BÁO CÁO THỰC HÀNH GIỮA KỲ 2024.2

Họ và tên: Nguyễn Thành Duy

MSSV: 20235696

Bài tập được giao: A-9, B-3, C-12

Assignment A-9: Nhập số nguyên dương N có từ 2 chữ số trở lên. In ra màn hình chữ số nhỏ nhất của N.

Code:

```
.data
message_1: .asciz "Nhap so nguyen duong N co tu 2 chu so tro len: "
message_2: .asciz "Chu so nho nhat cua N: "
message_3: .asciz "Nhap sai, vui long thuc hien lai !"
space: .ascii " "
.text
main:
#------ Yeu cau nguoi dung nhap vao mot chu so -------
# In ra message_1
li a7, 4
la a0, message_1
ecall
# Nhap N
li a7, 5
ecall
mv s0, a0 #  s0 = N
# Kiem tra chu so nhap vao tu nguoi dung
bgt s0, zero, find_min
# Neu N < 0 thi in ra message_3
li a7, 55
la a0, message_3
li a1, 0
ecall
j end_main
#------ Tim chu so nho nhat co trong N -------
find_min:
li a2, 10 # a2 = 10
li s2, 0x7fffffff # s2 duoc gan gia tri duong lon nhat
loop:
beqz s0, exit # Lap cho toi khi s0 = 0
rem s1, s0, a2 # s1 = s0 % 10
div s0, s0, a2 # s0 = s0 / 10
```

```
bge s2, s1, update_min # s2 > s1 thi cap nhat

j loop
# Kiem tra va cap nhat gia tri nho nhat
update_min:
mv s2, s1 # Cap nhat gia tri min
j loop
# ------ In ra chu so nho nhat va ket thuc chuong trinh -------
exit:
# In ra message_2
li a7, 4
la a0, message_2
ecall
# In ra chu so nho nhat
li a7, 1
mv a0, s2
ecall
# Ket thuc chuong trinh
end_main:
li a7, 10
ecall
```

Phân tích:

- Chương trình yêu cầu nhập số nguyên dương có từ 2 chữ số trở lên.

+) Nếu người dùng nhập chữ số âm hoặc số 0 thì báo lỗi và yêu cầu thực hiện lại

+) Nếu người dùng nhập số có 1 chữ số dương thì chữ số nhỏ nhất của số đó chính là số đó.

Giải thích:

Ý tưởng giải quyết:

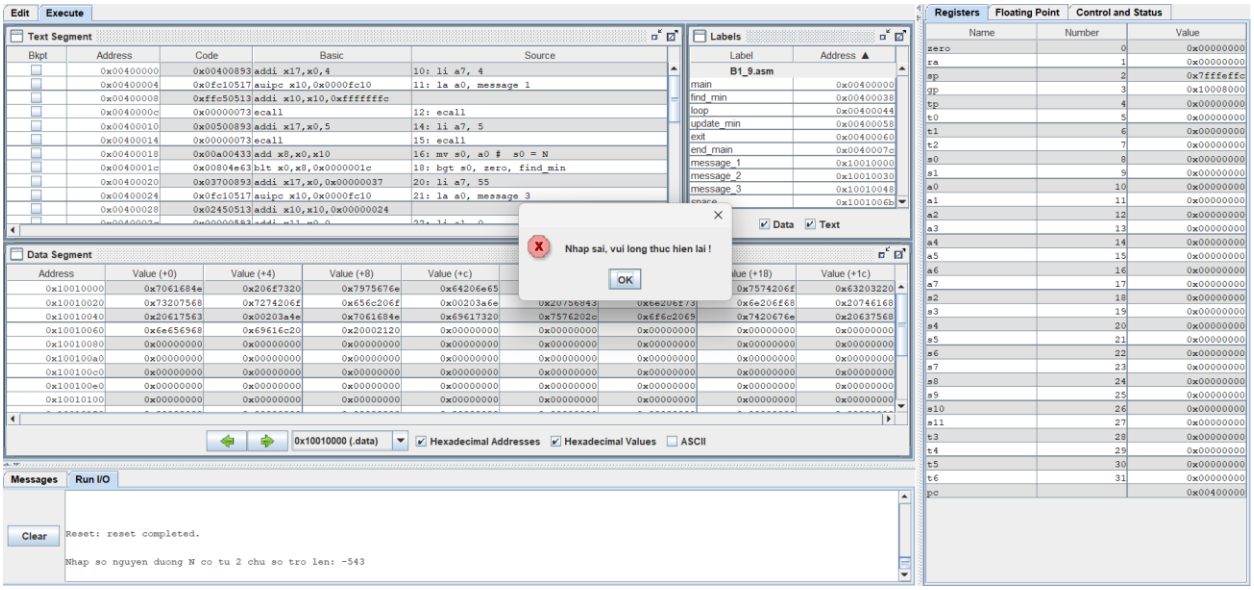
- Nhận vào số nguyên N
- Duyệt qua từng chữ số N, giữ lại chữ số nhỏ nhất
- In ra kết quả.

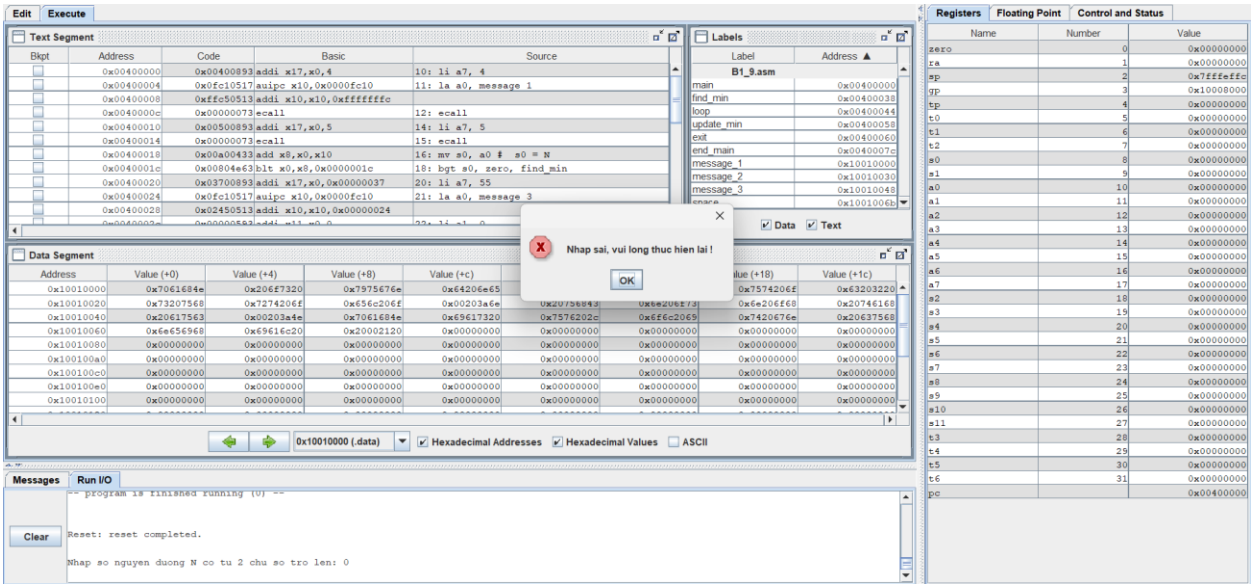
Giải thích đoạn chương trình:

- Đoạn chương trình sau nhận số N từ người dùng.

```
main:
#------ Yeu cau nguoi dung nhap vao mot chu so -------
# In ra message_1
li a7, 4
la a0, message_1
ecall
# Nhap N
li a7, 5
ecall
mv s0, a0 #  s0 = N
# Kiem tra chu so nhap vao tu nguoi dung
bgt s0, zero, find_min
# Neu N < 0 thi in ra message_3
li a7, 55
la a0, message_3
li a1, 0
ecall
j end_main
```

Nếu như người dùng nhập sai yêu cầu đề bài ví dụ như nhập số âm hoặc số 0 thì sẽ báo lỗi và nhảy tới nhãn kết thúc chương trình:





- Đoạn chương trình sau để tìm chữ số nhỏ nhất bằng cách duyệt qua từng chữ số của N kiểm tra và cập nhật lại chữ số nhỏ nhất.

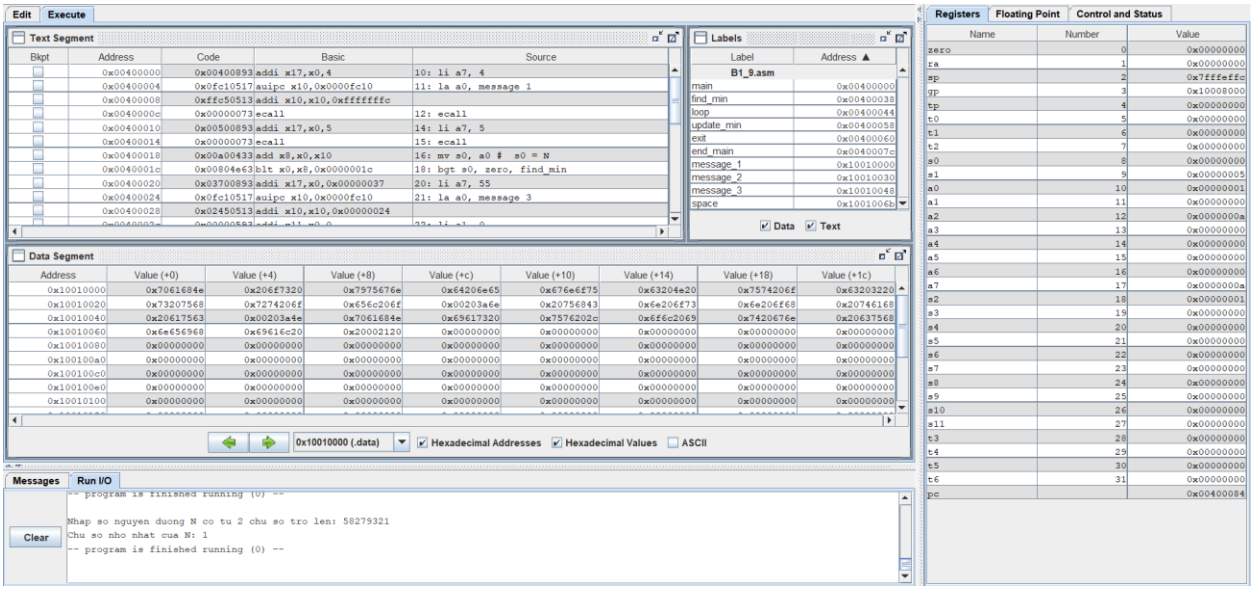
```
#------ Tim chu so nho nhat co trong N -------
find_min:
li a2, 10 # a2 = 10
li s2, 0x7fffffff # s2 duoc gan gia tri duong lon nhat
loop:
beqz s0, exit # Lap cho toi khi s0 = 0
rem s1, s0, a2 # s1 = s0 % 10
div s0, s0, a2 # s0 = s0 / 10
bge s2, s1, update_min # s2 > s1 thi cap nhat

j loop
# Kiem tra va cap nhat gia tri nho nhat
update_min:
mv s2, s1 # Cap nhat gia tri min
j loop
# ------ In ra chu so nho nhat va ket thuc chuong trinh -------
exit:
```

- Đoạn chương trình sau in ra chữ số nhỏ nhất của số N:

```
# ------ In ra chu so nho nhat va ket thuc chuong trinh -------
exit:
# In ra message_2
li a7, 4
la a0, message_2
ecall
# In ra chu so nho nhat
li a7, 1
mv a0, s2
ecall
# Ket thuc chuong trinh
end_main:
li a7, 10
ecall
```

Kết quả:

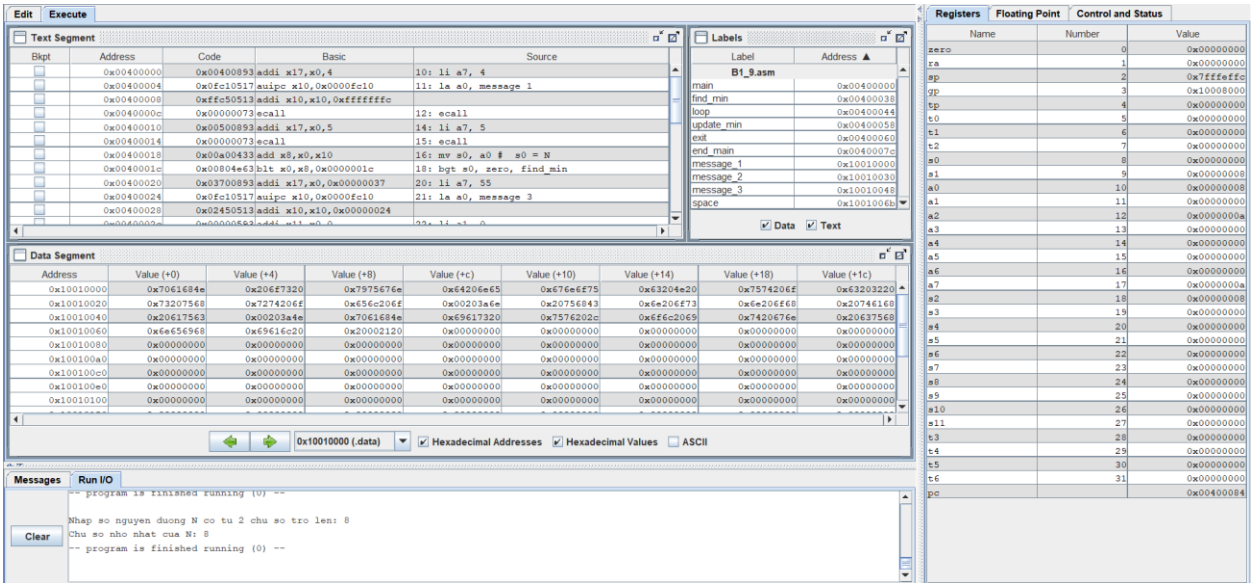
- Khi nhập số: 58279321
- Kết quả mong muốn: 1



Kết quả chương trình đúng như mong muốn.

Khi nhập chữ số có 1 chữ số thì chữ số nhỏ nhất là chính nó.

VD: N = 8 -> Chữ số nhỏ nhất của N: 8



Assignment B-3: Nhập mảng số nguyên từ bàn phím. In ra màn hình cặp phần tử liền kề có tích lớn nhất. Ví dụ: Nhập mảng [3, 6, -2, -5, 7, 3], cặp phần tử liền kề có tích lớn nhất là 7 và 3.

Code:

```
.data
A: .space 100
message_1: .asciz "Nhap so phan tu cua mang A: "
message_2: .asciz "Cap phan tu lien ke co tich lon nhat: "
space: .asciz " "
.text
main:
# In ra yeu cau nhap mang A
   li a7, 4
   la a0, message_1
   ecall

# Nhap so phan tu cua mang A
   li a7, 5
   ecall
   mv s0, a0  # Luu so phan tu vao s0

# Nhap mang A
   la s1, A  # Con tro dau mang
   li t0, 0  # Bien dem
```

```
loop:
   bge t0, s0, done  # Neu nhap du so phan tu thi thoat

   li a7, 5  # Nhap so nguyen
   ecall

   slli s2, t0, 2  # t0 * 4
   add s2, s2, s1  # Dia chi phan tu t0 trong A
   sw a0, 0(s2)    # Lưu gia tri vào mảng

   addi t0, t0, 1  # Tang bien dem
   j loop

done:
# Tim cap co tich lon nhat
   li t0, 0 # Bien dem
   li a5, -2147483648  # Gia tri tich nho nhat co the
   li s2, 0  # Luu phan tu thu nhat
   li s5, 0  # Luu phan tu thu hai

   la s1, A  # Con tro dau mang

find_max:
   addi t1, t0, 1 # t1 = t0 + 1
   bge t1, s0, exit  # Neu duyet het thi thoat

   lw a3, 0(s1)   # Lay phan tu t0
   lw a4, 4(s1)   # Lay phan tu t0 + 1

   mul s3, a3, a4 # Tinh tich

# Neu tich lon hon a5, cap nhat gia tri
   bge s3, a5, update

# Cap nhat con tro va bien dem
next:
   addi s1, s1, 4  # Di chuyen con tro den phan tu tiep theo
   addi t0, t0, 1  # Tang bien dem
   j find_max

update:
   mv a5, s3  # Cap nhat tich lon nhat
   mv s2, a3  # Cap nhat phan tu thu nhat
   mv s5, a4  # Cap nhat phan tu thu hai
```

```
    j next

exit:
# In kết quả va ket thuc chuong trinh
    li a7, 4
    la a0, message_2
    ecall

    li a7, 1
    mv a0, s2  # In phan tu thu nhat
    ecall

    li a7, 4
    la a0, space
    ecall

    li a7, 1
    mv a0, s5  # In phan tu thu hai
    ecall

    li a7, 10  # Ket thuc chuong trinh
    ecall
```

Phân tích:

- Nhập số phần tử của mảng A
- Duyệt qua từng phần tử i và tính tích phần tử A[i] và phần tử A[i + 1] nếu lớn hơn max hiện tại thì cập nhật max và 2 phần tử kề.

Giải thích:

- Đoạn chương trình sau in ra message_1 và yêu cầu người dùng nhập số phần tử của mảng A

```
main:
# In ra yeu cau nhap mang A
    li a7, 4
    la a0, message_1
    ecall

# Nhap so phan tu cua mang A
    li a7, 5
    ecall
    mv s0, a0  # Luu so phan tu vao s0
```

- Nhập từng phần tử và lưu vào mảng A tới khi bằng số phần tử đã nhập ở trên:

```
# Nhap mang A
    la s1, A  # Con tro dau mang
    li t0, 0  # Bien dem

loop:
    bge t0, s0, done  # Neu nhap du so phan tu thi thoat

    li a7, 5  # Nhap so nguyen
    ecall

    slli s2, t0, 2  # t0 * 4
    add s2, s2, s1  # Dia chi phan tu t0 trong A
    sw a0, 0(s2)    # Luu gia tri vào mảng

    addi t0, t0, 1  # Tang bien dem
    j loop
```

- Đoạn chương trình sau duyệt qua từng phần tử và tính tích phần tử đó và phần tử kế tiếp rồi cập nhật để tìm tích lớn nhất

```
done:
# Tim cap co tich lon nhat
    li t0, 0 # Bien dem
    li a5, -2147483648  # Gia tri tich nho nhat co the
    li s2, 0  # Luu phan tu thu nhat
    li s5, 0  # Luu phan tu thu hai

    la s1, A  # Con tro dau mang

find_max:
    addi t1, t0, 1 # t1 = t0 + 1
    bge t1, s0, exit  # Neu duyet het thi thoat

    lw a3, 0(s1)   # Lay phan tu t0
    lw a4, 4(s1)   # Lay phan tu t0 + 1

    mul s3, a3, a4 # Tinh tich

# Neu tich lon hon a5, cap nhat gia tri
    bge s3, a5, update

# Cap nhat con tro va bien dem
next:
    addi s1, s1, 4  # Di chuyen con tro den phan tu tiep theo
    addi t0, t0, 1  # Tang bien dem
    j find_max
```

```
update:
    mv a5, s3  # Cap nhat tich lon nhat
    mv s2, a3  # Cap nhat phan tu thu nhat
    mv s5, a4  # Cap nhat phan tu thu hai
    j next
```

- In ra cặp phần tử có tích lớn nhất:

```
exit:
# In kết quả va ket thuc chuong trinh
    li a7, 4
    la a0, message_2
    ecall

    li a7, 1
    mv a0, s2  # In phan tu thu nhat
    ecall

    li a7, 4
    la a0, space
    ecall

    li a7, 1
    mv a0, s5  # In phan tu thu hai
    ecall

    li a7, 10  # Ket thuc chuong trinh
    ecall
```

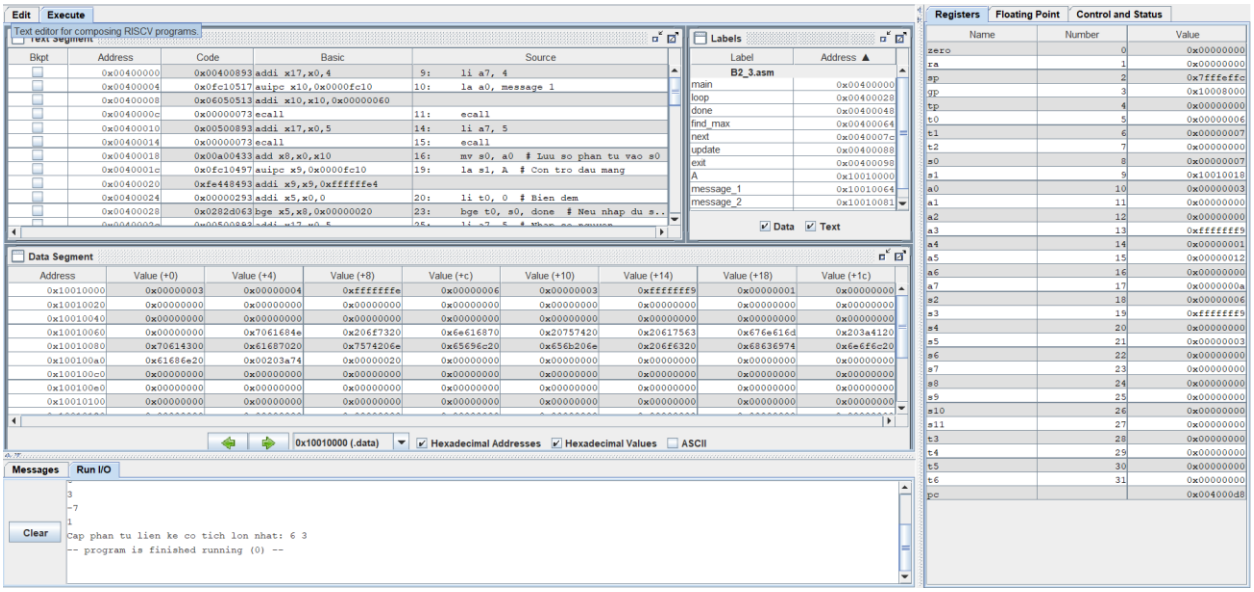
Kết quả:

Input: n = 7

3 4 -2 6 3 -7 1

Kết quả mong muốn: 6 3

Kết quả khi chạy chương trình:



Đúng như mong muốn: 6 3

Assignment C-12: Nhập vào 2 xâu ký tự A và B. In ra màn hình các ký tự chữ cái thường chỉ có trong A và không có trong B.

Code:

```
.data
A: .space 100
B: .space 100
message_1: .asciz "INPUT: \n"
message_2: .asciz "OUTPUT: \n"
A_input: .asciz "Nhap xau A: "
B_input: .asciz "Nhap xau B: "
space: .ascii " "
.text
main:

# INPUT

li a7, 4
la a0, message_1
ecall
#--------Nhap xau 1---------
# In ra thong bao nhap xau A
la a0, A_input
ecall
# Nhap xau A:
li a7, 8
```

```
la a0, A
li a1, 100
ecall
#-------Nhap xau 2----------
# In ra thong bao nhap xau B
li a7, 4
la a0, B_input
ecall
# Nhap xau B:
li a7, 8
la a0, B
li a1, 100
ecall

#------------------------------

# OUTPUT
li a7, 4
la a0, message_2
ecall

la s1, A # Lay dia chi xau A
li s10, 10
check_A:
lb t0, 0(s1) # Lay gia tri tai dia chi s1
beq t0, s10, done # Neu gia tri la ky tu xuong dong tuc da duyet xong xau
beqz  t0, done # Neu gia tri la ky tu rong tuc da duyet xong xau
li t1, 'a'
li t2, 'z'
blt t0, t1, next_A # Neu nhu nam ngoai a-z thi chuyen toi phan tu tiep theo
bgt t0, t2, next_A
# Kiem tra ky tu co trong xau B
la s2, B # Lay dia chi xau B
check_B:
lb t3, 0(s2) # Lay gia tri tai dia chi s2
beq t0, t3, next_A # Neu co bo qua xet ky tu dang sau
beq t3, s10, print # Neu gap ky tu xuong dong thi da duyet het xau
beqz t3, print # Neu duyet het ma khong thay thi in ra man hinh
addi s2, s2, 1 # s2 = s2 + 1
j check_B
next_A:
addi s1, s1, 1 # s1 = s1 + 1
j check_A
# In ky tu thuong khong xuat hien trong B
```

```
print:
li a7, 11
mv a0, t0
ecall
# In dau cach
li a7, 4
la a0, space
ecall

j next_A
# Ket thuc chuong trinh
done:
li a7, 10
ecall
```

Phân tích:

- Nhập vào 2 xâu A, B
- Lặp qua từng ký tự của xâu A, với mỗi ký tự duyệt qua xâu B nếu như ký tự thường của A nằm trong B thì xét tới ký tự tiếp theo còn duyệt hết xâu B mà không thấy thì in ra màn hình.

Giải thích:

- Đoạn chương trình sau yêu cầu người dùng nhập xâu A, B

```
# INPUT

li a7, 4
la a0, message_1
ecall
#--------Nhap xau 1---------
# In ra thong bao nhap xau A
la a0, A_input
ecall
# Nhap xau A:
li a7, 8
la a0, A
li a1, 100
ecall
#-------Nhap xau 2----------
# In ra thong bao nhap xau B
li a7, 4
la a0, B_input
ecall
# Nhap xau B:
li a7, 8
la a0, B
li a1, 100
ecall
```

- Xét từng ký tự của mảng A, ứng với mỗi phần tử duyệt qua mảng B để kiểm tra ký tự thường đó có nằm trong mảng B không

```
# Xet cac ky tu
li a7, 4 # In ra message_2
la a0, message_2
ecall

la s1, A # Lay dia chi xau A
li s10, 10
check_A:
lb t0, 0(s1) # Lay gia tri tai dia chi s1
beq t0, s10, done # Neu gia tri la ky tu xuong dong tuc da duyet xong xau
beqz  t0, done # Neu gia tri la ky tu rong tuc da duyet xong xau
li t1, 'a'
li t2, 'z'
blt t0, t1, next_A # Neu nhu nam ngoai a-z thi chuyen toi phan tu tiep theo
bgt t0, t2, next_A
# Kiem tra ky tu co trong xau B
la s2, B # Lay dia chi xau B
check_B:
lb t3, 0(s2) # Lay gia tri tai dia chi s2
beq t0, t3, next_A # Neu co bo qua xet ky tu dang sau
beq t3, s10, print # Neu gap ky tu xuong dong thi da duyet het xau
beqz t3, print # Neu duyet het ma khong thay thi in ra man hinh
addi s2, s2, 1 # s2 = s2 + 1
j check_B
next_A:
addi s1, s1, 1 # s1 = s1 + 1
j check_A
```

- Với các phần tử thuộc xâu A mà không thuộc xâu B thì in ra màn hình:

```
# OUTPUT
# In ky tu thuong khong xuat hien trong B
print:
li a7, 11
mv a0, t0
ecall
# In dau cach
li a7, 4
la a0, space
ecall

j next_A
# Ket thuc chuong trinh
done:
li a7, 10
ecall
```

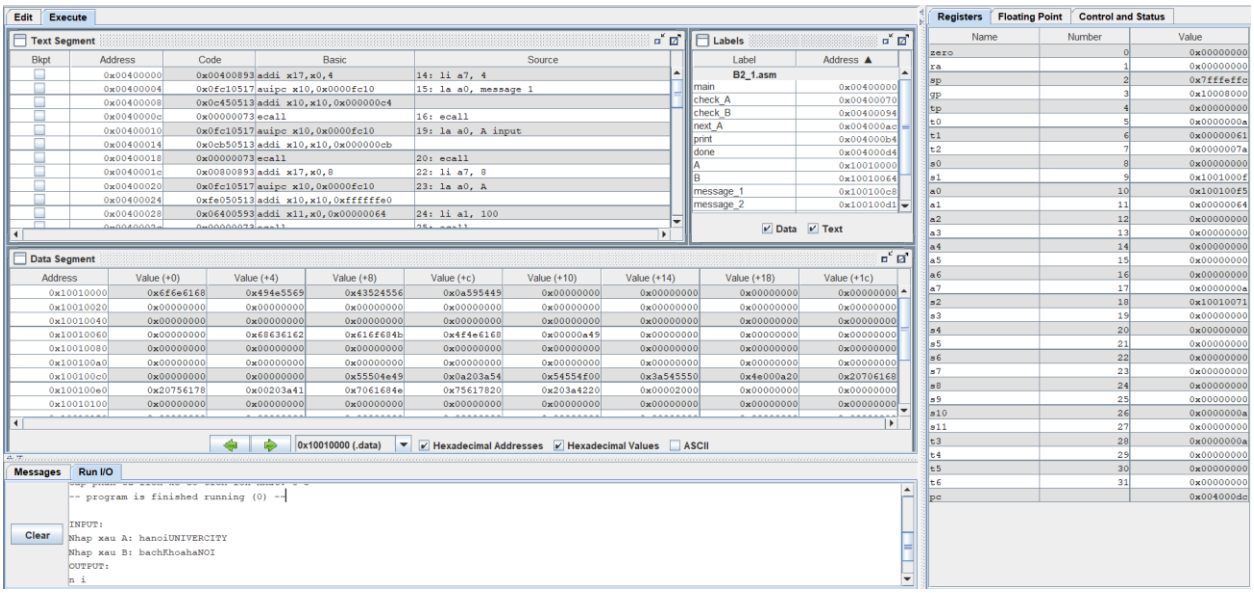
Kết quả:

INPUT: A: hanoiUNIVERCITY

B: bachKhoahaNOI

Ta thấy ký tự n, i của xâu A không có trong xâu B nên output mong muốn là: n i

Kết quả khi chạy chương trình:



Kết quả như mong muốn.