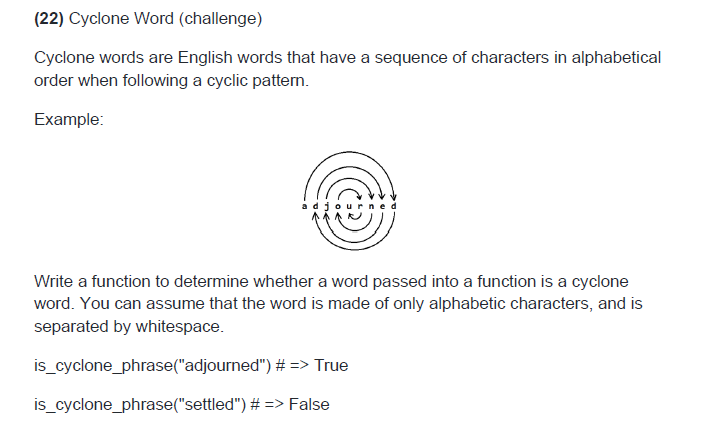
Computer Architecture Lab – 147798

Lecturer: Hoàng Văn Hiệp

Mid-term 2023.2 Report

Nguyễn Khánh Nam - 20225749

Bài 22:



* Cách thực hiện:

+ Đọc từng character được nhập vào.

+ Sử dụng 2 con trỏ địa chỉ trỏ vào đầu và cuối chuỗi character được nhập.

+ Thực hiện dịch chuyển địa chỉ 2 con trỏ đầu - cuối chuỗi character lần lượt so sánh từng character trong chuỗi theo hình xoắn ốc từ ngoài vào chữ cái ở giữa chuỗi cùng chiều kim đồng hồ.

+ So sánh thứ tự chữ cái tăng dần theo bảng ASCII trả kết quả TRUE, ngược lại thì trả kết quả FALSE.

* Thanh ghi:

+ $sp: chứa địa chỉ đỉnh Stack lưu từng character từ word được nhập vào. Giảm 1 byte mỗi lần lưu 1 giá trị word (a-z và A-Z). Chương trình mặc định đọc và chuyển hết về Lower-case để kiểm tra Cyclone Word.

+ $s0: chứa địa chỉ character tận cùng bên phải ở lần so sánh 1 và tiếp theo sau lần so sánh thứ LẺ sẽ chứa giá trị địa chỉ con trỏ $s1 dịch xuống 1 byte.

+ $s1: chứa địa chỉ character tận cùng bên trái ở lần so sánh 1 và tiếp theo sau lần so sánh thứ CHẴN sẽ chứa giá trị địa chỉ con trỏ $s0 dịch lên 1 byte.

+ $v0: chứa giá trị từng character được nhập vào. Dùng để kiểm tra điều kiện nhập vào.

+ $t1: đếm số character trong word.

+ $s2 = $t1 – 1: đếm số lần so sánh kiểm tra Cyclone Word.

* Chương trình con:

+ check\_upper và check\_lower: kiểm tra chuỗi character nhập vào nằm trong khoảng (A-Z) và (a-z),

+ check\_cyclone\_word: chương trình con chính của chương trình. Thay đổi giá trị địa chỉ được trỏ bởi $s0 và $s1 và kiểm tra Cyclone Word.

* Source code:

.data

Message\_error\_input: .asciiz "\nError input\nWord only!"

Message\_false: .asciiz "\nFALSE"

Message\_true: .asciiz "\nTRUE"

space: .asciiz " "

.text

addi $s1, $sp, -1

addi $s0, $sp, -1

li $t1, 0 #count\_characters cnt = 0 $t1

input\_loop: # Each char

addi $sp, $sp, -1

li $v0, 12

syscall

beq $v0, 0x20, stop\_input

nop

beq $v0, 0x0a, stop\_input

nop

check\_upper:

blt $v0, 0x41, end\_error #Check Upper A-Z

nop

bgt $v0, 0x5A, check\_lower

nop

addi $v0, $v0, 32 #Chuyển về lower-case

j skip\_check\_lower

nop

check\_lower:

blt $v0, 0x61, end\_error #Check lower a-z

nop

bgt $v0, 0x7A, end\_error

nop

skip\_check\_lower:

sb $v0, 0($sp)

addi $t1, $t1, 1 # cnt++

addi $s0, $s0, -1 # string[i++]

j input\_loop

stop\_input:

addi $s2, $t1, -1 # $s2 = cnt - 1

# - check for number of comparisons

addi $s0, $s0, 1 # = last character of the word

check\_cyclone\_word:

lbu $t2, 0($s1) # first half character

lbu $t3, 0($s0) # last half character

slt $t4, $t3, $t2 # compare: $t4 = $t2 <= $t3 ? continue : end

beq $t4, 1, end\_not\_cyclone\_word

nop

addi $s2, $s2, -1 # number of comparisons - 1 after each comparison

beq $s2, 0, end\_check\_cyclone\_word # number of comparisons = 0 then stop check

beq $ra, $0, skip # Check $ra whether returns or not->skip

nop

jr $ra

skip:

# Sau so sánh lần thứ 1, so sánh tiếp những lần tiếp theo

# Sau lần so sánh thứ lẻ,

# địa chỉ s1 đang trỏ ở nửa TRÁI word cần check

# địa chỉ s0 đang trỏ ở nửa PHẢI word cần check

addi $t5, $s0, 0 # temp = s0(Odd)

addi $s0, $s1, -1 # s0(Even) = s1(Odd) + 1

addi $s1, $t5, 0 # s1(Even) = s0(Odd)

# phải giữ nguyên địa chỉ s0 của chữ cái STT nhỏ hơn

# để so sánh lần thứ 2 sau lần so sánh thứ 1

# nên cần jal so sánh luôn lần thứ chẵn (từ lần thứ 2 trở đi)

jal check\_cyclone\_word

nop

add $ra, $0, $0 # Reset $ra after each Even - comparison

# Sau lần so sánh thứ CHẴN,

# địa chỉ s1 đang trỏ ở nửa PHẢI word cần check

# địa chỉ s0 đang trỏ ở nửa TRÁI word cần check

add $t5, $s0, 0 # temp = s0(Even)

addi $s0, $s1, 1 # s0(Odd) = s1(Even) - 1

addi $s1, $t5, 0 # s1(Odd) = s0(Even)

j check\_cyclone\_word

nop

end\_check\_cyclone\_word:

li $v0, 4

la $a0, Message\_true

syscall

j end

nop

end\_not\_cyclone\_word:

li $v0, 4

la $a0, Message\_false

syscall

j end

nop

end\_error:

li $v0, 4

la $a0, Message\_error\_input

syscall

j end

nop

end:

* Kết quả:

+ ‘adjourned’:

A screenshot of a computer

Description automatically generated

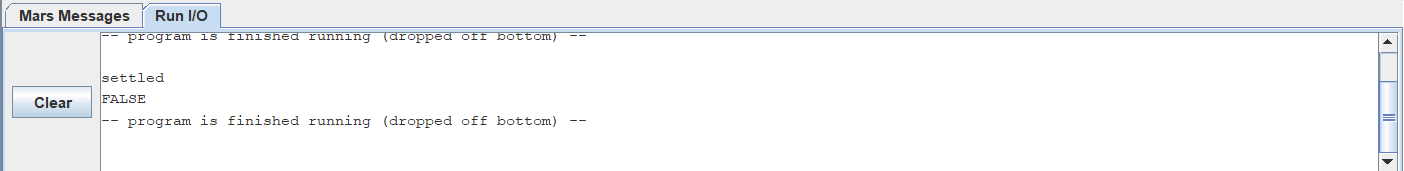
A white screen with blue and black text

Description automatically generated with medium confidence

+ ‘settled’:

A screenshot of a computer

Description automatically generated



Bài 23:



* Cách thực hiện:

+ Sử dụng thuật toán đệ quy theo công thức:

+ Nhập vào n và k. Sử dụng Stack để lưu giá trị n, k, kết quả và địa chỉ jump return của từng bước đệ quy

+ Dịch chuyển thanh ghi con trỏ $sp từng 4 byte để lưu và xuất giá trị n, k, kết quả và địa chỉ return từng bước đệ quy.

+ Chương trình kiểm tra những trường hợp đặc biệt của tính tổ hợp và đưa ra kết quả ngay lập tức. Nếu không sẽ thực hiện gọi đệ quy để tính toán.

* Thanh ghi:

+ $s0: chứa n

+ $s1: chứa k

+ $sp: chứa địa chỉ đỉnh stack lưu n, k, kết quả C(k – 1, n - 1) và địa chỉ $ra. Được reset lại sau mỗi lần tính được đệ quy C(k-1, n-1) + C(k,n-1).

* Chương trình con:

+ Base\_CKN: dùng để đệ quy chứa trường hợp nền của công thức tính tổ hợp đệ quy trên

+ findCKN: Thực hiện từng bước đệ quy thay đổi giá trị k và n, gọi lại chương trình con Base\_CKN để kiểm tra trường hợp nền. Trả kết quả tính tổ hợp cuối cùng.

* Source code:

.data

    input\_msg1: .asciiz "Input n: "

    input\_msg2: .asciiz "Input k: "

    output\_res: .asciiz "C(k,n) = "

    error\_msg1: .asciiz "wrong input"

    result: .word 0

.text

#----------------------INPUT------------------------

    li $v0, 4

    la $a0, input\_msg1

    syscall

    li $v0, 5

    syscall

    addi $s0, $v0, 0  #s0 = n

    li $v0, 4

    la $a0, input\_msg2

    syscall

    li $v0, 5

    syscall

    addi $s1, $v0, 0  #s1 = k

#----------------CHECKING----------------------------------

    blt $s0, $0, error\_input1   # n < 0 -> error

    nop

    blt $s1, $0, error\_input1   # k < 0 -> error

    nop

    beq $s0, $0, special\_case   # n = 0 ->special case

    nop

    bne $s1, 1, skip1       # If k = 1 AND n > 0 -> C(k,n) = n

    nop

    addi $s3, $s0, 0

    j EXIT\_result

    skip1:

special\_case:

    bne $s0, $s1, skip2     # If k = n AND n >= 0 -> C(k,n) = 1

    nop

    addi $s3, $0, 1

    j EXIT\_result

    skip2:

    bne $s1, 0, skip3       # k = 0 AND n > 0 -> C(k,n) = 1

    nop

    addi $s3, $0, 1

    j EXIT\_result

    skip3:

    slt $t0, $s0, $s1       #Check n < k -> ERROR

    beq $t0, 1, error\_input1

    nop

#-------------Recursive Calculation of C(k,n)-----------------

#print result

    jal findCKN

    move $s3, $v0

    sw $s3, result

    #display result

    li $v0, 4

    la $a0, output\_res

    syscall

    li $v0, 1

    lw $a0, result

    syscall

    #end program

    li $v0, 10

    syscall

    # v0 = result of the function

    #

Base\_CKN:

    seq $t0, $s1, $0

    seq $t1, $s0, $s1

    add $t2, $t0, $t1

    blt $t2, 1, continue1   # If (k == 0 || k == n)

    nop

    addi $v0, $0, 1     # return 1

    jr $ra

    continue1:

    bne $s1, 1, continue2   # If (k == 1)

    nop

    addi $v0, $s0, 0        # return n

    jr $ra

    continue2:

    j findCKN       # return C(k - 1, n - 1) + C(k,n-1)

findCKN:

    subu $sp, $sp, 16

    sw $ra 0($sp)

    sw $s1, 4($sp)   #save k

    sw $s0, 8($sp)   #save n

    addi $s1, $s1, -1   # k - 1

    addi $s0, $s0, -1   # n - 1

    jal Base\_CKN     # -> C(k - 1, n - 1)

    lw $s1, 4($sp)   #restore k

    lw $s0, 8($sp)   #restore n

    sw $v0, 12($sp) #save C(k - 1 , n - 1)

    addi $s0, $s0, -1   # n - 1

    jal Base\_CKN     # -> C(k , n - 1)

    lw $t7, 12($sp)     #restore C(k-1,n-1)

    add $v0, $t7, $v0   #return C(k-1,n-1) + C(k,n-1)

    lw $ra, 0($sp)       #restore $ra

    add $sp, $sp, 16        #

    jr $ra

#--------------------------print out----------------------

error\_input1:

    li $v0, 4

    la $a0, error\_msg1

    syscall

    j EXIT

EXIT\_result:

    li $v0, 4

    la $a0, output\_res

    syscall

    li $v0, 1

    addi $a0, $s3, 0

    syscall

EXIT:

    li $v0, 10

    syscall

* Kết quả:

+ C(12, 3):

A screenshot of a computer

Description automatically generated

+ C(30, 5):

A screenshot of a computer

Description automatically generated