax, ebx

dec ecx

jz ex2\_end

jmp cycle2

ex2\_end:

xor edx, 10000000h

mov CR32, edx

mov ADDRCR32, offset(CR32)

ex3:

mov ecx, 512

cycle3:

mov ebx, 512

sub ebx, ecx

sal ebx, 1

add eax, ebx

mov dx, [eax]

sub eax, ebx

mov bx, dx

sar bx, 1

jc c3

mov ebx, EvenWordsC

mov EvenWordsAcc[ebx\*2], dx

inc EvenWordsC

jmp c3End

c3:

mov ebx, OddWordsC

mov OddWordsAcc[ebx\*2], dx

inc OddWordsC

c3End:

dec ecx

jz ex3\_end

jmp cycle3

ex3\_end:

ex4:

mov ecx, 1024

cycle4:

mov ebx, 1024

sub ebx, ecx

sal ebx, 2

add eax, ebx

mov edx, [eax]

sub eax, ebx

mov ebx, edx

sar ebx, 1

jnc c4

mov ebx, Ex4\_counter

mov Ex4\_Arr[ebx\*2], edx

inc Ex4\_counter

c4:

dec ecx

jz ex4\_end

jmp cycle4

ex4\_end:

mov ebx, Ex4\_counter

ex5:

mov bx, [eax+4094]

ex6:

mov bl, [eax]

mov cx, [eax]

shl bl, 1

sar cx, 3

invoke GlobalUnlock, pMemory

invoke GlobalFree, hMemory

invoke CloseHandle, hFile

jmp End\_code

ErrorMsg:

invoke MessageBox, NULL, addr BadText, addr BadText, MB\_OK

invoke ExitProcess, 0

End\_code:

RET

END START