**Лабораторная работа №6**

**Тема**: "Подпрограммы"  
**Цель работы**: Целью работы является изучение, создание и применение подпрограмм на языке ассемблера.

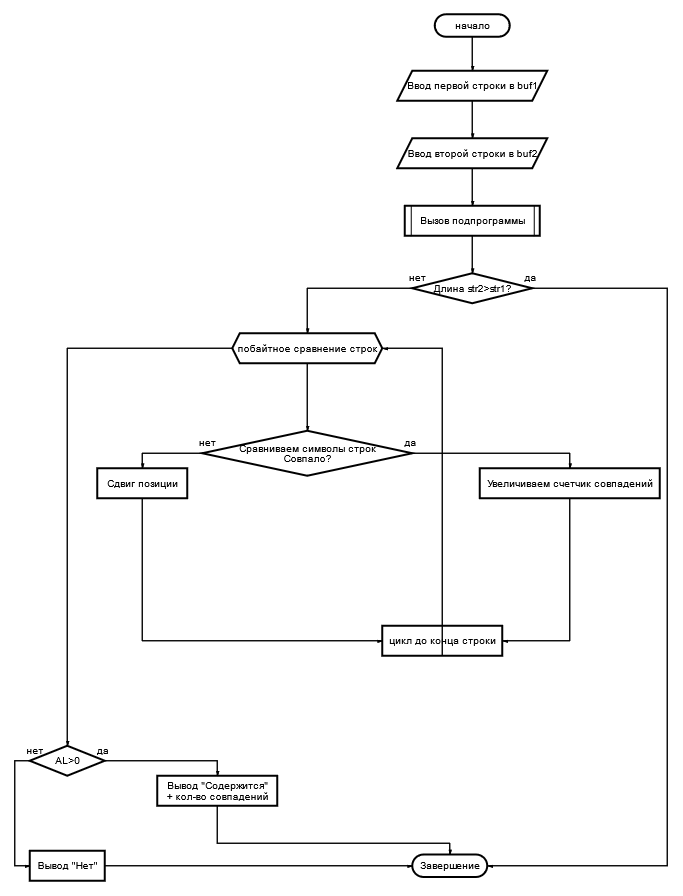
**Примечание:** Вариант №1  
Задание: Разработать подпрограмму, которая определяет, содержится ли одна

заданная строка в другой заданной строке, и если да, то начиная с какой

позиции. Разработать программу, которая вводит с клавиатуры две строки и

сообщает содержится ли одна в другой и сколько раз.

**Ход работы.**

**Блок-схема:** **Код:**data segment

prompt1 db 'Введите 1 строку: ',13,10,'$'

prompt2 db 13,10,'Введите 2 строку: '

crlf db 13,10,'$'

txtno db 13,10,'Нет$'

txtyes db 'Содержится $'

buf1 db 255,?

str1 db 255 dup (?)

buf2 db 255,?

str2 db 255 dup (?)

data ends

code segment

assume cs:code,ds:data

;Подпрограмма подсчёта кол-ва вхождений

;подстроки [si] в строку [di].

;Длины строк должны находиться по адресам

;[si-1] и [di-1]

;Кол-во вхождений возвращается в al

subroutine proc

xor al,al

mov cl,[di-1]

sub cl,[si-1]

jb Exit

inc cl

mov ch,al; или xor ch,ch

m1:

push cx

push di

push si

mov cl,[si-1]

xor ch,ch

compare\_strings:

mov dl, [si] ; Считываем байт из строки source

mov bl, [di] ; Считываем байт из строки destination

cmp dl, bl ; Сравниваем байты

jne m2

; Если байты равны:

inc si ; Переходим к следующему байту строки source

inc di ; Переходим к следующему байту строки destination

loop compare\_strings ; Уменьшаем CX и повторяем, если CX > 0

inc al

m2: pop si

pop di

inc di

pop cx

loop m1

Exit: ret

subroutine endp

start: mov ax,data

mov ds,ax

mov ah,9

lea dx,prompt1

int 21h

mov ah,10

lea dx,buf1

int 21h

mov ah,9

lea dx,prompt2

int 21h

mov ah,10

lea dx,buf2

int 21h

lea di,str1

lea si,str2

call subroutine; вызов подпрограммы

or al,al; или cmp al,0

jne Yes

No: lea dx,txtno

mov ah,9

int 21h

jmp quit

Yes: push ax

lea dx,crlf

mov ah,9

int 21h

lea dx,txtyes

int 21h

pop ax

xor ah,ah

mov bx,300Ah

div bl

add bh,ah

xor ah,ah

div bl

mov dx,3030h

add dx,ax

mov ah,2

int 21h

mov dl,dh

int 21h

mov dl,bh

int 21h

quit: xor ah,ah

int 16h

mov ah,4Ch

int 21h

code ends

end start

**Вывод:** Изучил, создал и применил подпрограммы на языке ассемблера.

**Контрольные вопросы**

**1. Что такое “ближние” и “дальние” подпрограммы?**  
Тип перехода near показывает, что тело процедуры описано в том же сегменте, что и её вызов. Тип перехода far обеспечивает вызов процедуры из других сегментов, с другим значением регистра CS. Такие процедуры обычно используются как отдельные объектные модули или в составе библиотек.

**2. Как определить “ближний” или “дальний” вариант команды call**

**использован в программе?**

Нужно либо посмотреть на тип процедуры в самой подпрограмме, либо, при косвенном вызове, посмотреть тип вызова (word – ближний, dword - дальний)

**3. Какие способы используются для передачи параметров в**

**подпрограммы?**

Для передачи входных параметров в подпрограмму и выходных в программу существует несколько способов. Чаще всего передача параметров осуществляется через регистры или через стек. Также может использоваться память.

**4. Может ли массив быть параметром подпрограммы?**

Да, может, если указать адрес его начала

**5. Можно ли использовать для чтения из стека параметров регистр sp**

**вместо bp?**

Можно, только вместо прямого чтения из sp, нужно занести его в bp:  
push bp  
mov bp,sp  
…  
pop bp  
Непосредственно использовать sp нежелательно, потому что сотрутся данные

**6. Что означает операнд команды ret?**call [ ptr] < тип вызова > - near (word) или far (dword).   
Если тип не задан, по умолчанию принимается near.   
< адрес процедуры > - имя или адрес процедуры. При ближнем вызове в стеке запоминается текущее значение регистра IP. При дальнем вызове в стеке запоминаются значения CS:IP.  
Если тип вызова не указан явно, он определяется типом, на который указывает , аналогично команде безусловного перехода jmp.ret <выражение>  
Значение <выражение> указывает размер стека в байтах, восстанавливаемого при возврате из процедуры.   
Переходит в программу по адресу, записанному в IP или CS:IP, в зависимости от типа перехода.  
**7. Какой последовательностью команд можно заменить команду “ret 8”?**Её можно заменить следующим блоком команд  
pop <регистр из стека>   
…  
pop <регистр из стека> ; 4 регистра (8 байт), которые нам нужно очистить   
ret