**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: .asciiz "Nhap N (Nhap so co tu hai chu so tro
len va nho hon 25 chu so): "
Message2: .asciiz "N co it nhat 2 chu so trung nhau!"
Message3: .asciiz "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,…)

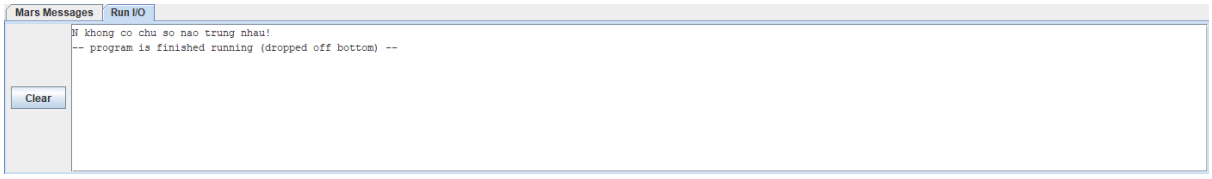
+ B2: Sử dụng mảng A lưu các chữ số khác nhau của số nguyên N

+ 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: .asciiz "Nhap so phan tu: "
Message2: .asciiz "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   # nhap lai neu khong phai so
nguyen
  beq $a1, -3, main   # nhap lai neu khong phai so
nguyen
  beq $a1, -2, exit   # thoat khi nguoi dung an cancel

  add $s0, $zero, $a0   # N - so phan tu cua mang

  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  # nhap lai neu khong phai
so nguyen
  beq $a1, -3, input_element  # nhap lai neu khong phai
so nguyen
  beq $a1, -2, exit     # thoat khi nguoi dung an
cancel

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

  sw $a0, 0($t1)      # lưu so vua nhap vao mang 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:
```

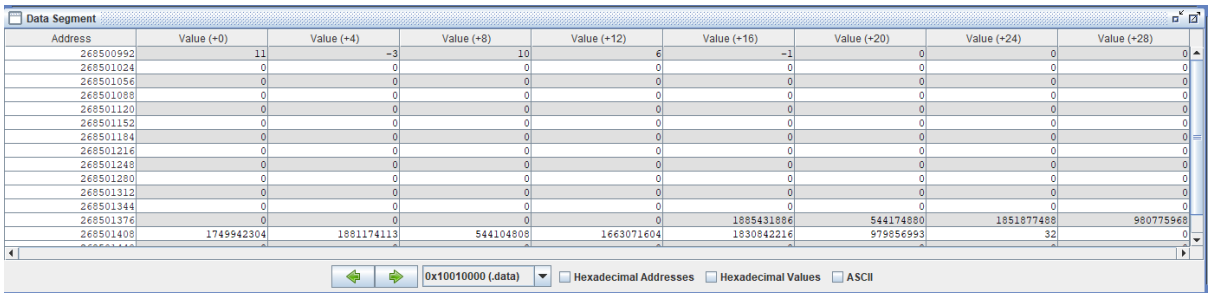
**- Cách thực hiện:**

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

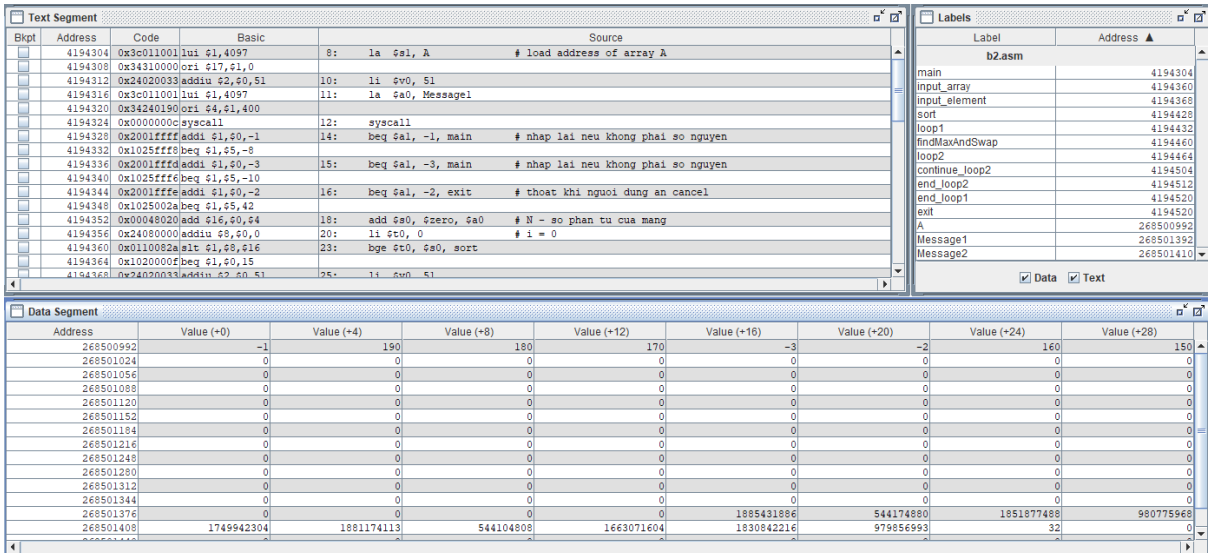
+ B2: Ta chọn phần tử lớn hơn 0 đầu tiên làm current_max. Sau đó lặp qua các phần tử trong mảng A, phần tử nào lớn hơn 0 và lớn hơn current_max thì đổi chỗ current_max với phần tử đó rồi gán current_max bằng phần tử đó. Sau khi lặp hết mảng A, ta lại chọn phần tử lớn hơn 0 đầu tiền trong phần chưa được 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]:




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




## *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: .asciiz "\nInput: "
  message: .asciiz "\nShortest words is: "
  newline: .asciiz "\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

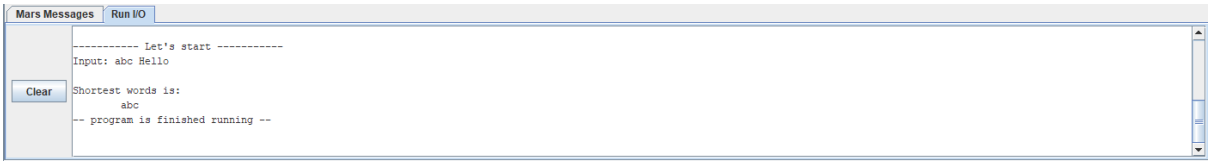
+ B2: Lấy các từ có trong chuỗi.

+ 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”: