МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ»

# Кафедра электронных вычислительных машин

Отчет по предмету

Программирование на языке ассемблера

Лабораторная работа №7

**«Загрузка и выполнение программ. Работа с памятью»**

|  |  |
| --- | --- |
| **Выполнил:**  Студент группы 230501  Кочеров Р.С. | **Проверил:**  Туровец Н.О. |

Минск 2023

Цель работы: Ознакомиться с загрузкой и выполнением программ, рассмотреть работу с памятью..

**Текст программы:**

;convert to string

toString proc

pusha

xor di, di

lea di, tempString

mov ax, tempVariable

xor bx, bx

mov bx, di

xor cx, cx

mov cx, 256

setZeroString:

mov [di], byte ptr '$'

loop setZeroString

lea di, tempString

itoaLoop:

xor dx, dx

div base

add dl, '0'

mov [di], dl

inc di

cmp ax, 0

ja itoaLoop

dec di

xor si, si

mov si, bx

reverseMini:

xor dx, dx

xor cx, cx

mov dl, byte ptr [si]

mov cl, byte ptr [di]

mov [si], cl

mov [di], dl

inc si

dec di

cmp si, di

je reverseMini

popa

ret

endp

;app error

applicationStartError:

lea dx, applicationError

call puts

call exit

ret

;memory allocating

allocateMemory proc

push ax

push bx

mov bx, ((csize/16)+1)+256/16+((dsize/16)+1)+256/16

mov ah, 4Ah ;change size of memory block

int 21h

jc allocateMemoryError

jmp allocateMemoryEnd

allocateMemoryError:

lea dx, allocatingError

call puts

call exit

allocateMemoryEnd:

pop bx

pop ax

ret

endp

;get iterations

getIterations proc

pusha

xor ax, ax

call toInteger

mov ax, tempVariable

mov iterations, ax

popa

ret

endp

;load and run application

loadAndRun proc

mov ax, 4B00h ;load and execute

lea dx, applicationName

lea bx, env

int 21h

jb applicationStartError ;if lower

ret

endp

;file error call

fileErrorCall:

lea dx, fileError

call puts

call exit

ret

;get file name

getFilename proc

pusha

lea di, path

xor bx, bx

xor ax, ax

mov bx, spacePos

skipSpacesString:

cmp [di + bx], byte ptr ' '

jne unskippingString

inc bx

jmp skipSpacesString

unskippingString:

lea si, fileName

copyFilename:

xor ax, ax

mov al, [di + bx]

mov [si], al

inc bx

inc si

cmp [di + bx], byte ptr '$'

jne copyFilename

mov spacePos, bx

popa

ret

endp

;get string number

getStringNumber proc

pusha

xor ax, ax

call toInteger

mov ax, tempVariable

mov stringNumber, ax

popa

ret

endp

;get application name

getApplicationName proc

pusha

xor ax, ax

mov dx, offset fileName

mov ah, 3Dh ;open file

mov al, 00h

int 21h

mov descriptor, ax

mov bx, ax

jnc readFilePart

jmp fileErrorCall;

readFilePart:

mov ah, 42h ;set pointer of file (lseek)

mov cx, word ptr [offset pointerPosition]

mov dx, word ptr [offset pointerPosition + 2]

mov al, 0

mov bx, descriptor

int 21h

mov cx, partSize

lea dx, part

mov ah, 3Fh ;read file with descriptor

mov bx, descriptor

int 21h

mov realPartSize, ax

call searchApplicationName

call memset

cmp realPartSize, partSize

jb closeFile

mov bx, stringNumber

cmp endlCounter, bx

je closeFile

mov cx, word ptr [offset pointerPosition]

mov dx, word ptr [offset pointerPosition + 2]

add dx, ax

adc cx, 0

mov word ptr [offset pointerPosition], cx

mov word ptr [offset pointerPosition + 2], dx

jmp readFilePart

closeFile:

exitFromFile:

mov ah, 3Eh ;close file

mov bx, descriptor

int 21h

popa

ret

endp