Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет

информатики и радиоэлектроники»

Факультет компьютерного проектирования

Кафедра Информатики

Дисциплина «Конструирование программ»

**ОТЧЕТ**

к лабораторной работе №3

на тему:

**«Создание простой программы на языке ассемблера. Обработка символьных данных.»**

БГУИР 6-05-0612-02 005

|  |
| --- |
| Выполнила студент группы 353504  АНТОНОВА Лидия Сергеевна |
|  |
| (дата, подпись студента) |
| Проверил ассистент каф. Информатики  РОМАНЮК Максим Валерьевич |
|  |
| (дата, подпись преподавателя) |

Минск 2024

# 1 Индивидуальное задание

**Задание 1. Вариант 1.** Написать программу «Hello, world!».

**Задание 2. Вариант 5.** Отсортировать слова в строке по значению ASCII кодов символов.

# 2 Выполнение работы

В данной лабораторной работе были реализованы две программы на ассемблере с использованием эмулятора Emu8086. Первая программа выводит сообщение "Hello, World!", а вторая сортирует слова в строке по значению ASCII кодов символов.

В первом задании была написана программа, которая выводит на экран сообщение "Hello, World!". Реализация выполнена в двух вариантах: для .COM и .EXE файлов. Ниже представлен листинг кода для .COM файла.

.model tiny

.code

org 100h

start:

mov ah,9

mov dx,offset message

int 21h

ret

message db "Hello World!",0Dh,0Ah,'$'

end start

Листинг для .EXE файла.

.model small

.stack 100h

.code

start:

mov ax, @data

mov ds, ax

mov dx, offset message

mov ah, 9

int 21h

mov ax, 4C00h

int 21h

.data

message db "Hello World!",0Dh,0Ah,'$'

end start

**.COM файл**: Используется простая модель памяти. Вывод осуществляется с помощью прерывания 21h, функция 09h, которая отображает строку до символа $.

**.EXE файл**: Используется модель small, которая позволяет работать с данными в сегментах. После вывода сообщения программа завершает свою работу с помощью функции 4C00h.

Во втором задании была реализована программа, сортирующая слова в строке по значению ASCII кодов символов. Пользователь вводит строку, и программа выводит отсортированную версию. Ниже представлен листинг кода.

.model small

.stack 100h

.data

str db 202 dup(0)

len db 0

msgEmpty db 'String is empty.$'

msgOverflow db 'String is overflow.$'

msgInput db 'Input your string: $'

msgOutput db 'Sorted words: $'

msgEnter db 0Dh, 0Ah, '$'

.code

start:

mov ax, @data

mov ds, ax

mov dx, offset msgInput

mov ah, 09h

int 21h

xor cx, cx

lea si, str

inputLoop:

mov ah, 01h

int 21h

cmp al, 0Dh

je inputEnd

mov [si], al

inc si

inc cx

cmp cx, 200

jge inputEnd

jmp inputLoop

inputEnd:

mov byte ptr[si], 0Dh

mov len, cl

cmp cx, 0

je empty

cmp cx, 201

jge overflow

lea di, str

xor bx, bx

mov cx, 0

goSort:

xor dx, dx

mov dl, len

jmp sort

outputSting:

mov dx,offset msgEnter

mov ah,09h

int 21h

lea si, str

mov dx,offset msgOutput

mov ah,09h

int 21h

lea si, str

jmp outputSorted

empty:

mov dx, offset msgEmpty

mov ah, 09h

int 21h

jmp exit

overflow:

mov dx, offset msgOverflow

mov ah, 09h

int 21h

jmp exit

exit:

mov ax, 4C00h

int 21h

sort proc

mov cx, 0

xor bx, bx

lea si, str

lea di, str

nextChar:

cmp cl, len

je endSort

cmp byte ptr[di], ' '

je spaceFound

cmp bx, 0

je newWord

jmp processChar

spaceFound:

mov bx, 0

processChar:

inc di

inc cx

jmp nextChar

newWord:

inc bx

cmp si, di

jne sortWords

jmp processChar

sortWords:

jmp compareWords

endCompare:

cmp ax, di

je caseFirst

mov bx, 1

cmp ax, si

je caseSecond

caseFirst:

push di

push si

mov cx, 0

jmp countWordLength

countWordLength:

cmp byte ptr [di], ' '

je shiftStringRight

cmp byte ptr [di], 0Dh

je shiftStringRight

inc di

inc cx

jmp countWordLength

shiftStringRight:

pop si

pop di

push si

push di

inc cx

shiftLoop:

mov al, [di]

push si

mov bl, [si]

mov [si], ' '

inc si

innerShiftLoop:

mov dl, [si]

mov [si], bl

mov bl, dl

inc si

cmp si, di

jl innerShiftLoop

pop si

mov [si], al

inc si

inc di

dec cx

jnz shiftLoop

mov [si-1], ' '

pop di

pop si

mov cx, di

lea si, str

jmp nextChar

caseSecond:

inc si

cmp si, di

je nextChar

cmp byte ptr[si], ' '

je findSpaceForSi

cmp bx, 0

je sortWords

jmp caseSecond

findSpaceForSi:

mov bx, 0

jmp caseSecond

endSort:

lea si, str

jmp outputSting

sort endp

compareWords proc

; si first

; di second

push bx

push dx

push cx

mov cx, si

mov dx, di

compareLoop:

mov al, [si]

mov bl, [di]

cmp al, 20h

je returnFirst

cmp bl, 20h

je returnSecond

cmp al, bl

jne returnLess

inc si

inc di

jmp compareLoop

returnLess:

cmp al, bl

jb returnFirst

ja returnSecond

returnFirst:

mov ax, cx

jmp endCompareWords

returnSecond:

mov ax, dx

jmp endCompareWords

endCompareWords:

mov si, cx

mov di, dx

pop bx

pop dx

pop cx

jmp endCompare

compareWords endp

outputSorted proc

lea si, str

mov cl, len

outputLoop:

cmp cl, 0

je endOutput

mov ah, 02h

mov dl, [si]

int 21h

inc si

dec cl

jmp outputLoop

endOutput:

jmp exit

outputSorted endp

end start

В рамках второго задания лабораторной работы была разработана программа, которая сортирует слова в строке по значению ASCII кодов символов. Программа начинается с инициализации необходимых сегментов памяти и определения переменных, которые будут использоваться для хранения пользовательского ввода и сообщений.

При запуске программа запрашивает у пользователя ввод строки с помощью прерывания 21h, функции 09h. Ввод осуществляется посимвольно, пока не будет нажата клавиша Enter (код 0Dh). Каждый введённый символ сохраняется в массиве, который изначально инициализируется нулями. После завершения ввода программа проверяет, был ли введён текст. Если строка пуста, выводится сообщение об этом, а если превышен лимит длины строки, выводится соответствующее предупреждение.

Если пользователь ввёл строку, программа переходит к процессу сортировки. Она использует алгоритм пузырьковой сортировки для организации слов в строке. Для этого программа проходит по каждому символу строки и определяет, является ли он пробелом, чтобы разделить слова. При нахождении слова программа сравнивает его с другими словами, используя их ASCII коды. Если текущее слово больше по значению, происходит обмен значениями.

После завершения сортировки программа готовит вывод. Сначала она выводит сообщение о том, что будет показан отсортированный список слов, затем выводит отсортированную строку. Для этого используется функция прерывания 21h, которая позволяет выводить символы на экран.

Изображение выглядит как текст, снимок экрана, Шрифт

Автоматически созданное описание

Рисунок 2.1 — Пример выполнения задания

# Вывод

В ходе выполнения лабораторной работы была изучена работа с ассемблером, а также освоены основные операции ввода и вывода, сортировки строк и манипуляции с памятью. Обе программы успешно выполнили поставленные задачи, продемонстрировав возможности языка ассемблера.