BÁO CÁO THỰC HÀNH GIỮA KÌ
IT3280 – 156788 – THỰC HÀNH KIẾN TRÚC MÁY TÍNH

Họ và tên	Nguyễn Minh Quân
Mã số sinh viên	20235816

Assignment A - 2

Nhập số nguyên dương N từ bàn phím, in ra màn hình dãy số Fibonacci nhỏ hơn N.

1. **Phân tích cách thực hiện**

   Chương trình có nhiệm vụ in ra **dãy số Fibonacci nhỏ hơn một số nguyên N** được nhập từ bàn phím.

   Bước 1: In thông báo yêu cầu nhập số nguyên (mess1), sau đó nhận Input số nguyên N từ bàn phím và lưu vào thanh ghi t0

   Bước 2: Kiểm tra nếu N <= 0 thì kết thúc chương trình

   Bước 3: In thông báo (mess2) và in hai số đầu tiên của dãy Fibonacci là 0 và 1

   Bước 4: Dùng vòng lặp để tính các số Fibonacci tiếp theo t3 = t1 + t2 , in ra nếu nhỏ hơn `N`. Cập nhật lại t1 và t2

   Bước 5: Khi số tiếp theo >= N, chương trình sẽ nhảy tới nhãn kết thúc

2. **Ý nghĩa của các chương trình con**

   get_Input: Đoạn mã này thực hiện việc đọc một số nguyên từ bàn phím và lưu trữ giá trị đó vào thanh ghi t0.

   fibonacci: Đây là 1 vòng lặp với mục đích tính số fibonacci tiếp theo, nếu vượt quá N thì nhảy tới nhãn ket_thuc, còn không thì in ra sau đó cập nhất 2 số trước đó

3. **Mã nguồn**

```
.data
    mess1: .asciz "Nhap so nguyen: "
    mess2: .asciz "Day fibonnaci nho hon N la: "
    space: .asciz " "
    error_msg: .asciz "N phai lon hon 0"
.text
main:
    #In chuoi Nhap so nguyen
    li a7,4
    la a0, mess1
    ecall

get_Input:
    #Nhap so nguyen
    li a7,5
    ecall
    mv t0,a0  # t0 = N

    #Kiem tra N
    bltz t0, error   # Nếu N < 0
    beqz t0, error   # Nếu N = 0
    beq x0,x0, valid_Input
error:
    li a7, 4
     la a0, error_msg  # Chuỗi "N phai lon hon 0!"
     ecall
     j main

valid_Input:
    #In mess2
    li a7,4
```

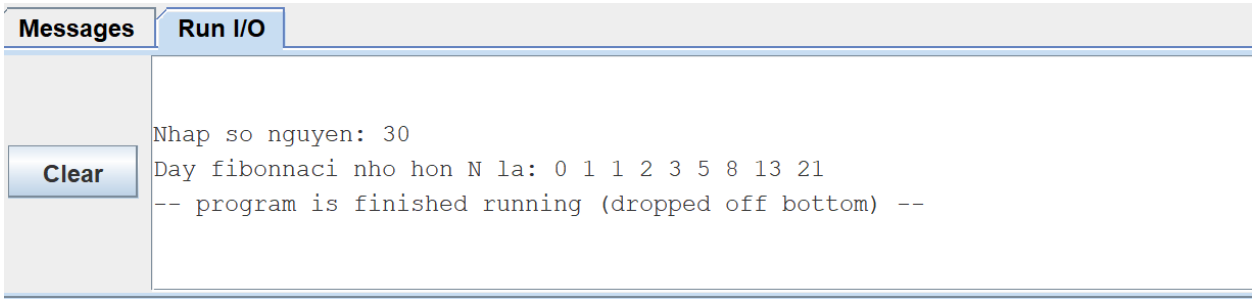
```
    la a0, mess2
    ecall

    #Gan 2 so dau voi 0 va 1
    li t1,0 #t1 =0
    li t2,1 #t2 =1

    #In so dau tien( 0 )
    li a7,1
    mv a0,t1
    ecall

    #In space
    li a7,4
    la a0, space
    ecall

    #In so thu 2 ( 1 )
    li a7,1
    mv a0,t2
    ecall

    #In space
    li a7,4
    la a0, space
    ecall


fibonacci:
    add t3, t1, t2  #t3 = t1 + t2

    bgt t3, t0,ket_thuc #if t1 + t2 >= N --> end
```

```
    #In so tiep theo t3
    li a7, 1
    mv a0, t3
    ecall

    #In space
    li a7,4
    la a0, space
    ecall

    #update t1, t2
    mv t1,t2  #t1 = t2
    mv t2,t3  #t2 = t3

    #loop
    beq x0,x0,fibonacci

ket_thuc:
```

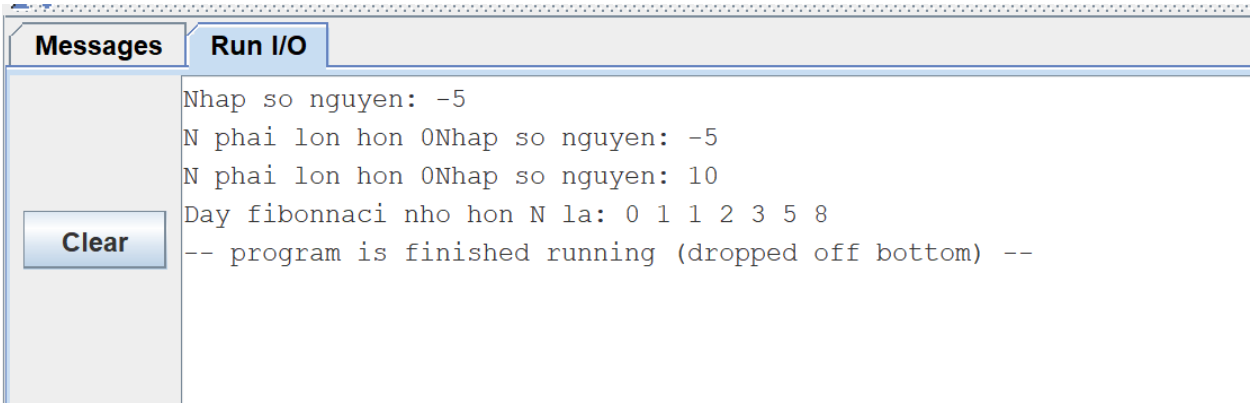
ket_thuc:

4. Kết quả thực hiện chương trình

TH1: N > 0



TH2: N<=0



Assignment B – 5

Nhập mảng từ bàn phím. Sắp xếp các phần tử có giá trị dương tăng dần, các phần tử còn lại giữ nguyên vị trí. Ví dụ: Nhập mảng [-1, 150, 190, 170, -2, -3, 160, 180], kết quả sau khi sắp xếp [-1, 150, 160, 170, -2, -3, 180, 190].

1. **Phân tích cách thực hiện**

   Chương trình thực hiện nhiệm vụ sắp xếp các phần tử có giá trị dương tăng dần, các phần tử còn lại giữ nguyên vị trí.

   Bước 1: Nhập số phần tử của mảng từ người dùng (N). Nếu N <= 0 thì kết thúc chương trình.

   Bước 2: Nhập từng phần tử cho mảng A.

   Bước 3: Thực hiện sắp xếp mảng A tăng dần chỉ với các số dương. Các phần tử âm hoặc bằng 0 được giữ nguyên vị trí.

   Bước 4: In ra mảng sau khi sắp xếp.

2. **Ý nghĩa của các chương trình con**

input : Vòng lặp với mục đích cho phép người dùng nhập số phần tử của mảng

loop1 : Là vòng lặp ngoài duyệt qua từng phần tử A[i] để làm mốc. Nếu A[i] <= 0, bỏ qua. Ngược lại, chuyển sang loop2 để tìm phần tử phía sau nhỏ hơn để hoán đổi.

loop2 : Duyệt các phần tử A[j] với j > i. Bỏ qua nếu A[j] <= 0. Nếu A[j] < A[i], thực hiện hoán đổi qua nhãn swap:. Lặp đến hết dãy.

swap : Thực hiện hoán đổi A[i] và A[j] khi A[i] > A[j]. Sau khi hoán đổi, cập nhật lại biến s1 để tiếp tục so sánh trong vòng lặp.

print : Đảm nhận việc in ra toàn bộ mảng đã sắp xếp. Vòng lặp kết thúc khi đã in đủ N phần tử.

3. Mã nguồn

```
.data
        mess1: .asciz "Nhap so phan tu cua mang : "
        mess2: .asciz "Nhap tung phan tu cho mang : "
        mess3: .asciz "Chuoi sau khi sap xep : "
        space: .asciz " "
        .align 2 #Set up dia chi bat dau mang A chia het cho 4,neu k co dong nay khi chay se loi o dong sw a0, 0(t2) do dia chi hien tai khong chia het cho 4
        A: .space 400
.text
        #In mess1
        li a7, 4
        la a0, mess1
        ecall
```

```
        #Input N
        li a7,5
        ecall
        mv a2,a0 # a2 = N ( kich thuoc cua mang )

        #kiem tra N hop le
        blez a2, end       # Nếu N <= 0 -> kết thúc


        # In mess2
        li a7, 4
        la a0, mess2
        ecall

        li t0, 0 # i = 0
        la s0, A #a0 = address A[0]

input:
        bge t0, a2, endInput # if i >= N --> endInput

        #nhap A[i]
        li a7,5
        ecall


        slli t1, t0, 2 # t1 = 4 * i
        add t2, s0, t1 # t2 = A + 4 * i
        sw a0, 0(t2) # A[i] = a0

        addi t0,t0,1
        beq x0,x0, input
```

```
endInput:
        # a2 = N

        li t0, 0 #i = 0
        addi t2, a2, -1 # t2 = N -1
loop1:
        bge t0,t2, endloop1  # if i >= N - 1 --> endloop
        #get a[i]
        slli t3,t0,2
        add t4,s0,t3
        lw s1, 0(t4) # s1 = A[i]
        blez s1, endloop2 # check if s0 <= 0 --> i++ va quay lai loop
        addi t1, t0,1 # j = i + 1

loop2:
        bge t1, a2, endloop2 # if j >= N --> endloop2
        #get a[j]
        slli t5,t1,2
        add t6, s0, t5
        lw s2, 0(t6) #s2 = A[j]

        blez s2, no_swap # if s2 <= 0 --> No swap
        ble s1,s2,no_swap  # if A[i] <= A[j] --> No swap

swap:
        sw s1, 0(t6)
        sw s2, 0(t4)
        mv s1, s2  #Cap nhat A[i[ moi]
no_swap:
        addi t1,t1,1
        beq x0,x0, loop2

endloop2:
```

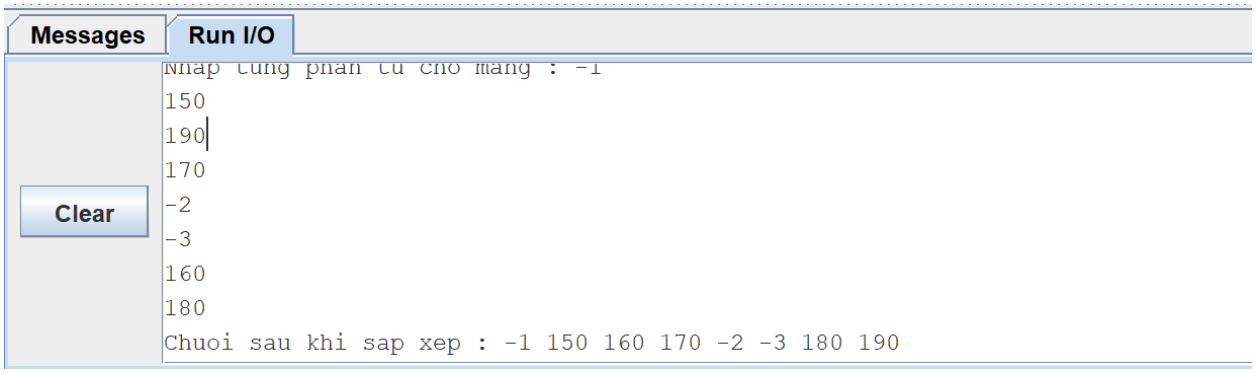
```
        addi t0,t0, 1
        beq x0,x0, loop1
endloop1:

        #In chuoi ra man hinh
        li a7, 4
        la a0, mess3
        ecall

        li t6, 0
print:
        bge t6,a2,end
        slli t3,t6, 2
        add t5, s0, t3
        lw s3, 0(t5)

        li a7, 1
        mv a0,s3
        ecall

        #in space
        li a7,4
        la a0,space
        ecall

        addi t6,t6,1
        beq x0,x0,print

end:
    # Kết thúc chương trình
    li a7, 10
    ecall
```

4. Kết quả thực hiện chương trình

TH1: N = 8 theo đề bài

A[8] = [-1, 150, 190, 170, -2, -3, 160, 180]

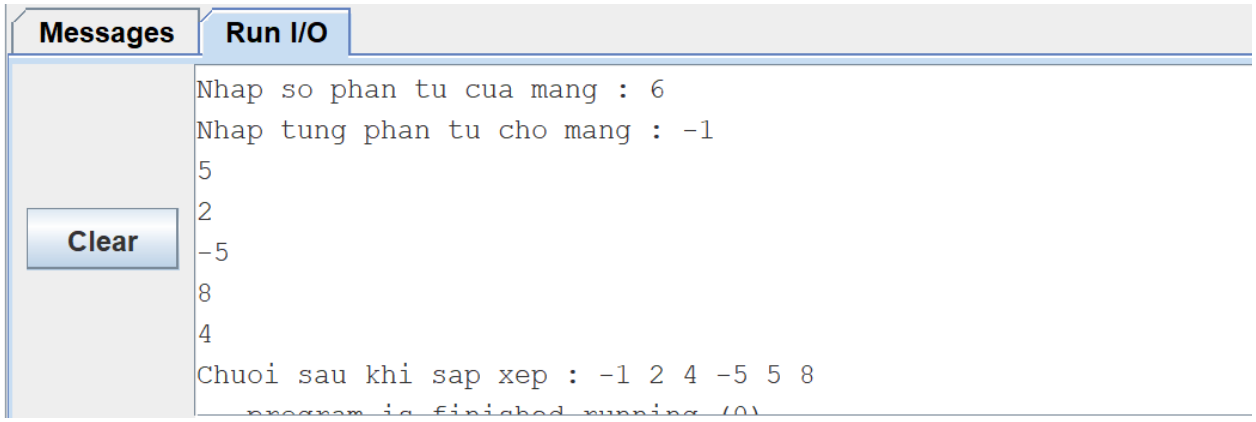
→ Kết quả sau khi sắp xếp



TH2 : N = 6

A[6] =[-1, 5, 2, -5, 8, 4]

➔ Kết quả sau khi sắp xếp [-1, 2, 4, -5, 5 ,8]



Assignment C –15

Nhập vào 2 xâu ký tự A và B. In ra màn hình các ký tự chữ số không xuất hiện cả trong A và B.

1. Phân tích cách thực hiện

Chương trình cho phép người dùng nhập vào hai chuỗi ký tự. Sau đó, nó đếm số lần xuất hiện của các chữ số từ '0' đến '9' trong cả hai chuỗi và in ra những chữ số không xuất hiện trong cả hai chuỗi.

Bước 1: In lời nhắc và cho phép người dùng nhập vào hai chuỗi ký tự.

Bước 2: Duyệt qua từng ký tự trong chuỗi thứ nhất. Nếu ký tự là chữ số ('0' đến '9'), chương trình tăng bộ đếm tương ứng trong mảng C.

Bước 3: Tương tự, duyệt qua chuỗi thứ hai và cập nhật mảng C cho các chữ số xuất hiện.

Bước 4: Sau khi quét cả hai chuỗi, chương trình kiểm tra các phần tử trong mảng C. Với những chỉ số j có C[j] = 0, tức là chữ số j không xuất hiện trong cả hai chuỗi, chương trình sẽ in j ra màn hình.

2. Ý nghĩa của các chương trình con

loopA : Vòng lặp đọc từng ký tự trong chuỗi A. Nếu ký tự nằm trong khoảng '0' đến '9', thì tính chỉ số tương ứng trong mảng C (dựa vào digit = ký tự - '0'), sau đó tăng C[digit]++. Như vậy C[0..9] ghi lại số lần xuất hiện các chữ số trong chuỗi A

loopB : Tương tự loopA, vòng lặp này duyệt chuỗi B, và cũng đếm số lần xuất hiện các chữ số '0' đến '9', cập nhật tiếp vào mảng C. Tổng cộng sau hai vòng lặp loopA và loopB, mảng C chứa số lần xuất hiện của mỗi chữ số trong cả hai chuỗi

loopC : Duyệt mảng C từ 0 đến 9. Nếu C[i] == 0 nghĩa là chữ số i không xuất hiện trong cả hai chuỗi → in ra số đó. Sau mỗi số, in một khoảng trắng

3. Mã nguồn

```
.data
        mess1: .asciz "Nhap chuoi thu nhat : "
        mess2: .asciz "Nhap chuoi thu hai : "
        mess3: .asciz "Cac chu so ko co trong ca 2 chuoi la : "
        space: .asciz " "
        A: .space 100
        B: .space 100
        C: .word 0,0,0,0,0,0,0,0,0,0  #Mang dem chu so 0-9
.text
        #mess1
        li a7,4
        la a0,mess1
        ecall

        #Nhap xau 1
        li a7,8
        la a0,A
        li a1,100
        ecall

        #mess2
        li a7,4
        la a0,mess2
        ecall

        #Nhap xau 2
        li a7,8
        la a0,B
        li a1,100
        ecall

###### DEM KY TU CHU SO TRONG A ########
        li t0,0   #i = 0
```

```
        la s1, A
        la s2, C
loopA:
        add s3,s1,t0    #Dia chi cua A[i]
        lb t2, 0(s3)    #Lay A[i]
        beqz t2, endloopA   #Neu A[i] = '\n' --> ket thuc
         li t5, '0'            # Load ASCII '0' into t5
         blt t2, t5, continue  # if (A[i] < '0') skip
         li t5, '9'            # Load ASCII '9' into t5
         bgt t2, t5, continue  # if (A[i] > '9') skip



        #Neu la chu so, tang C[digit]++
        li t5,'0'
        sub t3,t2,t5 #digit = A[i] -'0'
        slli t3,t3,2  #offset = digit * 4
        add s4,s2,t3  #Dia chi C[digit]
        lw t4, 0(s4) #t4 = C[A[i]]
        addi t4,t4,1
        sw t4,0(s4)



continue:
        addi t0,t0,1
        beq x0,x0, loopA

endloopA:
```

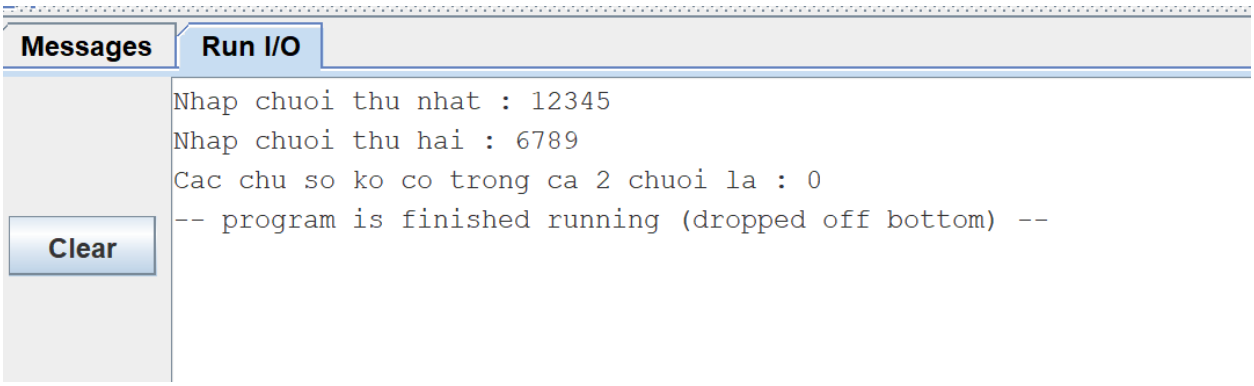
```
###### DEM KY TU CHU SO TRONG B ########
        li t0,0 #i = 0
        la s1, B
        la s2, C
loopB:
        add s3,s1,t0 # s3 = B + i (1 char = 1 byte)
        lb t2, 0(s3)  #( t2 = byte C[i] )
        beqz t2, endloopB  # if t2 = 0 ( null ) --> endloop
         li t5, '0'            # Load ASCII '0' into t5
         blt t2, t5, continues  # if (B[i] < '0') skip
         li t5, '9'            # Load ASCII '9' into t5
         bgt t2, t5, continues # if (B[i] > '9') skip

        li t5,'0'
        sub t3,t2,t5 #(byte)B[i] -'0'
        slli t3,t3,2 # t3 = t3 * 4
        add s4,s2,t3 # s4 = C + t3 *4
        lw t4, 0(s4) #t4 = C[B[i]]
        addi t4,t4,1
        sw t4,0(s4) #C[B[i]]++



continues:
        addi t0,t0,1
        beq x0,x0, loopB

endloopB:

###### In cac chu so ko xuat hien trong ca A va B #####
        li t5,9 # Strength of C = 9
        li t6,0 # j=0
        la t2,C
```

```
        #mess3
        li a7,4
        la a0,mess3
        ecall

loopC:
#printtest
        bgt t6,t5,end #if j > 9 --> end
        slli s8,t6,2 #s8 = 4*j
        add t3,t2,s8 # t3 =  C + 4*j
        lw s6, 0(t3) #s6 = C[j]
        bgt s6,zero, skip

print:
        li a7,1
        mv a0,t6
        ecall

        #space
        li a7,4
        la a0,space
        ecall

skip:

        addi t6,t6,1
        beq x0,x0,loopC

end:
```

4. Kết quả thực hiện chương trình

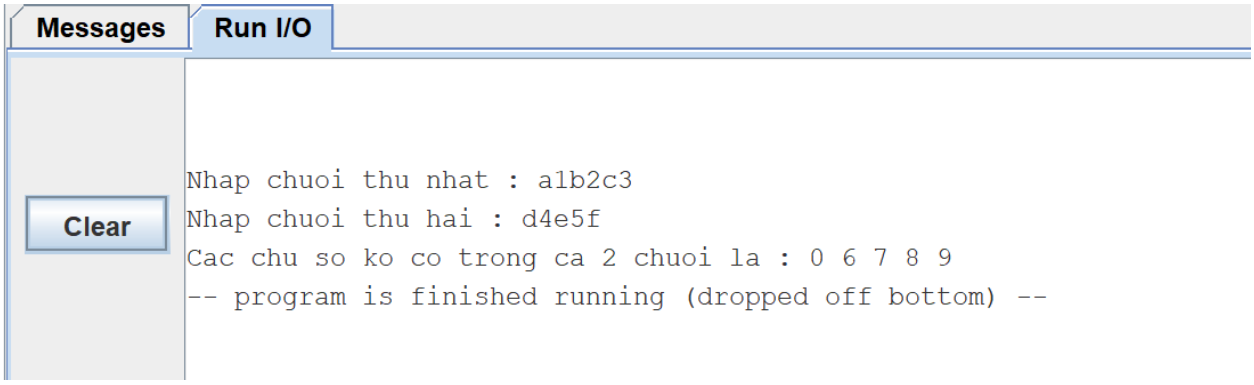
TH1 : Chuỗi 1: 12345

Chuỗi 2: 6789



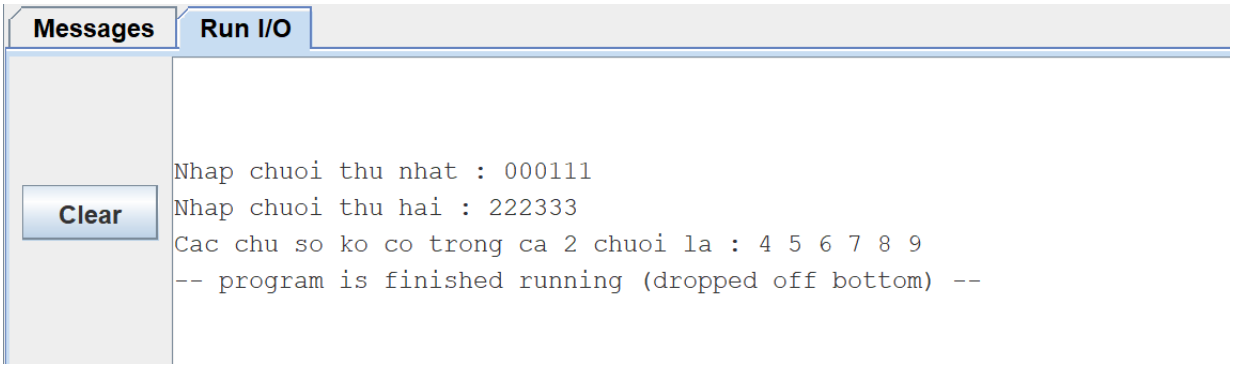
TH2 : Chuỗi 1: a1b2c3

Chuỗi 2: d4e5f



TH3 : Chuỗi 1: 000111

Chuỗi 2: 222333



TH4 : Chuỗi 1: abcdefg

Chuỗi 2: hijklmnop