

ECE3120: Computer Systems

Chapter 4: Strings

Manjeera Jeedigunta

<http://blogs.cae.tnitech.edu/msjeedigun21>

Email: msjeedigun21@tnitech.edu

Tel: 931-372-6181, Prescott Hall 120

-
- Prev...
 - Indexable data structures
 - Today...
 - Strings
 - Character and word counting
 - String Insertion
 - Data Conversion
 - Introduction to subroutines

Strings

- A sequence of characters terminated by a NULL (ASCII code 0)
- A number in the computer is represented as a binary number.
- Basic applications by manipulating strings
 - Character or word counting
 - String insertion
 - Word matching
 - Data Conversion

Character and Word Counting

- A string is terminated by the NULL character.
- A new word is identified by skipping over the white space characters.
- When a new word is identified, it must be scanned through before the next word can be identified.

Example 4.7 Write a program to count the number of characters and words contained in a given string.

Solution:

```
tab      equ      $09      ;ASCII Code
sp       equ      $20
cr       equ      $0D
lf       equ      $0A
         org      $1800
char_cnt rmb      1
word_cnt rmb      1
string_x fcc      "this is a strange test string to count chars and words."
         fcb      0
         org      $1000
         ldx      #string_x
         clr      char_cnt
         clr      word_cnt

string_lp ldab     1,x+      ; get one character and move string pointer
         lbeq     done      ; is this the end of the string?
         inc      char_cnt
```

; the following 8 instructions skip white space characters between words

cmpb	#sp
<hr/>	
beq	string_lp
cmpb	#tab
beq	string_lp
cmpb	#cr
beq	string_lp
cmpb	#lf
beq	string_lp

; a non-white character is the start of a new word

	inc	word_cnt	
wd_loop	ldab	1,x+	<i>; get one character and move pointer</i>
	beq	done	
	inc	char_cnt	

; the following 8 instructions check the end of a word

cmpb	#sp
lbeq	string_lp
cmpb	#tab
lbeq	string_lp
cmpb	#cr
lbeq	string_lp
cmpb	#lf
lbeq	string_lp
bra	wd_loop

done	swi
	end

String Insertion

- The pointers to the string and the substring to be inserted are given.
- The insertion point is given.
- The procedure is given in Figure 4.6.

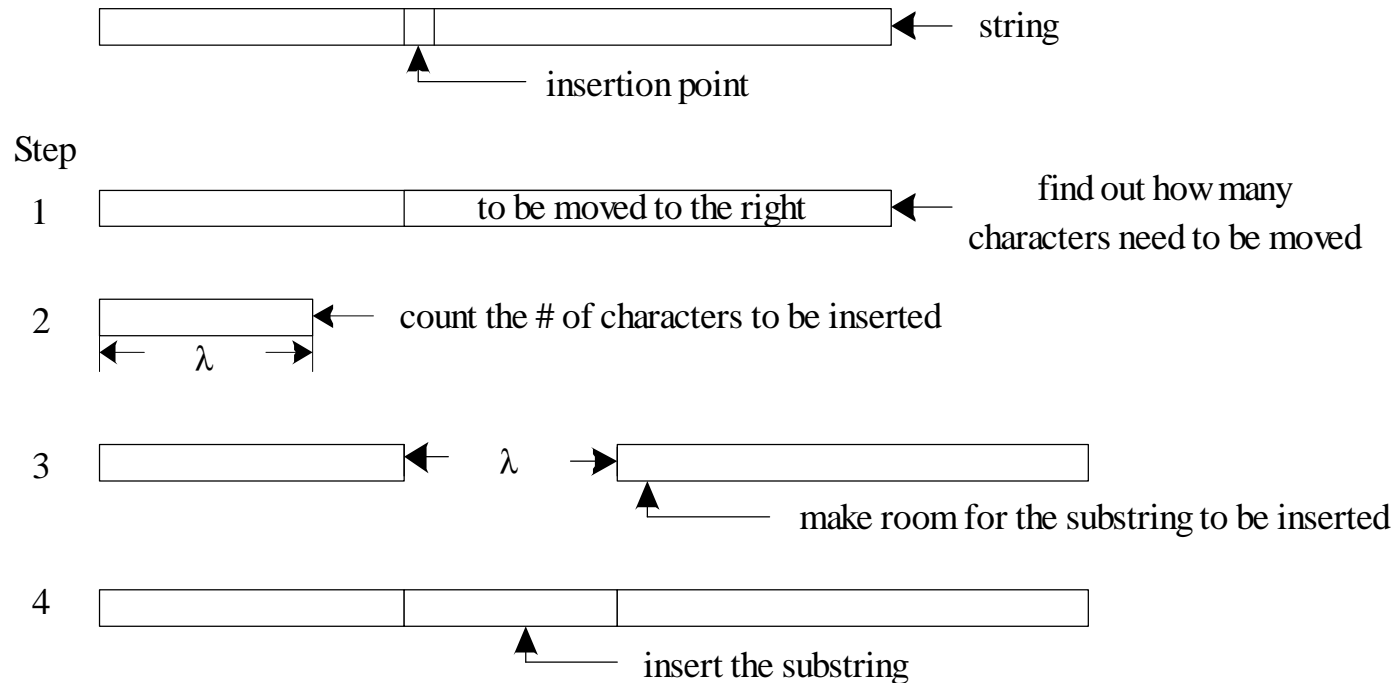


Figure 4.6 Major steps of substring insertion

Example 4.9 Write a program to implement the string insertion algorithm.

```

                                org    $1800
ch_moved    rmb    1
char_cnt    rmb    1
sub_strg     fcc    "the first and most famous "
                                fcb    0
string_x     fcc    "Yellowstone is national park."
                                fcb    0
offset       equ    15
ins_pos      equ    string_x+offset ; insertion point
                                org    $1000
; the next 7 instructions count the number of characters to be moved
                                ldaa   #1
                                staa   ch_moved
                                ldx    #ins_pos      ; use x to point to the insertion point
cnt_moved    ldaa   1,x+
                                beq    cnt_chars
                                inc    ch_moved
                                bra    cnt_moved
cnt_chars    dex                    ; subtract 1 from x so it points to the NULL character
                                ldy    #sub_strg     ; use y as a pointer to the substring
                                clr    char_cnt
```


; the following 3 instructions count the move distance

```
char_loop ldab    1,y+
          beq     mov_loop
          inc     char_cnt
          bra     char_loop
mov_loop  tfr     x,y          ; make a copy of x in y
          ldab    char_cnt
          aby     ; compute the copy destination
          ldab    ch_moved     ; place the number of characters to be moved in B
again     movb    1,x-,1,y-
          dbne    b,again      ; make room for insertion
          ldx     #ins_pos     ; set up pointers to prepare insertion
          ldy     #sub_strg    ;
          ldab    char_cnt
insert_lp movb    1,y+,1,x+
          dbne    b,insert_lp
          swi
          end
```

Word Matching

- More detail flowchart is on Page 139

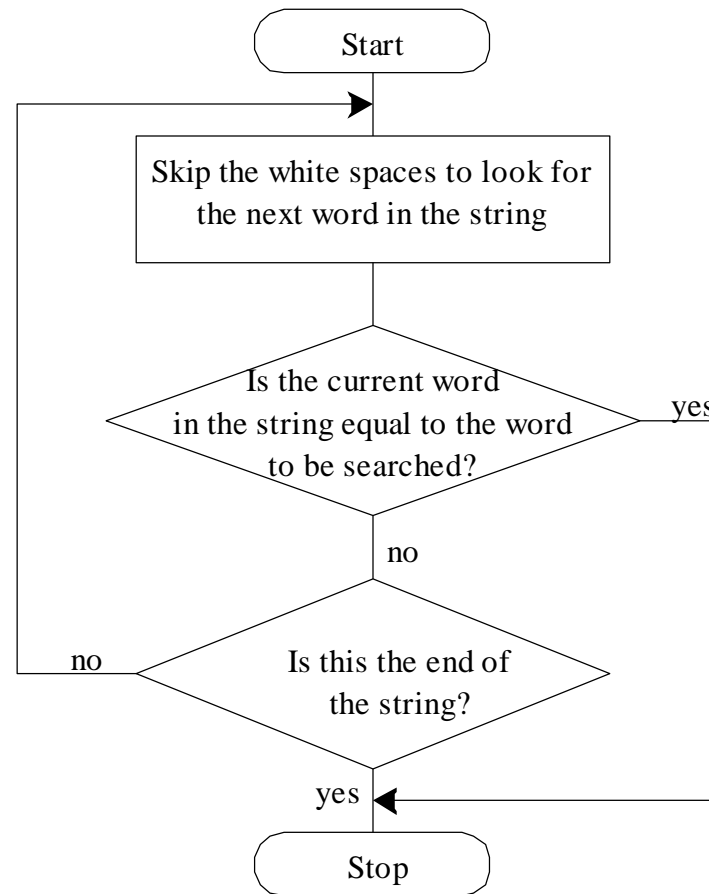


Figure 4P.8 Flowchart of the word search program

Program to search for a given word

tab	equ	\$09	; ASCII code of tab
sp	equ	\$20	; ASCII code of space character
cr	equ	\$0D	; ASCII code of carriage return
lf	equ	\$0A	; ASCII code of line feed
period	equ	\$2E	; ASCII code of period
comma	equ	\$2C	; ASCII code of comma
semicolon	equ	\$3B	; ASCII code of semicolon
exclamation	equ	\$21	; ASCII code of exclamation
null	equ	\$0	; ASCII code of NULL character
	org	\$1800	
match	rmb	1	
	org	\$1000	
	clr	match	

Initialization

	ldx	#string_x
--	-----	-----------

loop	ldab	1,x+
------	------	------

; the following 10 instructions skip white spaces to look for the next word in string_x

tstb

beq done

cmpb #sp

beq loop

cmpb #tab

beq loop

cmpb #cr

beq loop

cmpb #lf

beq loop

; the first nonwhite character is the beginning of a new word to be compared

	ldy	#word_x	
	ldaa	1,y+	
next_ch	cba		
	bne	end_of_wd	
	cmpa	#null	; check to see if the end of word is reached
	beq	matched	;
	ldaa	1,y+	; get the next character from the word
	ldab	1,x+	; get the next character from the string
	bra	next_ch	

; the following 10 instructions check to see if the end of the given word is reached

end_of_wd	cmpa	#null
	bne	next_wd
	cmpb	#cr
	beq	matched
	cmpb	#lf
	beq	matched
	cmpb	#tab
	beq	matched
	cmpb	#sp
	beq	matched
	cmpb	#period
	beq	matched
	cmpb	#comma
	beq	matched
	cmpb	#semicolon
	beq	matched
	cmpb	#exclamation
	beq	matched

; the following 11 instructions skip the remaining characters in the unmatched word

next_wd	ldab	1,x+
	beq	done
	cmpb	#cr
	lbeq	loop
	cmpb	#lf
	lbeq	loop
	cmpb	#tab
	lbeq	loop
	cmpb	#sp
	lbeq	loop
	bra	next_wd
matched	ldab	#1
	stab	match
done	swi	
string_x	fcc	"This string contains certain number of words to be matched."
	fcbl	0
word_x	fcc	"This"
	fcbl	0
	end	

Strings

- A sequence of characters terminated by a NULL (ASCII code 0)
- A number in the computer is represented as a binary number.
- Basic applications by manipulating strings
 - Character or word counting
 - String insertion
 - Word matching
 - Data Conversion

Data Conversion

Example 4.4 Write a program to convert the unsigned 8-bit binary number in accumulator A into BCD digits terminated by a NULL character. Each digit is represented in ASCII code.

Solution:

1. 8 bit number → 0 to 255
2. Include a null to terminate (say 255.)
3. 4 bytes are needed to hold the converted BCD digits.
4. Repeated division by 10 method is used to retrieve individual digits.
5. Conversion to ASCII → add \$30 to BCD digit

```
test_dat    equ    220
            org    $1000
out_buf     rmb    4
temp        rmb    2
            org    $1500    ; starting address of the program
            ldaa   #test_dat ; load the test data
            ldy    #out_buf
            tab     ; transfer the 8-bit value in B

; check to see if the number has only one digit
            cmpb   #9
            bhi    chk_99    ; section that checks if it has more than 2 digits
            addb   #$30        ; convert the digit into ASCII code
            stab   0,y         ; save the code and increment the pointer
            clr    1,y         ; terminated the string with NULL
            jmp    done
chk_99      clra
```

; check to see if the number has two digits

	cmpb	#99	; is the number greater than 99
	bhi	three_dig	; if yes, the string has three digits
	ldx	#10	
	idiv		; x ← Q(remaining digit) , D ← R (lower digit)
	addb	#\$30	; convert the lower digit
	stab	1,y	; store the lowest digit
	xgdx		
	addb	#\$30	
	stab	0,y	; save the upper digit
	clr	2,y	; terminated the string with NULL
	bra	done	
three_dig	ldx	#10	
	idiv		
	addb	#\$30	
	stab	2,y	; save the least significant digit
	xgdx		; swap the quotient to D(2 digits)
	ldx	#10	
	idiv		
	addb	#\$30	
	stab	1,y	; save the middle digit
	xgdx		; swap the hundred's digit to B (1 digit left)
	addb	#\$30	
	stab	0,y	; save the ASCII code of the highest digit
	clr	3,y	; terminate the string with NULL
done	swi		
	end		

Next...

- Subroutines