

**Национальный исследовательский университет
«Высшая школа экономики»**

Факультет компьютерных наук
Департамент программной инженерии

Титульный лист

Программа для нахождения убывающей подстроки

Исполнитель
Студент БПИ198
Смирнов Михаил Алексеевич

Москва, 2020

1. Постановка задачи

Требуется разработать консольную программу на языке FASM (flat assembler), которая находит в заданной строке первую слева направо последовательность N символов ($N = 4$), каждый из которых меньше предшествующего.

2. Описание входных и выходных данных

На вход к программе подается непустая строка, состоящая из ASCII-символов. Длина строки заранее не известна, однако считается, что она не должна превышать 1000 символов. Строка не содержит пробельных символов (включая символ перевода строки).

Если требуемая подстрока нашлась, то программа должна выводить эту подстроку и индекс ее первого элемента в исходной строке. Если подстрока не нашлась, то программа должна выводить соответствующее сообщение. Ввод и вывод программы должны сопровождаться пояснениями.

3. Описание алгоритма

Алгоритм находит убывающую подстроку за один (возможно, неполный) проход по исходной строке. Заведем переменную `int cur_len`, в которую запишем текущую длину убывающей последовательности. Занумеруем символы слева направо. Предположим, что первый символ - начало искомой последовательности. Тогда `cur_len = 1`. Теперь сравним первый символ со вторым. Если второй меньше, то продолжаем последовательность, т.е. увеличиваем `cur_len` на 1. В противном случае, начинаем новую последовательность, т.е. `cur_len = 1`. Продолжаем аналогичные действия до конца строки. Если в какой-то момент `cur_len` стала равной N, то мы нашли убывающую последовательность длины N. В силу порядка обхода эта последовательность будет первой. Текущий символ будет последним в подстроке, поэтому чтобы получить номер первый символа надо из номера текущего вычесть длину минус 1. Если иначе `cur_len` всегда $< N$, то такой последовательности нет. В этом случае алгоритм вернет -1. Код алгоритма на языке C:

```
// s - исходная строка.

int cur_len = 1;

int i = 1;

while(s[i] != '\0'){
    if(s[i] < s[i-1]){
        cur_len = cur_len + 1;
        if(cur_len == N) return i - (N - 1);
    }else cur_len = 1;
    i = i + 1;
}

return -1;
```

4. Описание методов и реализации

Для реализации ввода и вывода в консоль были использованы функции `printf` и `scanf` из языка C.

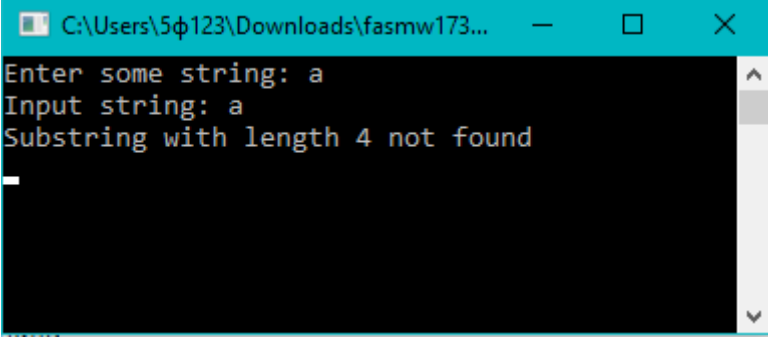
Метод `strFindSubstring` имеет 3 аргумента - входное сообщение (`char[]`), длина подстроки (`int`), указатель на возвращаемое значение (`int&`). Длина подстроки должна быть ≥ 2 . В противном случае метод завершится корректно, но ничего не найдет. Действительно, искать убывающую подстроку длины 1 не имеет смысла. Метод реализует описанный выше алгоритм с использованием цепочечных команд [1]. В данном случае регистры `si` и `di` будут указывать на одну строку, но со сдвигом на 1. Это требуется для сравнение соседних элементов в цикле.

Метод `getSubstring` имеет 4 аргумента - входное сообщение (`char[]`), выходная подстрока (`char[]`), длина (`int`), первый индекс (`int`). Метод копирует в подстроку символы входной строки в промежутке [первый индекс, первый индекс + длина). Считается, что данные корректны, т.е. копируемый промежуток не выходит за рамки входного сообщения. Также реализован с помощью цепочечных команд и цикла `loop`.

5. Тестирование

Ввод: a

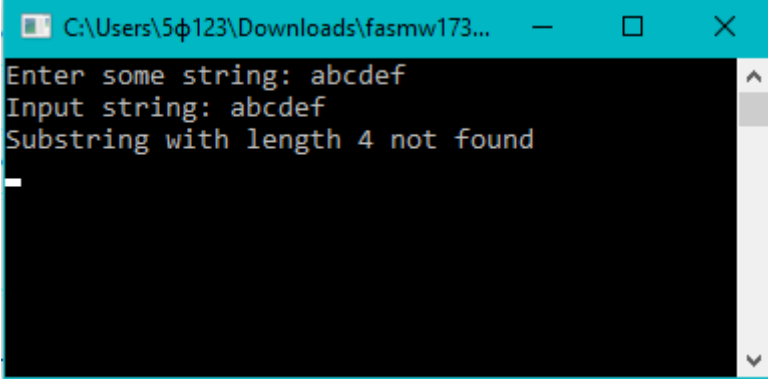
Обрабатывает 1 символ



```
C:\Users\5ф123\Downloads\fasmw173...
Enter some string: a
Input string: a
Substring with length 4 not found
```

Ввод: abcdef

Обрабатывает строку без убывающей последовательности



```
C:\Users\5ф123\Downloads\fasmw173...
Enter some string: abcdef
Input string: abcdef
Substring with length 4 not found
```

Ввод: dcba

Может находить убывающие последовательности

```
C:\Users\5ф123\Downloads\fasmw1732...
Enter some string: dcba
Input string: dcba
Substring was found: "dcba" at position 0
```

Ввод: edcba

Находит последовательность ровно заданной длины

```
C:\Users\5ф123\Downloads\fasmw17325\Proj...
Enter some string: edcba
Input string: edcba
Substring was found: "edcb" at position 0
```

Ввод: aaaa

Убывание должно быть строгим

```
C:\Users\5ф123\Downloads\fasmw173...
Enter some string: aaaa
Input string: aaaa
Substring with length 4 not found
```

Ввод: abcdef04321nDCBA

Находит только первую слева последовательность

```
C:\Users\5ф123\Downloads\fasmw17325\...
Enter some string: abcdef04321nDCBA
Input string: abcdef04321nDCBA
Substring was found: "4321" at position 7
```

Ввод: abcd dcba

Пробел является концом строки

```
C:\Users\5ф123\Downloads\fasmw173...
Enter some string: abcd dcba
Input string: abcd
Substring with length 4 not found
```

Ввод:

123567890qwertyuiopasdfghjklzxcvbnmQWERTYUIOPASDFHJKLZXCVBNM!@#\$
%^&*()_+~{}[]:;''/?.,<>dcba

Обработывает различные ASCII символы

```
C:\Users\5ф123\Downloads\fasmw17325\Projects\MicroProject.EXE
Enter some string: 123567890qwertyuiopasdfghjklzxcvbnmQWERTYUIOPASDFHJKLZXCVBNM!@#$%^&*()_+~{}[]:;''/?.,<>dcba
Input string: 123567890qwertyuiopasdfghjklzxcvbnmQWERTYUIOPASDFHJKLZXCVBNM!@#$%^&*()_+~{}[]:;''/?.,<>dcba
Substring was found: "dcba" at position 87
```

6. Источники

1. Строковые операции. <http://flatassembler.narod.ru/fasm.htm#2-1-8>

Приложение 1. Текст задания

Разработать программу, находящую в заданной ASCII-строке первую слева направо последовательность N символов, каждый элемент которой определяется по условию "меньше предшествующего" (N=4)

Приложение 2. Код программы

format PE console

entry start

include 'win32a.inc'

```
;-----
section '.data' data readable writable

N      dd  4
idx     dd  ?
strInput  rb  1001
strOutput rb  5 ; N + 1
strScanStr db  '%s', 0
strAskInput db  'Enter some string: ', 0
strShowInp db  'Input string: %s', 10, 0
strNotFound db  'Substring with length %d not found', 10, 0
strFound   db  'Substring was found: "%s" at position %d', 10, 0

;-----
```

section '.code' code readable executable

start:

```
invoke printf, strAskInput
add esp, 4
invoke scanf, strScanStr, strInput
add esp, 8
invoke printf, strShowInp, strInput
add esp, 8
```

```
stdcall strFindSubstring, strInput, [N], idx
add esp, 12
```

```
cmp dword [idx], -1 ; if idx == -1. strFindSubstring didn't find such substring
je notFound
```

```
stdcall getSubstring, strInput, strOutput, [N], [idx]
add esp, 16
```

```
invoke printf, strFound, strOutput, [idx]
add esp, 8
jmp finish
```

notFound:

```
invoke printf, strNotFound, [N]
add esp, 8
```

finish:

```
call [getch]
```

```
push 0
call [ExitProcess]
```

; The procedure finds a substring in which each element less than previous one

; void strFindSubstring(string str, int len, int& idx)

; str - input message

; len - substring length

; idx - returned value, first index of substring or -1 if substring wasn't found

strFindSubstring:

```
xor al, al ; delimiter = '\0'
```

```
mov esi, [esp+4] ; using str[0] as source
```

```
cmp [esi], al ; if length(str) == 0
```

```
je substrNotFound ; not found
```

```
mov edi, [esp+4] ; ...
```

```
inc edi ; using str[1] as destination
```

```
cmp [edi], al ; if length(str) == 1
```

```
je substrNotFound ; not found
```

```

        mov     ecx, -1        ; ecx < 0 for loop
        mov     ebx, [esp+8]    ; ebx = N
        mov     edx, 1         ; curLen = 1
loopStrCmp:
        cmpsb                     ; if str[i] > str[i+1]
        jg      lessPrev
        mov     edx, 1         ; else curLen = 1
        jmp     endComp

lessPrev:
        inc     edx            ; curLen++
        cmp     edx, ebx       ; if curLen == N
        je      substrFound    ; str found

endComp:
        cmp     [edi], al      ; while str[i+1] != '\0'
        loopne  loopStrCmp
        jmp     substrNotFound ; loop didn't find a substring
substrFound:
        neg     ecx            ; ecx = -ecx. Now ecx contains last index of a substring
        sub     ecx, ebx       ; ecx -= N
        inc     ecx            ; ecx++. Now ecx contains first index of a substring

        mov     ebx, [esp+12]
        mov     [ebx], ecx     ; idx = ecx
        ret

substrNotFound:
        mov     ebx, [esp+12]
        mov     dword [ebx], -1 ; idx = -1
        ret

```

```

;-----
; The procedure finds a substring in which each element less than previous one
; void getSubstring(string strInp, string strOut, int len, int idx)
; strInp - input message
; strOut - output message
; N - substring length
; idx - first index of substring in input message
getSubstring:

```

```

        mov     esi, [esp+4]    ; ...
        add     esi, [esp+16]   ; using strInp[idx] as source

        mov     edi, [esp+8]    ; using strOut as destination
        mov     ecx, [esp+12]   ; counter = N
loopCpy:
        movsb                     ; strOut[i] = strInp[idx+i]
        loop    loopCpy        ; counter--; while counter > 0

```

ret

section '.idata' import data readable

library kernel, 'kernel32.dll',\

msvcrt, 'msvcrt.dll',\

user32, 'USER32.DLL'

include 'api\user32.inc'

include 'api\kernel32.inc'

import kernel,\

ExitProcess, 'ExitProcess'

include 'api\kernel32.inc'

import msvcrt,\

printf, 'printf',\

scanf, 'scanf',\

getch, '_getch'