МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное образовательное   
учреждение высшего образования

«Чувашский государственный университет имени И. Н. Ульянова»

Факультет информатики и вычислительной техники

Кафедра вычислительной техники

# **Лабораторная работа 2.**

**ОРГАНИЗАЦИЯ ВНЕШНИХ ПОДПРОГРАММ НА ЯЗЫКЕ АССЕМБЛЕРА**

**Цель работы**: ознакомление с процедурами на языке программирования Assembler, написание собственной процедуры с сохранением параметров через стек.

**Задание.** Составить внешнюю подпрограмму. Подпрограмму оформить в отдельном модуле. Для проведения отладки подпрограммы необходимо также составить основную программу на языке ассемблера, вызывающую составленную подпрограмму как внешнюю. Тип вызова выбирается самостоятельно.   
procedure Delete(var S: string; Start, Len: byte). Удаляет в строке S символы с позиции Start и длинной Len. Если Start, больше длины S, то ничего не изменяется.

; Файл lab2p.asm  
code            segment

                assume  cs:code,ds:code

                public  Delete

; procedure Delete(var S: string; Start, Len: byte)

; Удаляет в строке S символы с позиции Start и длинной Len

; Если Start больше длины S, то ничего не изменяется

Delete          proc

S               equ     dword ptr[bp+8]

Start           equ     byte ptr[bp+6]

Len             equ     byte ptr[bp+4]

                push    bp

                mov     bp,sp

                push    bx

                push    cx

                push    si

                push    di

                push    es

; Формулы

; Len = min(Len, |S| - Start + 1)       // Длина удаляемого куска

; bx = addr(S)                          // адрес начала строки

; di = Start [+ addr(S)]                // адрес начала записи

; si = Start + Len [+ addr(S)]          // адрес записываемой части

; cx = |S| - Start - Len + 1            // количество записываемых символов

; |S| = |S| - Len                       // новая длина строки

                ; Загрузка данных

                les     bx,S            ; bx = addr(S)

                xor     cx,cx

                mov     cl,Start

                mov     di,cx           ; di = Start

                mov     cl,Len

                mov     si,cx           ; si = Len

                mov     cl,es:[bx]      ; cl = |S|

                cmp     di,cx           ; Start > |S|

                ja      exit            ; exit

                sub     cx,di           ; cx = |S| - Start

                add     cx,1            ; cx = |S| - Start + 1

                cmp     si,cx           ; Len ? |S| - Start + 1

                jb      del

                mov     si,cx           ; Len = |S| - Start + 1

del:            sub     es:[bx],si      ; |S| = |S| - Len

                sub     cx,si           ; cx = |S| - Start - Len + 1

                add     si,di           ; si = Start + Len

                add     di,bx           ; di = Start + addr(S)

                add     si,bx           ; si = Start + Len + addr(S)

                rep movs byte ptr [si],[di]

exit:           pop     es

                pop     di

                pop     si

                pop     cx

                pop     bx

                mov     sp,bp

                pop     bp

                ret

Delete          endp

code            ends

                end

; Файл lab2m.asm

sseg            segment stack   'stack'

                dw      256 dup(?)

sseg            ends

data            segment

msg1            db      "Original string: $"

msg2            db      0Dh,0Ah,"Converted string: $"

S1              db      9,"My String"

Start1          db      10

Len1            db      5

data            ends

code            segment

                assume  cs:code,ss:sseg,ds:data

                extrn   Delete: near

print\_msg       proc

                push    ax

                mov     ah,09h

                int     21h

                pop     ax

                ret

print\_msg       endp

print\_str       proc

                push    ax

                push    bx

                push    cx

                push    dx

                mov     ah,02h

                xor     cx,cx

                mov     cl,[bx]

next\_sym:       inc     bx

                mov     dl,[bx]

                int     21h

                loop    next\_sym

                pop     dx

                pop     cx

                pop     bx

                pop     ax

                ret

print\_str       endp

\_start:         mov     ax,data

                mov     ds,ax

                lea     dx,msg1

                call    print\_msg

                lea     bx,S1

                call    print\_str

                mov     ax,seg S1

                push    ax

                mov     ax,offset S1

                push    ax

                mov     al,Start1

                push    ax

                mov     al,Len1

                push    ax

                call    Delete

                lea     dx,msg2

                call    print\_msg

                lea     bx,S1

                call    print\_str

                mov     ax,4c00h

                int     21h

code            ends

                end     \_start

**Стек после вызова процедуры: Стек после сохранения bp:**

|  |  |  |  |
| --- | --- | --- | --- |
| Содержимое | ss:sp | Содержимое | ss:sp |
| IP выхода | +0 | bp | +0 |
| Len | +2 | IP выхода | +2 |
| Start | +4 | Len | +4 |
| offset S | +6 | Start | +6 |
| seg S | +10 | offset S | +8 |
|  |  | seg S | +12 |

**Результаты работы программы:**

Перед вызовом процедуры строка имела вид: “My string”, после вызова процедуры стала иметь вид: “Ming”. Таким образом из строки была удалена подстрока “y str”.

**Вывод**: произошло ознакомление с процедурами на языке программирования Assembler, была написана собственная процедура с сохранением параметров через стек.