

ĐẠI HỌC BÁCH KHOA HÀ NỘI
TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG



BÁO CÁO

Bài thực hành giữa kỳ

Học phần: Thực hành kiến trúc máy tính

Giảng viên hướng dẫn: Lê Bá Vui

Sinh viên thực hiện: Lê Đình Tuyên - 20194715

Mã lớp: 130938

Hà Nội, tháng 5 năm 2022

A-3

- Code:

```
.data
A: .space 100          # Mang A co do dai 25 luu tru cac
chu so cua so nguyen N
Message: .asciiiz "Nhap N (Nhap so co tu hai chu so tro
len va nho hon 25 chu so): "
Message2: .asciiiz "N co it nhat 2 chu so trung nhau!"
Message3: .asciiiz "N khong co chu so nao trung nhau!"
.text
main:

input_N:
    li $v0, 51
    la $a0, Message
    syscall

    beq $a1, -1, input_N    # nhap lai neu khong phai so
nguyen
    beq $a1, -3, input_N    # nhap lai neu khong phai so
nguyen
    beq $a1, -2, exit       # thoat khi nguoi dung an
cancel
    blt $a0, 10, input_N    # nhap lai neu nguoi dung
nhap so nho hon 10

    add $a1, $a0, 0

    li $t0, 0               # i la index trong mang A
    li $s0, 0               # do dai mang A hien tai

check:
    li $t3, 10
loop1: beq $a1, 0, end_loop1
    div $a1, $t3            # chia N cho 10 de lay cac chu so
    mflo $a1                # lay phan thuong
    mfhi $t1                # lay phan du
```

```

    li $t0, 0          # i = 0
loop2:  beq $t0, $s0, end_loop2 # tim xem chu so hien
tai da co trong mang A chua
        # neu chua co trong mang A thi them vao
        # cuoi mang A va tang do dai mang A len 1
        # neu da co trong mang A thi so nguyen
        # N co it nhat 2 chu so giong nhau
    sll $t2, $t0, 2      # 4*i
    lw $t2, A($t2)       # A[i]
    beq $t1, $t2, N_co_2cs_trung_nhau # neu chu so hien
tai dang xet(phan du cua phep chia N voi 10) da co
trong mang A
        # thi in ra N co 2 chu so trung
        # nhau
    add $t0, $t0, 1
    j loop2
end_loop2:
    sll $t4, $s0, 2      # neu chu so hien tai dang xet
chua co trong mang A
        # thi them vao cuoi mang A va tang do
        # dai mang A len 1 ($s0++)
    sw $t1, A($t4)
    add $s0, $s0, 1
    j loop1
end_loop1:
N_ko_co_cs_trung_nhau:      # in ra N ko co chu so
giong nhau
    li $v0, 4
    la $a0, Message3
    syscall
    j exit
N_co_2cs_trung_nhau:      # in ra N co it nhat 2 chu so
giong nhau
    li $v0, 4
    la $a0, Message2
    syscall
exit:

```

- Cách thực hiện:

+ B1: Nhập số nguyên N từ bàn phím có kiểm tra các trường hợp ngoại lệ (nhập sai định dạng số nguyên, ấn thoát, nhập số nhỏ hơn 10,...)

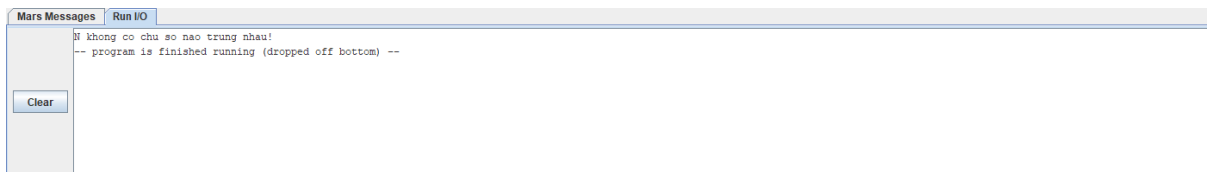
+ B2: Sử dụng mảng A lưu các chữ số khác nhau của số nguyên N

+ B3: Lấy các chữ số của số nguyên N bằng cách sử dụng vòng lặp chia N cho 10 rồi gán thương cho N và phần dư ta sẽ xem xét bên trong mảng A đã tồn tại chữ số này hay chưa. Nếu chưa ta sẽ thêm vào mảng A, nếu đã tồn tại in ra N có ít nhất 2 chữ số giống nhau. Tiếp tục lặp $N / 10$ cho đến khi được thương bằng 0.

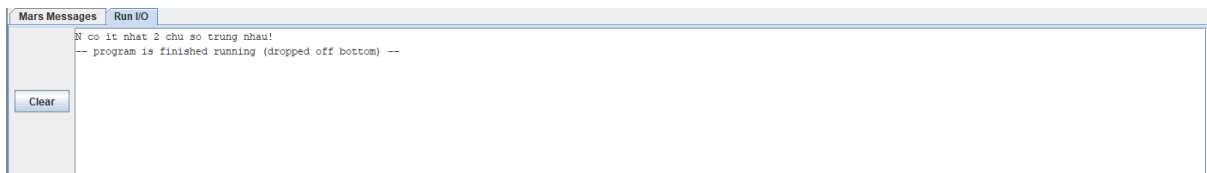
+ B4: Sau bước 3 mà N không có 2 chữ số giống nhau thì in ra N không có 2 chữ số giống nhau.

- Hình ảnh minh họa:

+ Với N = 102



+ Với N = 65263



B-2

- Code :

```
#Bai b2
.data
A: .space 400
Message1: .ascii "Nhap so phan tu: "
Message2: .ascii "Nhap phan tu cho mang: "
.text
main:
    la $s1, A          # load address of array A

    li $v0, 51
    la $a0, Message1
    syscall
```

```

    beq $a1, -1, main    # nhập lại nếu không phải số
nguyen
    beq $a1, -3, main    # nhập lại nếu không phải số
nguyen
    beq $a1, -2, exit    # thoát khi người dùng ấn cancel

    add $s0, $zero, $a0  # N - số phần tử của mảng

    li $t0, 0            # i = 0

input_array:
    bge $t0, $s0, sort
input_element:
    li $v0, 51
    la $a0, Message2
    syscall

    beq $a1, -1, input_element # nhập lại nếu không phải
so nguyen
    beq $a1, -3, input_element # nhập lại nếu không phải
so nguyen
    beq $a1, -2, exit          # thoát khi người dùng ấn
cancel

    sll $t1, $t0, 2           # $t1 = $t0 << 2 ( = 4*i)
    add $t1, $s1, $t1         # A + 4i

    sw $a0, 0($t1)            # lưu số vừa nhập vào mảng A

    add $t0, $t0, 1           # i++
    j input_array
sort:
    li $t0, 0
loop1: beq $t0, $s0, end_loop1

    sll $t1, $t0, 2           # $t1 = $t0 << 2 ( = 4*i)
    add $t1, $s1, $t1         # A + 4i

```

```

    lw $a0, 0($t1)      # current_max = A[i]
    bgtz $a0, findMaxAndSwap
    add $t0, $t0, 1      # i++
    j loop1

findMaxAndSwap:
    add $t2, $t0, 1      # j = i + 1
loop2:    beq $t2, $s0, end_loop2

    sll $t3, $t2, 2      # $t3 = $t2 << 2 ( = 4*j)
    add $t3, $s1, $t3    # address(A + 4j) =
address(A[j])

    lw $a1, 0($t3)      # A[j]
    bltz $a1, continue_loop2 # A[j] < 0 => continue
compare A[i] with A[j+1]

    slt $t4, $a0, $a1    # current_max < A[j] ?
    beqz $t4, continue_loop2

    sw $a1, 0($t1)      # store value of A[j] to A[i]
    sw $a0, 0($t3)      # store value of A[i] to A[j]

    add $a0, $a1, 0      # max = A[j]
continue_loop2:
    add $t2, $t2, 1      # j = j + 1
    j loop2
end_loop2:
    add $t0, $t0, 1      # i++
    j loop1
end_loop1:
exit:

```

- Cách thực hiện:

- + B1: Nhập vào mảng và kiểm tra các trường hợp ngoại lệ

+ B2: Ta chọn phần tử lớn hơn 0 đầu tiên làm current_max. Sau đó lặp qua các phần tử trong mảng A, phần tử nào lớn hơn 0 và lớn hơn current_max thì đổi chỗ current_max với phần tử đó rồi gán current_max bằng phần tử đó. Sau khi lặp hết mảng A, ta lại chọn phần tử lớn hơn 0 đầu tiên trong phần chưa được sắp xếp của mảng A làm current_max và lại tiếp tục như trên cho đến khi các số nguyên dương trong A đã được sắp xếp hoàn toàn.

- Hình ảnh minh họa:

+ Với mảng nhập vào là [10; -3; 6; 11; -1]:

Data Segment										
Address	Value (+0)	Value (+4)	Value (+8)	Value (+12)	Value (+16)	Value (+20)	Value (+24)	Value (+28)		
268500992	11	-3	10	6	-1	0	0	0		
268501024	0	0	0	0	0	0	0	0		
268501056	0	0	0	0	0	0	0	0		
268501088	0	0	0	0	0	0	0	0		
268501120	0	0	0	0	0	0	0	0		
268501152	0	0	0	0	0	0	0	0		
268501184	0	0	0	0	0	0	0	0		
268501216	0	0	0	0	0	0	0	0		
268501248	0	0	0	0	0	0	0	0		
268501280	0	0	0	0	0	0	0	0		
268501312	0	0	0	0	0	0	0	0		
268501344	0	0	0	0	0	0	0	0		
268501376	0	0	0	0	0	0	0	0		
268501408	1749942304	1881174113	544104808	1663071604	1885431886	544174880	1851877488	980775968		
268501440					1830842216	979856993	32			

+ Với mảng nhập vào là [-1;150;190;170;-3;-2;160;180]:

Text Segment					Labels	
Bkpt	Address	Code	Basic	Source	Label	Address
	4194304	0x3c011001	lui \$1,4097	8: la \$a1, A # load address of array A	b2.asm	
	4194308	0x34310000	ori \$17,\$1,0		main	4194304
	4194312	0x24020039	addiu \$2,\$0,\$1	10: li \$v0, \$1	input_array	4194360
	4194316	0x3c011001	lui \$1,4097	11: la \$a0, Message1	input_element	4194368
	4194320	0x34240190	ori \$4,\$1,400		sort	4194428
	4194324	0x0000000c	syscall	12: syscall	loop1	4194432
	4194328	0x2001ffff	addi \$1,\$0,-1	14: beq \$a1, -1, main # nhập lại neu không phải số nguyên	findMaxAndSwap	4194460
	4194332	0x1025ffff	beq \$1,\$5,-8		loop2	4194464
	4194336	0x2001ffff	addi \$1,\$0,-3	15: beq \$a1, -3, main # nhập lại neu không phải số nguyên	continue_loop2	4194504
	4194340	0x1025ffff	beq \$1,\$5,-10		end_loop2	4194512
	4194344	0x2001ffff	addi \$1,\$0,-2	16: beq \$a1, -2, exit # thoát khi người dùng ấn cancel	end_loop1	4194520
	4194348	0x1025002a	beq \$1,\$5,42		exit	4194520
	4194352	0x00048020	addi \$16,\$0,\$4	18: add \$a0, \$zero, \$a0 # N - số phần tử của mảng	A	268500992
	4194356	0x24080000	addiu \$8,\$0,0	20: li \$t0, 0 # i = 0	Message1	268501392
	4194360	0x0110082a	slt \$1,\$8,\$16	23: bge \$t0, \$a0, sort	Message2	268501410
	4194364	0x1020000f	beq \$1,\$0,15			
	4194368	0x24020039	addiu \$2,\$0,\$1	25: li \$v0, \$1		

Data Segment										
Address	Value (+0)	Value (+4)	Value (+8)	Value (+12)	Value (+16)	Value (+20)	Value (+24)	Value (+28)		
268500992	-1	190	180	170	-3	-2	160	150		
268501024	0	0	0	0	0	0	0	0		
268501056	0	0	0	0	0	0	0	0		
268501088	0	0	0	0	0	0	0	0		
268501120	0	0	0	0	0	0	0	0		
268501152	0	0	0	0	0	0	0	0		
268501184	0	0	0	0	0	0	0	0		
268501216	0	0	0	0	0	0	0	0		
268501248	0	0	0	0	0	0	0	0		
268501280	0	0	0	0	0	0	0	0		
268501312	0	0	0	0	0	0	0	0		
268501344	0	0	0	0	0	0	0	0		
268501376	0	0	0	0	0	0	0	0		
268501408	1749942304	1881174113	544104808	1663071604	1885431886	544174880	1851877488	980775968		
268501440					1830842216	979856993	32			

C-1

- Code:

```
# C-1

.data
    tmp: .space 32          # save current word that we
                             # have just read
    input: .space 100       # save user input
    start: .asciiz "\n\n----- Let's start -----
    --"
```

```

request: .asciiiz "\nInput: "
message: .asciiiz "\nShortest words is: "
newline: .asciiiz "\n\t"

.text
    li    $v0, 4           # print_string
    la    $a0, start       # start message
    syscall

init: li    $v0, 4           # print_string call number
    la    $a0, request
    syscall

    li    $v0, 8
    la    $a0, input       # pointer to string in memory
    la    $a1, 100
    syscall

    li    $s0, 0           # i = 0 run through all text
    li    $s2, 0           # j = 0 is the index of tmp
    li    $s4, 9999        # current shortest length of
result
    li    $s6, 0           # k = 0: index to clear tmp
    la    $a2, tmp         # load tmp address

main: jal    checkNonAlpha  # loop through input
string

    li    $v0, 4           # print_string
    la    $a0, message     # message to print shortest
words
    syscall

    la    $a1, input       # load input address into
$a1
    j     printWord        # print all words with min
length

```



```

exit: li $v0, 10          # syscall to terminate
      syscall

#####
#####
# checkNonAlpha method: if current character is non-
alphabet --> we got a word      #
#####
#####
checkNonAlpha:
    add $t4, $s0, $a0      # address of A[i] in $t4
    lb  $s1, 0($t4)        # load value of A[i]

    slti $t1, $s1, 65      # if ascii code is less than
48
    bne $t1, $zero, checkLength # get a word

    slti $t1, $s1, 91      # if ascii code is greater
than 90
                                # and
    slti $t2, $s1, 97      # if ascii code is less than
97
    slt $t3, $t1, $t2
    bne $t3, $zero, checkLength # get a word

    slti $t1, $s1, 123     # if ascii character is
greater than 122
    beq $t1, $zero, checkLength # get a word

    addi $s0, $s0, 1       # i = i + 1
    addi $s2, $s2, 1       # j = j + 1
    j checkNonAlpha        # go to checkNonAlpha
#####
#####
# checkLength method: if current word's length < min ->
we have new min length      #
#####
#####
checkLength:

```

```

    slt    $t3, $s2, $s4      # if length of current word
is not shorter than current min (j > min)
    beqz   $t3, next         # reset j and move to the next
one
    add    $s4, $zero, $s2    # else, we have new min
length
next: beq   $s1, 10, done      # if A[i] = '\n' -> done
    addi   $s0, $s0, 1        # i = i + 1
    li     $s2, 0             # j = 0
    j      checkNonAlpha      # proceed next character
done: li    $s0, 0
    li     $s2, 0
    jr     $ra                # return to main
printWord:
    add    $t4, $s0, $a1      # address of A[i] in $t4
    lb     $s1, 0($t4)        # load value of A[i]

    slti   $t1, $s1, 65       # if ascii code is less than
65
    bne    $t1, $zero, compare # get a word

    slti   $t1, $s1, 91       # if ascii code is greater
than 90
                                # and
    slti   $t2, $s1, 97       # if ascii code is less than
97
    slt    $t3, $t1, $t2
    bne    $t3, $zero, compare # get a word

    slti   $t1, $s1, 123      # if ascii character is
greater than 122
    beq    $t1, $zero, compare # get a word

    add    $t5, $s2, $a2      # address of tmp[i] in $t5
    sb     $s1, 0($t5)        # store current character to
tmp[i]
    addi   $s0, $s0, 1        # i = i + 1
    addi   $s2, $s2, 1        # j = j + 1

```

```

j printWord          # go to checkNonAlpha
#####
#####
# compare method: if current word's length = min length
-> print            #
#####
#####
compare:
    beq    $s2, $s4, print    # if length of current word
is equal to current min
    j reset          # reset j and move to the next one
print:
    li     $v0, 4           # print_word
    la     $a0, newline     # line break and tab
    syscall
    li     $v0, 4           # print_string
    la     $a0, tmp         # word with min length
    syscall
reset:
    addi   $s0, $s0, 1      # i = i + 1
    li     $s6, 0          # k = 0
clear_tmp:
    beq    $s6, $s2, return  # if k = j end
    add    $t4, $s6, $a2     # loop through tmp string
    sb     $0, 0($t4)        # turn tmp[k] into 0
    addi   $s6, $s6, 1      # k = k + 1
    j clear_tmp             # continue to clear
return:
    li     $s2, 0           # j = 0
    beq    $s1, 10, end_main # if A[i] = '\n' -> done. Press
enter to continue
    j printWord             # proceed next character
end_main:
    j exit

```

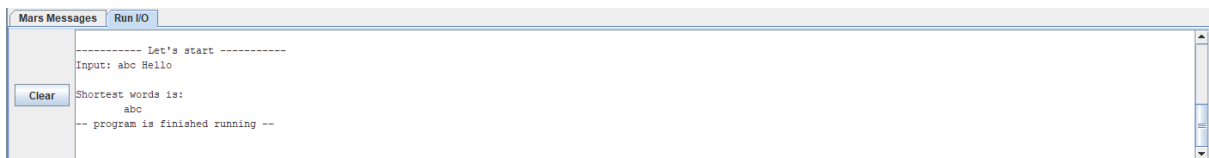
- Cách thực hiện:

- Hình ảnh minh họa:

- + B1: Nhập vào chuỗi
- + B2: Lấy các từ có trong chuỗi.
- + B3: Xét từ nào có độ dài nhỏ nhất thì in ra màn hình và lấy độ dài đó làm độ dài nhỏ nhất
- + B4: Tìm các từ còn lại có độ dài bằng độ dài nhỏ nhất thì in ra màn hình

- Hình ảnh minh họa:

- + Với chuỗi “abc Hello”:



- + Với chuỗi “Hello olleh abc cba”:

