

Учреждение образования
«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ»
Кафедра информатики

Отчет по лабораторной работе №4

Работа со строками

Выполнил: студент гр. 053502
Герчик Артём Вадимович

Руководитель: ст. преподаватель
Шиманский В.В.

Минск 2021

СОДЕРЖАНИЕ

1. Введение
2. Постановка задачи
3. Программная реализация
4. Выводы

Литература

Приложение

1. Введение

Целью данной работы является изучить следующий материал:

- 1) Представление строки в языке Ассемблер.
- 2) Команды `movsb`, `movsw`, `stosb`, `stosw`, `lodsб`, `lodsw`.
- 3) Назначение флага направления, команды `CLD` и `STD`.
- 4) Префиксы `REP`, `REPE`, `REPNE`.

2. Постановка задачи

2.1. Текст задания

В четвертой лабораторной работе необходимо ввести строку с клавиатуры, сделать ее обработку согласно заданию и показать результат на экране.

2.2. Условие задания

С клавиатуры вводится предложение. Необходимо найти и показать все слова, в которых гласные и согласные буквы чередуются. Считается, что строка состоит из слов, разделенных произвольным числом пробелов.

3. Программная реализация

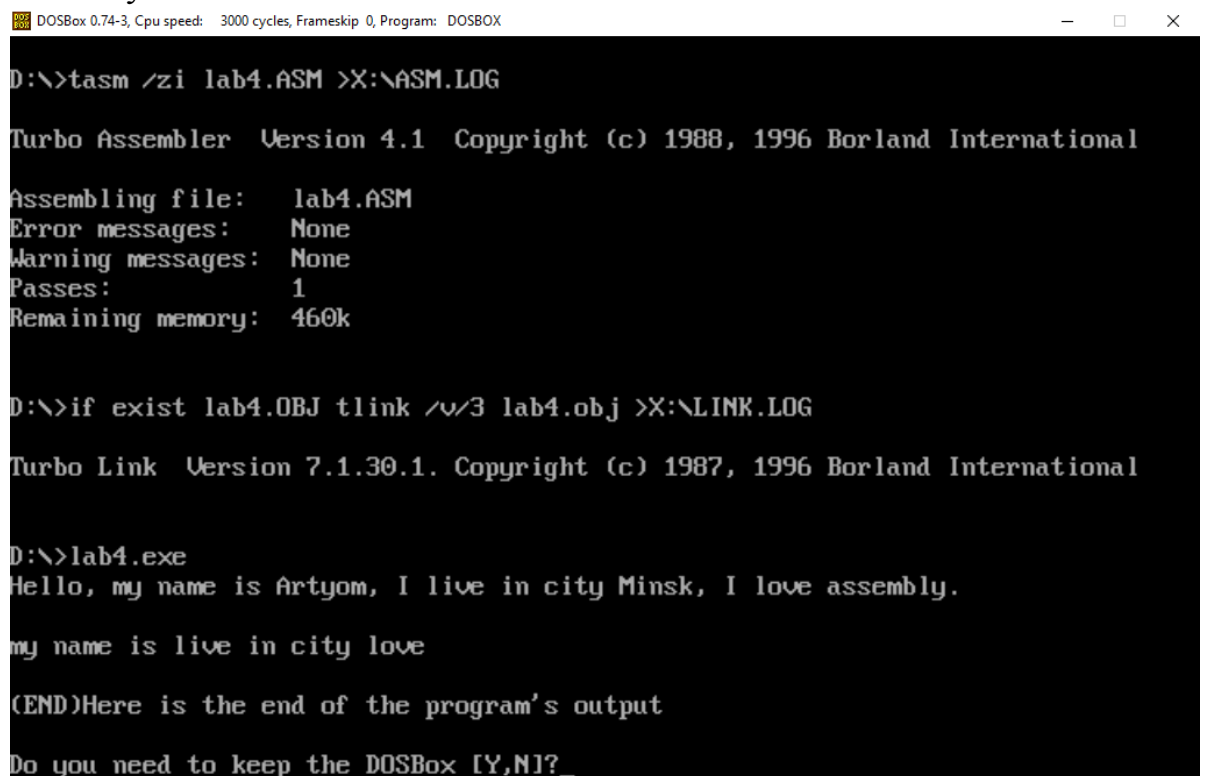
3.1. В главной программе вводится произвольное предложение, с произвольным количеством пробелом между словами. После этого, происходит проверка каждого слова на чередование гласных и согласных.

3.2. Результат можно видеть в окне консоли

3.3. Примеры:

3.3.1 Пример ввода предложения с нормальным количеством пробелов:

Предложение: “Hello, my name is Artyom, I live in city Minsk, I love assembly.”

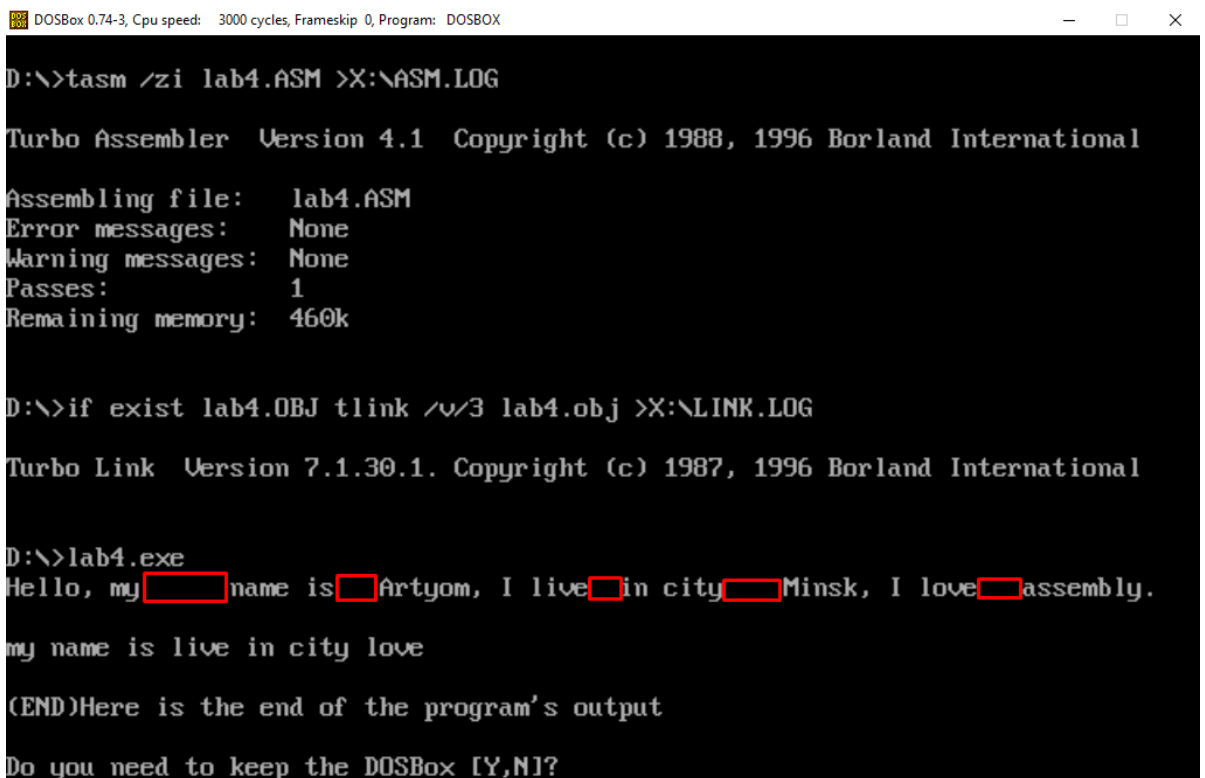


```
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DOSBOX
D:\>tasm /zi lab4.ASM >X:\ASM.LOG
Turbo Assembler Version 4.1 Copyright (c) 1988, 1996 Borland International
Assembling file: lab4.ASM
Error messages: None
Warning messages: None
Passes: 1
Remaining memory: 460k
D:\>if exist lab4.OBJ tlink /v/3 lab4.obj >X:\LINK.LOG
Turbo Link Version 7.1.30.1. Copyright (c) 1987, 1996 Borland International
D:\>lab4.exe
Hello, my name is Artyom, I live in city Minsk, I love assembly.
my name is live in city love
(END)Here is the end of the program's output
Do you need to keep the DOSBox [Y,N]?_
```

В ответ получаем слова: my, name, is, live, in, city, love. В этих словах действительно происходит чередование гласных и согласных букв.

3.3.2 Пример ввода предложения с произвольным количеством пробелов:

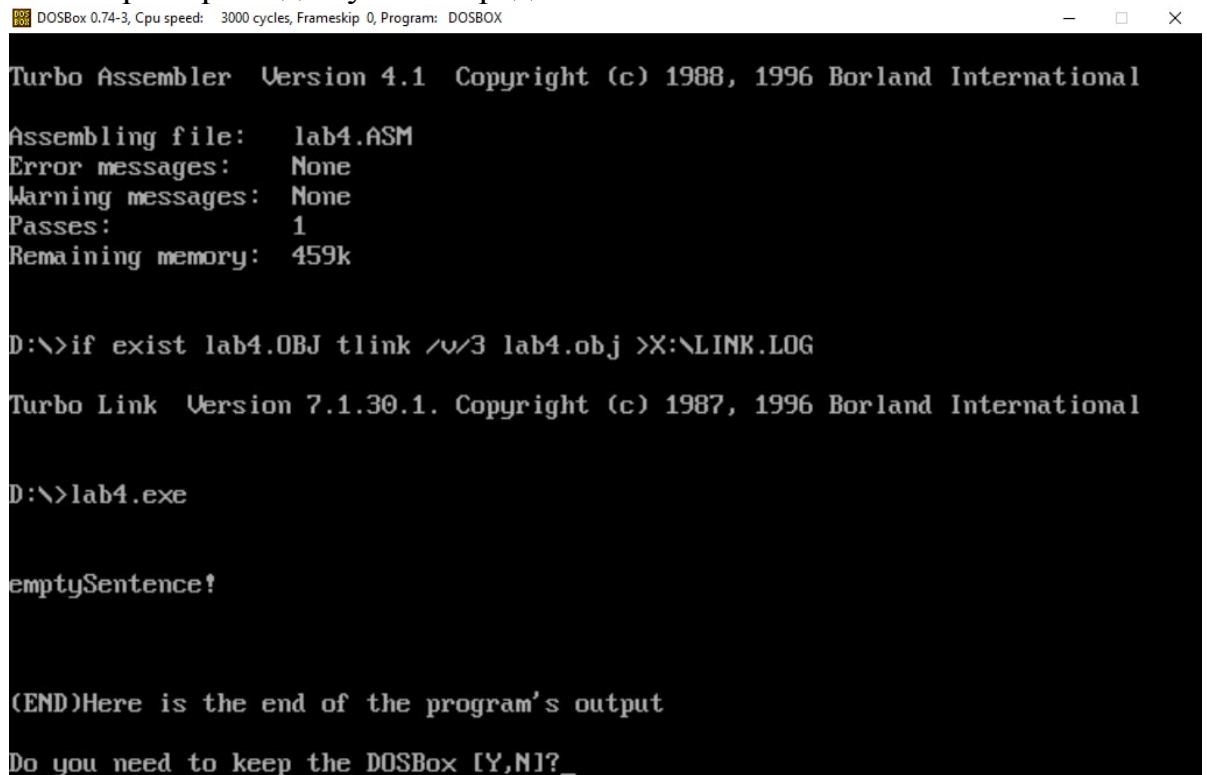
Предложение: “Hello, my name is Artyom, I live in city Minsk, I love assembly.” Места, где пробелов больше 1 выделены красным цветом.



```
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DOSBOX
D:\>tasm /zi lab4.ASM >X:\ASM.LOG
Turbo Assembler Version 4.1 Copyright (c) 1988, 1996 Borland International
Assembling file: lab4.ASM
Error messages: None
Warning messages: None
Passes: 1
Remaining memory: 460k
D:\>if exist lab4.OBJ tlink /v/3 lab4.obj >X:\LINK.LOG
Turbo Link Version 7.1.30.1. Copyright (c) 1987, 1996 Borland International
D:\>lab4.exe
Hello, my name is Artyom, I live in city Minsk, I love assembly.
my name is live in city love
(END)Here is the end of the program's output
Do you need to keep the DOSBox [Y,N]?
```

В ответ получаем слова: my, name, is, live, in, city, love. В этих словах действительно происходит чередование гласных и согласных букв.

3.3.3 Пример ввода пустого предложения:



```
Turbo Assembler Version 4.1 Copyright (c) 1988, 1996 Borland International
Assembling file: lab4.ASM
Error messages: None
Warning messages: None
Passes: 1
Remaining memory: 459k

D:\>if exist lab4.OBJ tlink /v/3 lab4.obj >X:\LINK.LOG

Turbo Link Version 7.1.30.1. Copyright (c) 1987, 1996 Borland International

D:\>lab4.exe

emptySentence!

(END)Here is the end of the program's output

Do you need to keep the DOSBox [Y,N]?_
```

Результат: вывод ошибки, и завершение программы.

3.3.4 Пример ввода предложения с дополнительными символами:

Предложение: “Hello, My name is Artyom, I Have walkie-talkies!”

```
Drive D is mounted as local directory ./code/

Z:\>d:

D:\>set PATH=C:\TASM

D:\>TASM TEST.ASM
Turbo Assembler Version 4.1 Copyright (c) 1988, 1996 Borland International

Assembling file:    TEST.ASM
Error messages:     None
Warning messages:   None
Passes:             1
Remaining memory:   462k

D:\>TLINK TEST
Turbo Link Version 7.1.30.1. Copyright (c) 1987, 1996 Borland International

D:\>TEST
Hello, My name is Artyom, I Have walkie-talkies!

My name is Have

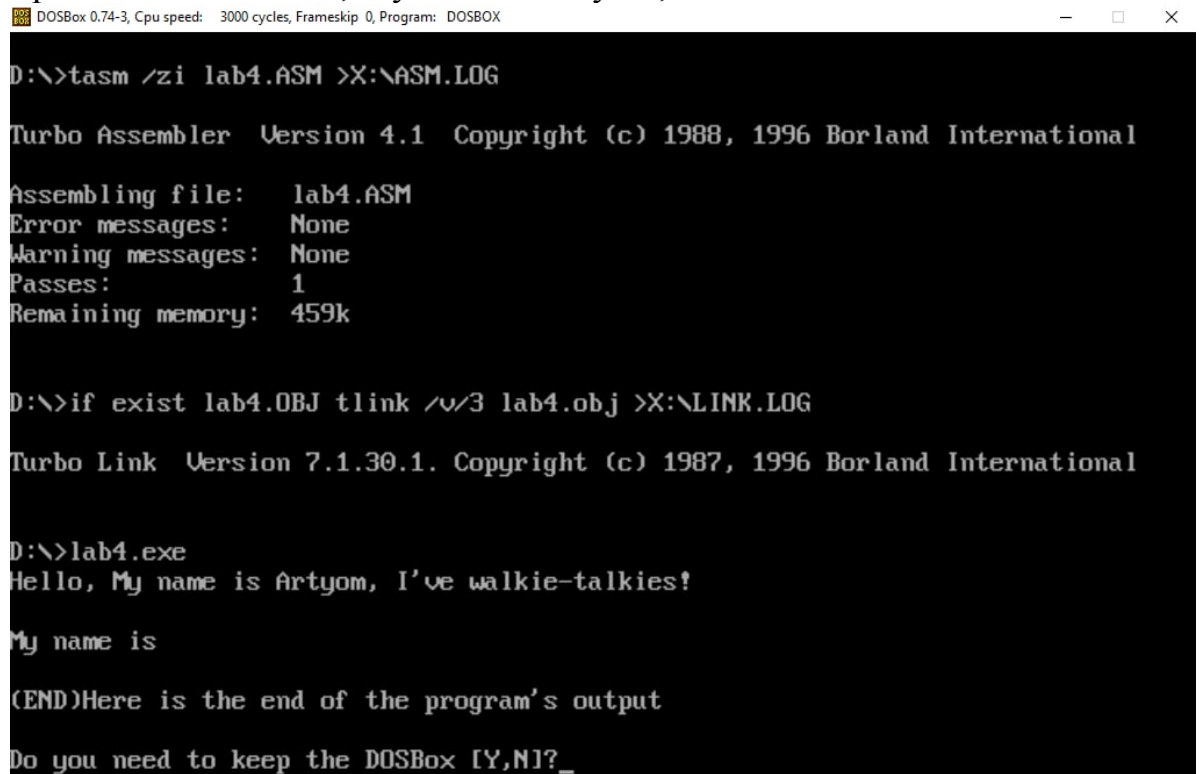
D:\>[
```

В ответ получаем слова: My, name, is, Have. В этих словах действительно происходит чередование гласных и согласных букв.

P.S. Символ апострофа просто не вводится в консоли (лично на моем устройстве) MAC OS.

3.3.5 Пример ввода предложения с дополнительными символами:

Предложение: “Hello, My name is Artyom, I’ve walkie-talkies!”



```
DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: DOSBOX

D:\>tasm /zi lab4.ASM >X:\ASM.LOG

Turbo Assembler Version 4.1 Copyright (c) 1988, 1996 Borland International

Assembling file:   lab4.ASM
Error messages:   None
Warning messages: None
Passes:           1
Remaining memory: 459k

D:\>if exist lab4.OBJ tlink /v/3 lab4.obj >X:\LINK.LOG

Turbo Link Version 7.1.30.1. Copyright (c) 1987, 1996 Borland International

D:\>lab4.exe
Hello, My name is Artyom, I’ve walkie-talkies!

My name is

(END)Here is the end of the program’s output

Do you need to keep the DOSBox [Y,N]?_
```

В ответ получаем слова: My, name, is. В этих словах действительно происходит чередование гласных и согласных букв.

P.S. Данный скриншот сделан на операционной системе Windows 10.
Здесь апостроф вводится в консоль.

4. Выводы

На практике было изучено и опробовано, в соответствии с поставленной задачей:

Представление строки в языке Ассемблер.

Команды movsb, movsw, stosb, stosw, lodsb, lodsw.

Назначение флага направления, команды CLD и STD.

Префиксы REP, REPE, REPNE.

Литература

1. Юров В.И. – «Assembler. Учебник для вузов. 2-ое издание, 2003 год».
2. Юров В.И. – «Assembler. Практикум. 2-ое издание, 2006 год».
3. Калашников О.А. – «Ассемблер - это просто. 2-ое издание, 2011 год».

Приложение

.model small

.stack 512

.386

.data

inputStr db 1000 dup(?)

space db 10,'\$'

vowelWasRecently dw 0

consonantsWasRecently dw 0

thisWordBad dw 0

counter dw 0

testTemp dw 0

spaceWas dw 0

.code

main:

mov ax, @data

mov ds, ax

mov si, offset inputStr

input:

mov ah, 01h

int 21h

cmp al, 13 ; enter button

je displayWord

mov [si], al

inc si

jmp input

vowelsHere:

mov ax, vowelWasRecently

cmp ax, 1

je resume

inc di

inc counter

mov consonantsWasRecently, 0
mov vowelWasRecently, 1

jmp again

consonantsHere:

mov ax, consonantsWasRecently

cmp ax, 1
je resume

inc di
inc counter

mov vowelWasRecently, 0
mov consonantsWasRecently, 1

jmp again

movCounterForGetBack:

mov cx, counter

getBack:

cmp cx, 0
je movCounterForOutput

dec di
dec cx

jmp getBack

movCounterForOutput:

mov cx, counter

cmp cx, 1
je label1

jmp outputWord

label1:

```
    mov spaceWas, 1
    inc di
    jmp tempLabel
```

outputWord:

```
    cmp cx, 0
    je tempLabel
```

```
    mov dl, [di]
```

```
    mov ah, 02h
    int 21h
```

```
    mov spaceWas, 0
```

```
    inc di
    dec cx
```

```
    jmp outputWord
```

doInc:

```
    inc di
    jmp again
```

tempLabel:

```
    cmp spaceWas, 0
    jne labelInTempLabel
```

```
    mov dl, ' ' ; newline
    mov ah, 02h
    int 21h
```

```
    mov spaceWas, 1
    mov consonantsWasRecently, 0
    mov vowelWasRecently, 0
    mov counter, 0
    mov thisWordBad, 0
    inc di
```

jmp again

labelInTempLabel:

mov consonantsWasRecently, 0

mov vowelWasRecently, 0

mov counter, 0

mov thisWordBad, 0

cmp cx, 1

je doInc

jmp testForNextSpace

checkForNextSpace:

mov spaceWas, 0

jmp again

testForNextSpace:

inc di

mov dl, [di]

cmp dl, ' '

jne checkForNextSpace

mov spaceWas, 1

jmp again

displayWord:

mov si, '\$'

mov di, offset inputStr

mov dl, 13 ; enter buton

mov ah, 02h

int 21h

mov dl, 10 ; newline

mov ah, 02h

int 21h

jmp again

littleLoop:

inc di

mov dl, [di]

cmp dl, ' '
je temphuita

mov counter, 0
mov thisWordBad, 0

jmp littleLoop

temphuita:

mov thisWordBad, 0
inc di
jmp again

dotFinded:

cmp counter, 0
jg movCounterForGetBack
jmp last

testLoop:

cmp vowelWasRecently, 1
je movCounterForGetBack

cmp consonantsWasRecently, 1
je movCounterForGetBack

inc di

again:

cmp thisWordBad, 1
je littleLoop

mov dl, [di]

cmp dl, ' '
je skip


```
cmp dl, 10  
je last
```

```
cmp dl, '.'  
je dotFinded
```

```
startCheck:
```

```
testForVowel: ; Aa Ee Ii Oo Uu Yy
```

```
    cmp dl, 'A'  
    je vowelsHere  
    cmp dl, 'a'  
    je vowelsHere
```

```
    cmp dl, 'E'  
    je vowelsHere  
    cmp dl, 'e'  
    je vowelsHere
```

```
    cmp dl, 'I'  
    je vowelsHere  
    cmp dl, 'i'  
    je vowelsHere
```

```
    cmp dl, 'O'  
    je vowelsHere  
    cmp dl, 'o'  
    je vowelsHere
```

```
    cmp dl, 'U'  
    je vowelsHere  
    cmp dl, 'u'  
    je vowelsHere
```

```
    cmp dl, 'Y'  
    je vowelsHere  
    cmp dl, 'y'  
    je vowelsHere
```

```
testForConsonant: ; Bb Cc Dd Ff Gg Hh Jj Kk Ll Mm Nn Pp Qq Rr Ss  
Tt Vv Ww Xx Zz
```

```
    cmp dl, 'B'
```

je consonantIsHere
cmp dl, 'b'
je consonantIsHere

cmp dl, 'C'
je consonantIsHere
cmp dl, 'c'
je consonantIsHere

cmp dl, 'D'
je consonantIsHere
cmp dl, 'd'
je consonantIsHere

cmp dl, 'F'
je consonantIsHere
cmp dl, 'f'
je consonantIsHere

cmp dl, 'G'
je consonantIsHere
cmp dl, 'g'
je consonantIsHere

cmp dl, 'H'
je consonantIsHere
cmp dl, 'h'
je consonantIsHere

cmp dl, 'J'
je consonantIsHere
cmp dl, 'j'
je consonantIsHere

cmp dl, 'K'
je consonantIsHere
cmp dl, 'k'
je consonantIsHere

cmp dl, 'L'
je consonantIsHere
cmp dl, 'l'
je consonantIsHere

cmp dl, 'M'

je consonantIsHere
cmp dl, 'm'
je consonantIsHere

cmp dl, 'N'
je consonantIsHere
cmp dl, 'n'
je consonantIsHere

cmp dl, 'P'
je consonantIsHere
cmp dl, 'p'
je consonantIsHere

cmp dl, 'Q'
je consonantIsHere
cmp dl, 'q'
je consonantIsHere

cmp dl, 'R'
je consonantIsHere
cmp dl, 'r'
je consonantIsHere

cmp dl, 'S'
je consonantIsHere
cmp dl, 's'
je consonantIsHere

cmp dl, 'T'
je consonantIsHere
cmp dl, 't'
je consonantIsHere

cmp dl, 'V'
je consonantIsHere
cmp dl, 'v'
je consonantIsHere

cmp dl, 'W'
je consonantIsHere
cmp dl, 'w'
je consonantIsHere

cmp dl, 'X'

```
je consonantIsHere  
cmp dl, 'x'  
je consonantIsHere
```

```
cmp dl, 'Z'  
je consonantIsHere  
cmp dl, 'z'  
je consonantIsHere
```

```
mov testTemp, 1  
jmp testLoop
```

resume:

```
mov thisWordBad, 1  
mov consonantsWasRecently, 0  
mov vowelWasRecently, 0  
jmp again
```

skip:

```
jmp movCounterForGetBack
```

next:

```
cmp thisWordBad, 0  
je movCounterForGetBack
```

last:

```
lea dx, space  
mov ah, 09h ; display string, which is stored in dx  
int 21h
```

```
mov ah, 4ch  
int 21h
```

end main