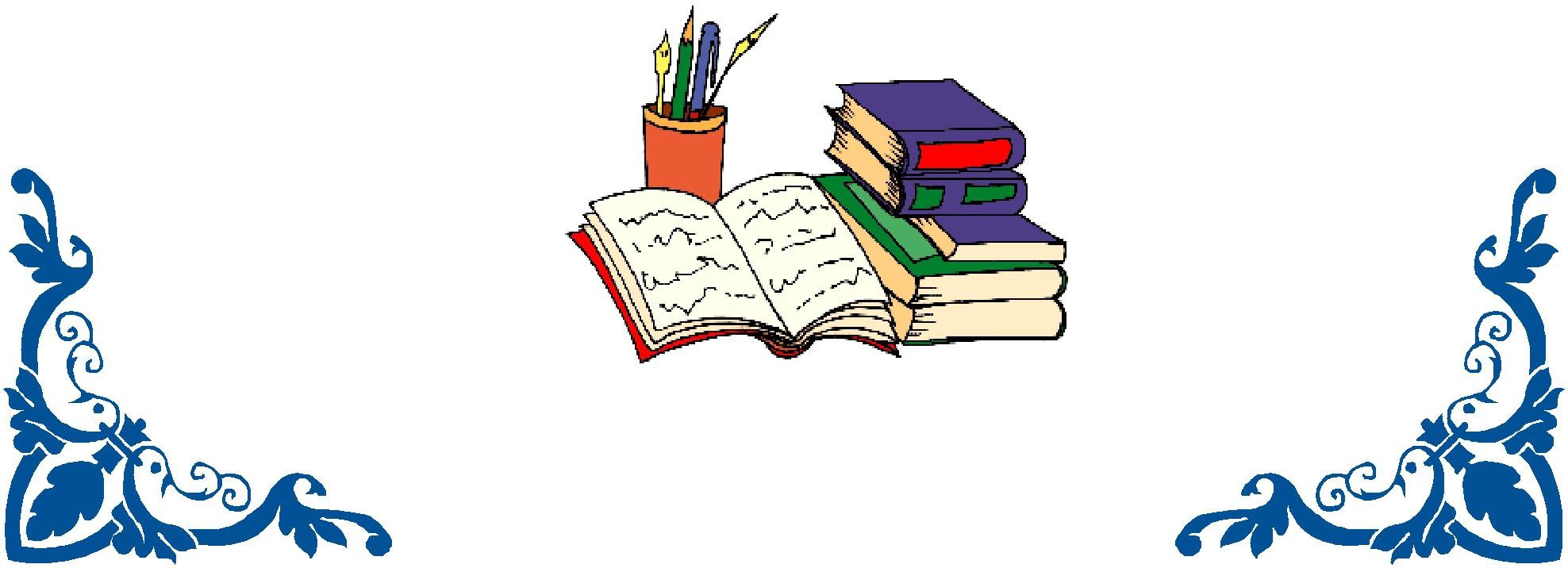


**ĐẠI HỌC BÁCH KHOA HÀ NỘI**



### BÁO CÁO MINI-PROJECTS

**MÔN THỰC HÀNH KIẾN TRÚC MÁY TÍNH**

**Giảng viên hướng dẫn: Lê XuânThành**

**Sinh viên thực hiện:**

**Nguyễn Mạnh Hiệp – 20176750**

**Đoàn Việt Hoàng - 20176766**

1

# Phân công nhiệm vụ:

# Nguyễn Mạnh Hiệp: 16, 21, 24

# Đoàn Việt Hoàng: 15, 23

# Nguyễn Mạnh Hiệp

# Bài 16: Given a sequence of integers as an array, determine whether it is possible to obtain astrictly increasing sequence by removing no more than one element from the array. Note: sequence a0, a1, ..., an is considered to be strictly increasing if a0 < a1 < ... <an. Sequences containing only one element are also considered to be strictly increasing. Example: - For sequence = [1, 3, 2, 1], the output should be almostIncreasingSequence(sequence) = false. There is no one element in this arraythat can be removed in order to get a strictly increasing sequence. - For sequence = [1, 3, 2], the output should be almostIncreasingSequence(sequence)= true. You can remove 3 from the array to get the strictly increasing sequence [1, 2].Alternately, you can remove 2 to get the strictly increasing sequence [1, 3].

# Phân tích cách thực hiện:

# 

# Mã nguồn:

# .data

# TRUE: .asciiz "Ket qua la: TRUE"

# FALSE: .asciiz "Ket qua la: FALSE"

# slpt: .asciiz "So luong phan tu cua mang:"

# element: .asciiz "Nhap phan tu:"

# err: .asciiz "Chuong trinh bi loi nhap lieu"

# #$s5 luu ket qua cua bai toan . Quy uoc true =1, false =0

# # [1, 3, 2, 1] returns 0

# # [1, 3, 2] returns 1

# # [1, 2, 1, 2] returns 0

# # [1, 4, 10, 4, 2] returns 0

# # [10, 1, 2, 3, 4, 5] returns 1

# # $s7 luu dia chi cua mang

# #$s0 luu so phan tu cua mang

# #$s1 luu gia tri cua bien count

# #$t0 luu gia tri bien chay i

# #$t2 luu dia chi cua A[i]

# #$t3 luu gia tri cua A[i]

# #$t4 luu dia chi, gia tri cua A[i+1]

# #$t5 luu dia chi,gia tri cua A[i-1]

# #$t6 luu gia tri cua n-2

# #$t7 luu gia tri cua A[i+2]

# .text

# main:

# la $s7,0x10010060 # dia chi co so cua mang

# li $v0,51

# la $a0,slpt

# syscall

# add $s0,$0,$a0 # so luong phan tu

# li $t0,0

# 

# input:

# beq $t0,$s0,end\_input

# sll $t2,$t0,2

# add $t1,$s7,$t2 # chua dia chi cua a[i]

# 

# li $v0,51

# la $a0,element

# syscall

# 

# bnez $a1,error # neu $a1 !=0 error

# 

# sw $a0,0($t1) # luu gia tri cua a[i]

# 

# add $t0,$t0,1 # i++

# 

# j input

# end\_input:

# #init value

# la $a0,0($s7)

# li $s1,0 # count = 0

# li $t0,0 # i=0

# addi $s0,$s0,-1 # n-1

# jal almostIncreasingSequence

# nop

# li $v0,10 # exit

# syscall

# almostIncreasingSequence:

# loop:

# beq $t0,$s0,done #i=n-1 thi re nhanh

# if:

# # load a[i]

# add $t2,$t0,$t0

# add $t2,$t2,$t2 # $t2 = 4\*i

# add $t2,$t2,$a0

# lw $t3,0($t2) # a[i]

# #load a[i+1]

# add $t4,$t2,4 # $t4= $t2+4

# lw $t4,0($t4) # a[i+1]

# 

# blt $t3,$t4,end\_if # a[i]<a[i+1] end\_if

# 

# add $s1,$s1,1 # count++

# 

# blt $t0,1,end\_if # i<1 end\_if

# 

# #load a[i-1]

# add $t5,$t2,-4 # chua dia chi cua a[i-1]

# lw $t5,0($t5) # a[i-1]

# 

# blt $t5,$t4,end\_if #a[i-1]<a[i+1]

# 

# add $t6,$s0,-1 # n-2

# ble $t6,$t0,end\_if # n-2<=i thi end\_if

# 

# #load a[i+2]

# add $t7,$t2,8 # chua dia chi cua a[i+2]

# lw $t7,0($t7) # a[i+2]

# 

# blt $t3,$t7,end\_if # a[i]<a[i+2] thi end\_if

# j false

# 

# end\_if:

# 

# add $t0,$t0,1 #i=i+1

# j loop

# end\_loop:

# done:

# bgt $s1,1,false #if count>1 return false

# true:

# la $a0,TRUE

# li $v0,55

# li $a1,1 # information message

# syscall

# j exit

# false:

# la $a0,FALSE

# li $v0,55

# li $a1,1 # information message

# syscall

# j exit

# exit:

# jr $ra

# error:

# la $a0,err

# li $v0,55

# li $a1,0 #error message

# syscall

# 

# li $v0,10 #exit

# syscall

# Kết quả:

# 

# 

# 

# 

# 

# Bài 21: Let's define the digit degree of some positive integer as the number of times we need to replace this number with the sum of its digits until we get to a one digit number.Given an integer, find its digit degree. Example ● For n = 5, the output should be digitDegree(n) = 0; ● For n = 100, the output should be digitDegree(n) = 1. 1 + 0 + 0 = 1. ● For n = 91, the output should be digitDegree(n) = 2. 9 + 1 = 10 -> 1 + 0 =1.

# Phân tích cách thực hiện

# 

# Mã nguồn:

# .data

# Ten: .byte 0x0A # Ten =10

# .text

# main:

# li $s5,0 # result

# lb $t0,Ten # $t0 =10

# 

# input:

# li $v0,5

# syscall

# add $s0,$0,$v0 # $s0 luu gia tri nhap vao

# jal digitDegree

# nop

# end\_main:

# digitDegree:

# #ham check dung de xem number <10 hay chua

# check:

# blt $s0,10,print\_result

# add $s5,$s5,1

# j sum\_of\_digit

# #ham print\_result de in ra ket qua

# print\_result:

# li $v0,1

# add $a0,$s5,$0

# syscall

# 

# done:

# li $v0,10

# syscall

# 

# #ham sum\_of\_digit dung de tinh tong cac chu so

# sum\_of\_digit:

# li $s1,0 # khoi tao sum=0

# add $t2,$s0,$0

# while:

# div $t2,$t0

# mfhi $t1 #du

# mflo $t2 #thuong

# add $s1,$s1,$t1

# beqz $t2,end\_sum\_of\_digit

# j while

# end\_sum\_of\_digit:

# 

# add $s0,$s1,$0 #cap nhat lai number = tong cac chu so cua no

# j check

# exit:

# jr $ra

# Kết quả:

# 

# Bài 24: Cyclone Word (challenge) Cyclone words are English words that have a sequence of characters in alphabetical order when following a cyclic pattern. Example: Write a function to determine whether a word passed into a function is a cyclone word. You can assume that the word is made of only alphabetic characters, and is separated by whitespace. is\_cyclone\_phrase("adjourned") # => True is\_cyclone\_phrase("settled") # => False

# Phân tích cách thực hiện

# Xấy dựng 1 dãy số gồm các chỉ số cyclone

# for(i=0; i<=n/2; i++){lưu 2 giá trị (I, (n-1)-i)}

# Nếu n-1 chẵn thì thêm phần tử (n-1)/2 còn không thì thôi.

# Ví dụ length = 9 thì dãy số a=[0, 8, 1, 7, 2 , 6, 3, 5, 4]. Sau đó sẽ so sánh các string[a[i]] và string[a[i+1]]

# if(string[a[i]] > string[a[i+1]] => false

# else => true

# Mã nguồn

# .data

# string: .space 10

# Message1: .asciiz "Nhap xau:"

# Message2: .asciiz "Do dai la "

# TRUE: .asciiz "Ket qua la: TRUE"

# FALSE: .asciiz "Ket qua la: FALSE"

# .text

# #is\_cyclone\_phrase

# #is\_cyclone\_phrase("adjourned") # => True

# #is\_cyclone\_phrase("settled") # => False

# main:

# get\_string:

# li $v0, 54

# la $a0, Message1

# la $a1, string

# la $a2, 100

# syscall

# 

# jal get\_length\_of\_string

# nop

# 

# jal is\_cyclone\_phrase

# nop

# 

# li $v0,10 # exit

# syscall

# end\_main:

# # ham tim do dai cua xau nhap vao

# get\_length\_of\_string:

# li $s1,10 # ma ascii cua ki tu xuong dong

# 

# get\_length:

# la $a0,string # a0 = Address(string[0])

# add $s5,$0,$a0

# xor $s0,$zero, $zero # s0 = length = 0

# xor $t0,$zero, $zero # t0 = i = 0

# 

# check\_char:

# add $t1,$a0, $t0 # t1 = a0 + t0

# lb $t2, 0($t1) # t2 = string[i]

# beq $t2,$s1,end\_of\_str

# addi $s0, $s0, 1 # s0=s0+1->length=length+1

# addi $t0, $t0, 1 # t0=t0+1->i = i + 1

# j check\_char

# 

# end\_of\_str:

# 

# end\_of\_get\_length:

# jr $ra

# #thuc hien thuat toan

# # y tuong: Xay dung mot day so gom cac chi so cyclone

# # for(i=0;i<=n/2-1;i++) {luu 2 gia tri (i,(n-1)-i}

# # neu n-1 chan thi them phan tu (n-1)/2 con khong thi thoi

# # VD length =9 thi day so la a=[0,8,1,7,2,6,3,5,4]

# # Sau do se so sanh cac string[a[i]] va string[a[i+1]]

# # string[a[i]]>string[a[i+1]] false else true

# is\_cyclone\_phrase:

# #init value

# 

# div $t1,$s0,2

# add $t1,$t1,-1 # n/2-1

# addi $s0,$s0,-1 # length = length -1

# li $t0,0 # i=0

# la $a0,0x1001005c # dia chi cua mang

# add $t2,$0,$a0 # $t2 cung luu dia chi co so cua mang

# 

# loop:

# bgt $t0,$t1,end\_loop #if i>(n/2-1) re nhanh

# 

# add $t3,$0,$t0

# sw $t3,4($t2) # luu gia tri cua i

# 

# sub $t4,$s0,$t0

# sw $t4,8($t2) # luu gia tri cua (n- 1)-i

# add $t0,$t0,1 # i++

# add $t2,$t2,8 # tang gia tri dia chi len 8

# 

# j loop

# end\_loop:

# 

# li $t7,2

# div $s0,$t7 # tinh div (length-1),2

# mfhi $t5 # lay so du

# 

# beqz $t5,even

# beq $t5,1,odd

# even:

# mflo $t6

# sw $t6,4($t2) # neu n=9 thi day so la 0,8,1,7,2,6,3,5,4

# odd: #neu n=8 thi day so 0,7,1,6,2,5,3,4

# # $s5 luu dia chi co so cua string

# add $t2,$a0,4 # dia chi mang chi so

# li $t0,0 # i=0

# 

# loop2:

# 

# bge $t0,$s0,end\_loop2

# 

# lw $t1,0($t2) # lay chi so a[i]

# lw $t3,4($t2) # lay chi so a[i+1]

# 

# add $t4,$s5,$t1

# lb $t4,0($t4) # string[a[i]]

# add $t5,$s5,$t3

# lb $t5,0($t5) # string[a[i+1]]

# bgt $t4,$t5,false

# 

# add $t0,$t0,1 #i++

# add $t2,$t2,4 #tang dia chi co so cua mang chi so

# j loop2

# 

# end\_loop2:

# true:

# li $v0,55

# la $a0,TRUE

# li $a1,1

# syscall

# j exit

# false:

# li $v0,55

# la $a0,FALSE

# li $a1,1

# syscall

# j exit

# exit:

# jr $ra

# end\_main:

# Kết quả:

# 

# Đoàn Việt Hoàng

# Bài 15: You are given an array of integers. On each move you are allowed to increase exactly one of its elements by one. Find the minimal number of moves required to obtain astrictly increasing sequence from the input. Example: For inputArray = [1, 1, 1], the output should be arrayChange(inputArray) =3. The minimal number of moves needed to obtain a strictly increasing sequence from inputArray. It's guaranteed that for the given test cases the answer always fits signed 32-bit integer type.

# Phân tích cách thực hiện

# Duyệt lần lượt các phần tử của mảng. Nếu một phần tử a[I] có giá trị nhỏ hơn phẩn tử ngay trước nó a[i-1] số lần moves sẽ tăng lên giá trị bằng a[i-1] – a[i] + 1. Sau đó tăng giá trị a[i]=a[i-1]+1

**public** **static** **int** arrayChange(**int**[] inputArray){

**int** result = 0;

**for** (**int** i = 1; i < inputArray.Length; i++){

**if** (inputArray[i] <= inputArray[i - 1]){

result += inputArray[i - 1] - inputArray[i] + 1;

inputArray[i] = inputArray[i - 1] + 1;

}

}

**return** result;

}

# Mã nguồn:

# .data

# slpt: .asciiz "So luong phan tu cua mang:"

# element: .asciiz "Nhap phan tu:"

# err: "Loi nhap lieu"

# .text

# main:

# la $s7,0x10010060 #dia chi co so cua mang

# li $v0,51

# la $a0,slpt

# syscall

# add $s0,$0,$a0 #so luong phan tu

# li $t0,0 #i=0

# input:

# beq $t0,$s0,end\_input #neu i=slpt=>ket thuc nhap

# sll $t2,$t0,2 #i=4i

# add $t1,$s7,$t2 # chua dia chi cua a[i]

# li $v0,51

# la $a0,element

# syscall

# bnez $a1,error #neu nhap bi loi thong bao error va thoat chuong trinh

# sw $a0,0($t1) # luu gia tri cua a[i]

# add $t0,$t0,1 # i++

# j input

# end\_input:

# la $a0,0($s7) #dia chi co so cua mang

# add $s1,$zero,$s0 #do dai mang luu o $s1

# xor $t0,$zero,$zero #add = 0

# xor $s0,$zero,$zero #result = 0

# xor $t1,$zero,$zero #i=0

# jal arrayChange

# nop

# li $v0,10

# syscall

# arrayChange:

# loop:

# slt $t4, $t1, $s1 #$t6 = i < length ? 1 : 0

# beq $t4, $zero, print\_result #if i>=length => end

# xor $t0, $zero, $zero #add = 0

# sll $t2, $t1, 2 #i=4i

# add $t3, $t2, $a0

# lw $t6, 0($t3) #A[i]

# beq $t1, $0, next\_step #neu i = 0 => next\_step

# slt $t4, $t5, $t6

# beq $t4, 1, next\_step #A[i]<A[i-1] => loop

# j add\_moves

# next\_step:

# add $t1, $t1, 1 #i=i+1

# add $t5, $t6, $0 #$t4=A[i]

# j loop

# add\_moves:

# sub $t0, $t5, $t6 #add = A[i-1]-A[i]

# add $t0, $t0, 1 #add = A[i-1]-A[i]+1

# add $s0, $s0, $t0 #result = result + add

# add $t6, $t6, $t0 #A[i]=A[i]+add

# j next\_step

# 

# print\_result:

# li $v0, 1

# add $a0, $s0, $zero

# syscall

# jr $ra

# error:

# la $a0,err

# li $v0,4

# syscall

# li $v0,10

# syscall

# Kết quả:

# 

# Các trường khác:

# 1,2,1,3

# 

# 1,1,1,1

# 

# Bài 23: Surpassing words are English words for which the gap between each adjacent pair of letters strictly increases. These gaps are computed without "wrapping around" from Z to A. For example: Write a function to determine whether a word passed into a function is a surpassing word. You can assume the word is made of only alphabetic characters, and are separated by whitespace. We will consider the empty string and a 1-character string to be a valid surpassing word. is\_surpassing\_word("superb") # => True is\_surpassing\_word("excellent") # => False.

# Ý tưởng thực hiện:

# Nhập xâu

# Duyệt lần lượt các kí tự trong xâu và lần lượt tính khoảng cách từng đôi kí tự liền kề nhau (khoảng cách mã ascii)

# Trả về true nếu dãy các khoảng cách là dãy tăng tuyệt đối, false nếu không phải dãy tăng tuyệt đối

# Mã nguồn

# .data

# string: .space 50

# message1: .asciiz "Nhap xau: "

# message2: .asciiz "Do dai la: "

# TRUE: .asciiz "TRUE"

# FALSE: .asciiz "FALSE"

# .text

# li $s1,10 #Ma ascii cua ki tu xuong dong

# li $s5, 0 #Ket qua

# get\_string:

# li $v0,54 #input dialog string code

# la $a0,message1 #address of the null-terminated message string

# la $a1,string #address of input buffer

# la $a2,100 #maximum number of characters to read

# syscall

# get\_length:

# la $a0,string #a0 = Address(string[0])

# xor $s0,$zero,$zero #s0 = length = 0

# xor $t0,$zero,$zero #t0 = i = 0

# check\_char:

# add $t1,$a0,$t0

# lb $t2,0($t1) #t2 = string[i]

# beq $t2,$s1,end\_of\_str #Is null char?

# addi $s0,$s0,1 #length=length+1

# addi $t0,$t0,1 #i = i + 1

# j check\_char #jump to check char

# end\_of\_str:

# end\_of\_get\_length:

# jal process

# process:

# #init value

# add $s2,$s0,-2 #$2=length-2

# li $t0, 0 #i = 0

# add $t1, $zero, $a0 #gia tri co so cua string

# loop:

# beq $t0,$s2,true #neu i = length -2 tra ve true

# lb $t4,0($t1) #string[i]

# lb $t5,1($t1) #string[i+1]

# lb $t6,2($t1) #string[i+2]

# sub $t7, $t5, $t4 #$t7=t5-$t4

# sub $t8, $t6, $t5 #$t8=$t6-$t5

# abs $t7, $t7 # $t7 = |$t7|

# abs $t8, $t8 # $t8 = |$t8|

# bge $t7,$t8,false #neu $t7 >= $t8 =>false

# add $t1,$t1, 1 #tang dia chi co so len 1

# add $t0,$t0,1 #i+=1

# j loop

# true:

# li $v0,4

# la $a0, TRUE

# syscall

# li $s5,1 #tra ve $s5=1 neu true

# li $v0, 10

# syscall

# j exit

# false:

# li $v0,4

# la $a0, FALSE

# syscall

# 

# li $s5,0 #tra ve $s5=0 neu false

# li $v0, 10

# syscall

# j exit

# exit:

# jr $ra

# Kết quả:

# 

# 

# 

# 