Bài 3a.

1. **Không có điều kiện.**

```
.data
        character: .word 1
        out: .asciiz "\nNhap ky tu (chi 1 ky tu):"
        out1: .asciiz "\nKy tu truoc:"
        out2: .asciiz "\nKy tu sau:"
.text
```

Khai báo các nhãn cần thiết.

```
li $v0, 4
la $a0, out
syscall
li $v0, 8
la $a0, character
la $a1, 2
syscall
move $t0, $a0
```

- load cho $v0 = 4
- Load $a0 = out
- Khi dung lệch syscall sẽ hiện "Nhap ky tu (chi 1 ky tu):
- 4 dòng sau tương tự dung để nhập 1 kí tự bất kỳ vào thang ghi $a0
- Vì thanh ghi $a0 lúc sau còn dung để nhập nhiều việc khác nên move $t0, $a0 để sao chép dữ liệu thanh ghi $a0 qua thanh ghi $t0.

Lúc này thanh ghi $t0 sẽ chứa giá trị của 1 ký tự mà chúng ta nhập vào.

```
lw $t1, 0($t0)
sub $t1, $t1, 1
addi $t2, $t1, 2
        # Xuat ki tu lien truoc
        li $v0, 4
        la $a0, out1
        syscall
        sw $t1, 0($t0)
        move $a0, $t0
        syscall

        # Xuat ky tu lien sau
        li $v0, 4
        la $a0, out2
        syscall
        sw $t2, 0($t0)
        move $a0, $t0
        syscall
```

- load store thanh ghi $t0 lên $t1, lúc nào thanh ghi $t1, sẽ chưa giá trị của kí tự ta nhập vào trong bảng mã ASCII.
- Sub $t1, $t1, 1: lấy ký tự trước của ký tự chúng ta nhập vào.
- Addi $t1, $t2, 2: Vì ở trên chúng ta trừ $t1 cho 1 rùi nên bây giờ muốn lấy ký tự sau ký tư chúng ta nhập vào thì phải cộng cho 2.

- Tiếp theo, xuất ra các ký tự đó. Thực hiện theo cú pháp xuất ra các nhãn out1, và out2 để cho rõ rang.

- Tiếp theo là xuất ra các ký tự

- Mà xuất ký tự thì xuất các giá trị trong thanh ghi $a0 nên ta phải move giá trị sanh thanh ghi $a0.

2. Có điều kiện.

```
.data
        character: .word 1
        out: .asciiz "\nNhap ky tu (chi 1 ky tu):"
        out1: .asciiz "\nKy tu truoc:"
        out2: .asciiz "\nKy tu sau:"
        out_else: .asciiz "\ninvalid type"
.text
```

- Vẫn là khai báo các nhãn cần thiết, như phần một.

```
    lw $t1, 0($t0)
    sub $t1, $t1, 1
    addi $t2, $t1, 2
            # Xuat ki tu lien truoc
            li $v0, 4
            la $a0, out1
            syscall
            sw $t1, 0($t0)
            move $a0, $t0
            syscall

            # Xuat ky tu lien sau
            li $v0, 4
            la $a0, out2
            syscall
            sw $t2, 0($t0)
            move $a0, $t0
            syscall

    j exit

    else:
    li $v0, 4
    la $a0, out_else
    syscall
```

Việc này giống như phần 1 chỉ khác nhau về điều kiện.

- Nếu điều kiện thoả như đề bài yêu cầu thì sẽ thực hiện như phần 1, và sau đó kết thúc chương trình bằng cách nhảy đến nhãn exit ở cuối chương trình bằng nhãn j exit.
- Nếu điều kiện không thoả thì sẽ xuất ra nhãn out_else “invalid type”, và sau đó kiết thúc chương trình.

Phần giải quyết điều kiện

```
lw $s0, 0($a0)
li $s1, 47
slt $s1, $s1, $s0
li $s2, 58
slt $s2, $s0, $s2
and $s3, $s1, $s2
li $s1, 64
slt $s1, $s1, $s0
li $s2, 91
slt $s2, $s0, $s2
and $s4, $s1, $s2
li $s1, 96
slt $s1, $s1, $s0
li $s2, 123
slt $s2, $s0, $s2
and $s5, $s1, $s2
or $s6, $s3, $s4
or $s6, $s6, $s5

beq $s6, $zero, else
```

- Cho $s0 bằng giá trị chúng ta nhập ở trên.

Load $s1, 47 trong bảng mã ASCII 0 = 48

Load $s2, 58 trong bảng mã ASCII 9 = 57

Và ta so sánh nó bằng lệnh slt.

Sau đó dung lệnh and.

Kết hợp 2 điều trên để kiểm tra xem $s0 có nằng trong 48 đến 57.

Tương tương: Kiểm tra xem $s0 có nằng trong 65 đến 90 không và $s0 có nằm trong 97 đến 122 không.

Or $s6, $s3, $s4

Or $s6, $s6, $s5

Dùng kiểm tra xem $s0 có nằm bất kì nằm trong 3 vị trí không? Nến có $s6 có giá trị bằng 1 và ngược lại $s0 không nằm trong vùng nào thì $s6 có giá trị bằng 0.

Beq $s6, $zero, else. Nếu $s6 có giá trị bằng 0 thì nhảy tời nhãn else túc là in ra invalid type. Ngược lại thì làm như phần 1.

Câu 3c:

```
li $v0, 4
la $a0, in
syscall
li, $v0, 5
syscall
move $t0, $v0
li $v0, 4
la $a0, in
syscall
li $v0, 5
syscall
move $t1, $v0
```

- Phần này dung để nhập vào 2 số nguyên vào thanh ghi $a0, mà thanh ghi $a0 còn dung để nhập nữa nên phải di chuyển giá trị sang lần lượt là thanh ghi $t0 và $t1 bằng lệnh:

Move $t0, $a0 và move $t1, $a0.

```
# compare
li $v0, 4
la $a0, com
syscall
slt $s0, $t1, $t0
beq $s0, $zero, label
        li $v0, 1
        add $a0, $zero, $t0
        syscall
        j exit
label:
        li $a0, 0
        li $v0, 1
        add $a0, $zero, $t1
        syscall
 exit:
```

- Phần đầu tiên sẽ là compare 2 số nguyên này.
- Dùng lệnh slt và beq tương tự như if else.
- Slt $s0, $t1, $t0 nếu $t1 < $t0 thì $s0 = 0, ngược lại thì $s0 = 1.
- Dùng beq $s0, $zero, label.

Nếu $s0 = $zero tức là $t1 < $t0 thì thực hiện nhãn label xuất ra $t1 (Xuất số lớn hơn)

Nếu $s0 != $zero tức $t1 >= $t0 thực hiện hiện tiếp xuất $t0 đến j exit thì dừng và kết thúc chương trình. (ở đây $t1 = $t0 thì xuất giá trị thanh ghi nào cũng được nên mặc định là xuất $t0).

```
# add
li $v0, 4
la $a0, sum
syscall
add $s1, $t0, $t1
li $v0, 1
li $a0, 0
add, $a0, $zero, $s1
syscall
```

- Công hai số nguyên được lưu trong thanh ghi $t0 và $ t1 bằng cách dung lệnh add $s1, $t0, $t1.
- Sau đó move $a0, $s1 lúc vì lệnh syscall xuất số nguyên thì phải ở thanh ghi $a0.

```
# sub
li $v0, 4
la $a0, diff
syscall
sub $s2, $t0, $t1
li $v0, 1
li $a0, 0
add, $a0, $zero, $s2
syscall
```

- Ta trừ giá trị trong thanh ghi $t0 không cho $t1 bằng lệnh sub $s2, $t0, $t1 sau đó lưu vào thanh ghi $s2.
- Sau đó phải move giá trị của $s2 qua thanh ghi $a0 rùi dung syscall để xuất.

```
# mul
li $v0, 4
la $a0, mull
syscall
mul $s3, $t0, $t1
li $v0, 1
li $a0, 0
add, $a0, $zero, $s3
syscall
```

- Dùng lệnh mul $s3, $t0, $t1 để nhân giá trị trong 2 thanh ghi $t0 và $t1 vào lưu vào $s3.
- Sau đó phải move giá trị của thanh ghi $s3 qua thanh ghi $a0 rùi dung syscall để xuất.

```
#div
li $v0, 4
la $a0, divi
syscall
div $s4, $t0, $t1
li $v0, 1
li $a0, 0
add, $a0, $zero, $s4
syscall
```

- Dùng lệnh div $s4, $t0, $t1 để lấy giá trị trong tanh ghi $t0 chia cho giá trị trong thanh ghi $t1 sau đó lưu vào thanh ghi $s4.
- Sau đó phải move giá trị của thanh ghi $s4 qua thanh ghi $a0 rùi dung syscall để xuất.

Câu 3b:

```
funerror:
        li $v0, 4
        la $a0, error
        syscall
        li $v0, 10
        syscall

countnumberofdigit:
        li $a0, 0
        while:
                div $a1, $a1, 10
                addi $a0, $a0, 1
                bnez $a1, while
        jr $ra
```

- Hàm funerror: dung để trả về chuỗi của nhãn error “invalid Entry” khi số chúng ta nhập vào không thoả mãn yêu cầu đề bài.
- Đoạn li $v0, syscall là để kết thúc chương trình.
- Coutnumberofdigit: đùng để đếm số lượng chữ số chúng ta nhập vào.
- Cuối hàm chúng ta nhận lệnh jr $ra để quay ngược lại address chúng ta gọi hàm. Hay tức là address mà thanh ghi $ra đang chứa.

```
str: #switch case:
        beq $a1, 0, zero_out
        beq $a1, 1, one_out
        beq $a1, 2, two_out
        beq $a1, 3, three_out
        beq $a1, 4, four_out
        beq $a1, 5, five_out
        beq $a1, 6, six_out
        beq $a1, 7, seven_out
        beq $a1, 8, eight_out
        beq $a1, 9, nine_out
        zero_out:
                la $a0, zero
                j out
        one_out:
                la $a0, one
                j out
        two_out:
                la $a0, two
                j out
        three_out:
                la $a0, three
                j out
        four_out:
                la $a0, four
                j out
        five_out:
                la $a0, five
                j out
        six_out:
                la $a0, six
                j out
        seven_out:
                la $a0, seven
                j out
        eight_out:
                la $a0, eight
                j out
        nine_out:
                la $a0, nine
        out:
                li $v0, 4
                syscall
        jr $ra
```

- Hàm str tương tự switch case trong C.

- Đối số vào là $a0 sẽ là các chữ số của số nhập vào và thông quà hàm này chuyển thành dạng chữ vd: One, Two, Three,…
- Vì đây là số có ba chữ số nên chỉ cần lặp 3 lần là được.

```
main:
        li $v0, 5
        syscall
        move $s1, $v0
        # so sanh voi $zero
        bltz $v0, funerror
        # xem co phai la so 3 chu so ko
        addi $a1, $s1, 0
        jal countnumberofdigit
        blt $a0, 3, funerror
        #luu tuong chu so vao 3 vung nho $s0, $s1, $s2
        li $s2, 10
        div $s1, $s1, $s2
        mfhi $t2
        div $s1, $s1, $s2
        mfhi $t1
        div $s1, $s1, $s2
        mfhi $t0
        #Print res
        addi $a1, $t0, 0
        jal str
        addi $a1, $t1, 0
        jal str
        addi $a1, $t2, 0
        jal str
        #end
        li $v0, 10
        syscall
```

- Đây phần chính của chương trình.
  3 dòng đầu là nhập và di chuyển giá trị của thanh ghi $v0 sang $s1 để sử dụng.

- Ờ phần tiếp theo, dùng giá trị đó xem có thoả điều kiện không.
  Phải là số có ba chữ số và không được là số âm, rồi đưa tới các hàm tương ứng.
- Tiếp theo, lấy các chữ số ta nhập vào bằng cách chia lấy phần dư. Khi chia phần dư sẽ được đưa vào thanh ghi (hi) và ta dùng lệnh mfhi $[thanh ghi muốn đưa giá trị vào], để lấy giá trị đó. Bằng cách này như chương trình ta sẽ lấy được các chữ số của số ta nhập vào đưa vào thanh ghi lần lượt là $t0, $t1, $t2.
- Và đưa đối số vào thanh ghi $a1 ta dung hàm str để đưa ra kết quả.