ФГБОУ ВО «Чувашский государственный университет»

им. И.Н. Ульянова.

Кафедра Вычислительной техники

Дисциплина «ЭВМ и периферийное устройство»

Лабораторная работа №2

ОРГАНИЗАЦИЯ ВНЕШНИХ ПОДПРОГРАММ НА ЯЗЫКЕ АССЕМБЛЕРА

Выполнил студент:

Группы ИВТ-11-21

Еремеев Сергей Алексеевич

Проверил:

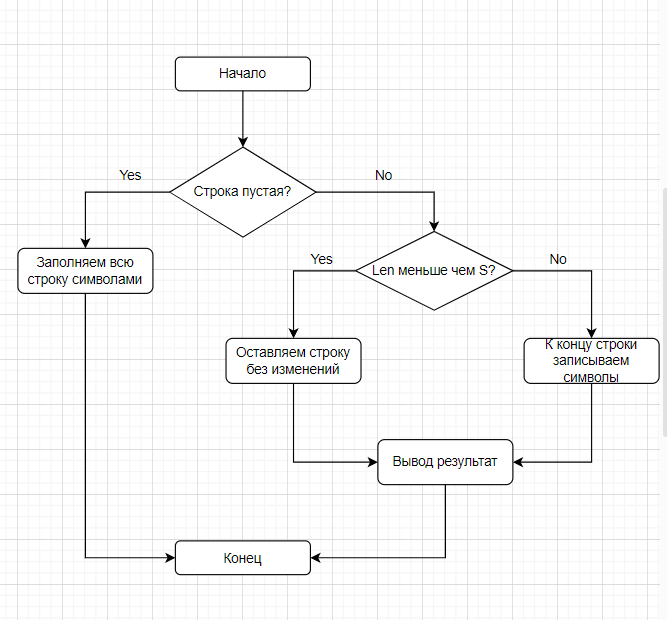
доц. Андреева А.А.

Чебоксары 2024

**Цель работы**: Составить внешнюю подпрограмму.

**Индивидуальное задание**: 9. function PadCh(S: string; C: char; Len: byte): string. Воз-вращает строку, в которой S смещена влево, а остаток строки заполнен символами С. Для этого знаки С включаются справа от конца S до тех пор, пока общая длина строки не станет равной Len. Если S длиннее, чем Len, то строка не изменяется. Если S пустая строка, то возвращается строка из Len символов С.

**Блок-схема алгоритма подпрограмм:**

****

**Стек:**

|  |  |  |
| --- | --- | --- |
| **Res** | **Bp+14** | **FFFC** |
| **ds** | **Bp+12** | **FFFA** |
| **S** | **Bp+10** | **FFF8** |
| **Ch1** | **Bp+8** | **FFF6** |
| **Len** | **Bp+6** | **FFF4** |
| **cs** | **Bp+4** | **FFF2** |
| **ip** | **Bp+2** | **FFF0** |
| **bp** | **Bp = sp** | **FFEE** |

**Текст внешней подпрограммы:**

code segment byte public

assume cs:code,ds:code

public PadCh

PadCh proc far

; адреса параметров в стеке:

Len equ byte ptr [bp+6] ; адрес параметра Len:Byte

Ch1 equ byte ptr [bp+8] ; адрес параметра Сh1 :Char

S equ dword ptr [bp+10]; адрес строки S:string

Res equ dword ptr [bp+14]; адрес строки результата

push bp ; сохранение bp

mov bp,sp ; настройка bp на вершину стека

push ds ; сохранение ds

les di,Res ; es:di:=адрес результата

lds si,S ; ds:si:=адрес исходной строки

cld ; очистка флага направления (инкремент)

lodsb ; al:=(ds:[si]), si:=si+1 (al - длина S)

stosb ; (es:[di]):=al, di:=di+1 (запись длины)

mov ch,0; подготовка сх в качестве счетчика

mov cl,al; количества символов строки S

jcxz CopyChStr ; если S - пустая строка (сх=0)

push cx ;Сохраняем значение длины строки

rep movsb ; Копируем строку cx:=cx-1, si:=si+1 и di:=di+1

pop cx; Получаем длину строки

sub cl, Len ;Находим разницу повторений

cmp cl,0

jge Exit

neg cl

jmp CopyStr

; Заполням строку символ С

CopyChStr:

mov cl, Len

CopyStr:

mov al, Ch1 ; записываем символ, которым будем заполнять строку

rep stosb ; заполняем строку символами cx:=cx-1 и di:=di+1

mov al, Len ; записываем длину строки

mov di, word ptr[bp+14] ; возвращаем индекс к началу строки

stosb ; записываем длину и смещаемся di:=di+1

Exit: pop ds ; восстановить ds

pop bp ; восстановить bp

ret 8; выход с удалением параметров Ch1,Ch2 и адреса S (Res удалять нельзя!)

PadCh endp

code ends

end

**Текст основной программы:**

code segment

assume cs:code,ds:code

extrn PadCh:far ; описание внешней far-процедуры

Res db 20 dup(?) ; строка результата

S db 6, 'String', 248 dup(?) ; исходная строка

start: mov ax, cs; настройка сегмента данных

mov ds,ax ; на сегмент кода

push ds; сегмент строки результата Res

mov ax, offset Res

push ax; смещение Res

push ds; сегмент исходной строки S

mov ax, offset S

push ax; смещение S

mov al, 'X'; параметр Ch1

push ax

mov al, 3; параметр Len

push ax

call PadCh ; far-вызов Change

; чтение адреса результата функции Change из стека

pop bx ; bx:=смещение Res

pop ds ; ds:=сегмент Res

mov ch, 0; подготовка в сх длины

; строки Res

mov cl, [bx]

jcxz Exit; выход, если строка Res пуста

Write: inc bx; продвижение указателя символа bx:=bx+1

mov dl, [bx]; dl:=очередной символ Res

mov ah, 2; вывод символа из dl

int 21h ; на экран средствами DOS

loop Write; цикл по длине строки Res

Exit: mov ax, 4c00h; завершение программы

int 21h ; чеpез функцию DOS

code ends

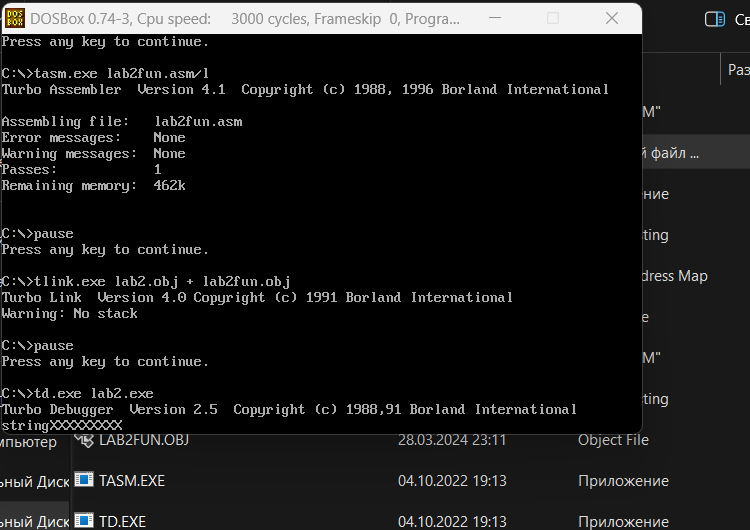
end start

**Использую команды:**

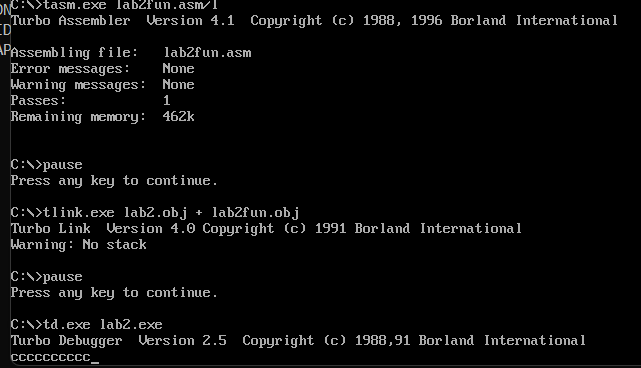
1. **JCXZ** проверяет значение регистра CX и, если оно равно нулю, производится передача управления по адресу, заданному операндом команды.
2. **REP** повторяет выполнение следующей за ними команды, работающей со строками. Количество повторений заносится в регистр CX до выполнения команды с префиксом.
3. **MOVSB** копирует один байт из ячейки памяти по адресу DS:SI в ячейку памяти по адресу ES:DI.  
   После выполнения команды, регистры SI и DI увеличиваются на 1, если флаг DF = 0, или уменьшаются на 1, если DF = 1.
4. **STOSB** сохраняет регистр AL в ячейке памяти по адресу ES:DI. После выполнения команды, регистр DI увеличивается на 1, если флаг DF = 0, или уменьшается на 1, если DF = 1.
5. **Команда CLD** в Ассемблере очищает флаг направления (DF).

**Результат работы:**

Когда строка = String, Len = 15, C = ‘X’



Когда строка пустая, Len = 10, C = ‘c’



Вопросы:

1. les di,Res ; es:di:=адрес результата написать с двумя mov?

mov di, word ptr [bp+14] ; загружает младшее слово адреса rезультата в регистр di

mov es, word ptr [bp+12] ; загружает базовый адрес сегмента адреса результата в регистр es

1. rep stosb ; заполняем строку символами cx:=cx-1 и di:=di+1 без строковых команд?

CopyChStr:

mov cl, Len ; Загружаем длину строки

mov al, Ch1 ; Загружаем символ, которым будем заполнять строку

FillLoop:

stosb ; Записываем символ в память

loop FillLoop ; Повторяем до завершения строки

1. ret 8; выход с удалением параметров Ch1,Ch2 и адреса S (Res удалять нельзя!) какие регистры изменяются?

1. BP - регистр базового указателя стека (BP) восстанавливается из стека.

2. SP - регистр указателя стека (SP) изменяется при удалении параметров Ch1, Ch2 и адреса S из стека.

BP обычно используется в качестве указателя базы текущего фрейма стека,  
SP - в качестве указателя вершины стека.