BÁO CÁO BÀI THỰC HÀNH LAB 4

Môn: Tổ chức & Cấu trúc máy tính II – IT012.L14.1

Tên: Nguyễn Đại Kỳ
Mssv: 19521731

3a. Nhập vào một ký tự, xuất ra cửa sổ I/O của MARS ký tự liền trước và liền sau của ký tự đó:

```
.data
string1:        .asciiz "\nNhap mot ki tu: "
string2:        .asciiz "\nKi tu truoc: "
string3:        .asciiz "\nKi tu sau: "
string4:        .asciiz "\ninvalid type"
```

Khai báo các label cần thiết cho chương trình.

```
.text
Nhap:
        li $v0, 4
        la $a0, string1
        syscall
        li $v0, 12
        syscall
        move $t1, $v0
```

- li $v0, 4: load $v0 = 4 cho mục đích print string.
- la $a0, string1: load string1 vào $a0 để print ra console.
- syscall: sau lệnh này thì dòng chữ: “Nhap mot ki tu: “ được print ra console.
- li $v0, 12: load $v0 = 12 cho mục đích nhập kí tự.
- syscall: lệnh này cho phép user nhập vào.
- move $t1, $v0: chuyển dữ liệu vừa nhập từ $v0 qua $t1 để tiện cho việc tính toán sử lí sau này.

```
Check:  #check valid type
        li $t2, 58
        li $t3, 47
        slt $t0, $t3, $t1
        beq $t0, 1, And1
re1:    li $t2, 64
        li $t3, 91
        slt $t0, $t2, $t1
        beq $t0, 1, And2
re2:    li $t2, 96
        li $t3, 123
        slt $t0, $t2, $t1
        beq $t0, 1, And3
re3:    li $v0, 4
        la $a0, string4
        syscall
        j End

And1:   slt $t4, $t1, $t2
        beq $t4, 1, Print
        j re1

And2:   slt $t4, $t1, $t3
        beq $t4, 1, Print
        j re2

And3:   slt $t4, $t1, $t3
        beq $t4, 1, Print
        j re3
```

Sau khi nhập thì kí tự đã được lưu ở $t1, ta tiến hành kiểm tra tính hợp lệ của dữ liệu.
Kiểm tra dữ liệu nằm trong kí tự số:
- li $t2, 58 và li $t3, 47: để kiểm tra dữ liệu nằm trong khoảng số (mã ASCII: mã 47 trước 0, mã 58 sau số 9).
- slt $t0, $t3, $t1: dùng để so sánh $t1 (chứa kí tự được lưu dưới dạng mã ascii) và $t3 (chứa số 47 đổi qua mã ascii là số đứng trước kí tự 0). Nếu $t3 < $t1 thì $t0 = 1, ngược lại $t0 = 0.
- beq $t0, 1, And1: dùng để nhảy đến nhãn And1(chứa điều kiện 2) nếu $t0 = 1.
- slt $t4, $t1, $t2: dùng để so sánh $t1 và $t2 (chứa số 58 đổi qua mã ascii là kí tự đứng sau ‘9’). Nếu $t1 < $t2 thì $t4 = 1, ngược lại $t4 = 0.
- beq $t4, 1, Print: Nếu $t4 = 1 thì dữ liệu kiểm tra đã thỏa yêu cầu (nằm trong đoạn từ ‘0’ đến ‘9’), ta nhảy đến nhãn Print.
- j re1: Nếu dữ liệu không thỏa thì ta nhảy về nhãn re1 để thực hiện so sánh với các điều kiện khác.

Kiểm tra kí tự thường (a-z) và kí tự hoa (A-Z) ta thực hiện so sánh bằng mã ascii tương tự như với kí tự số.
Dữ liệu nào không nằm trong 3 dạng trên thì bị loại. Thì không hợp lệ xuất ra màn hình thông báo và kết thúc chương trình:
- li $v0, 4 la $a0, string4 syscall: dùng để print ra thông báo: “invalid type” cho user.
- j End: kết thúc chương trình.

```
Print:
        addi $t2, $t1, -1
        addi $t3, $t1, 1
        #kiem tra dau cuoi
        beq $t1, 48, out1
        beq $t1, 65, out1
        beq $t1, 97, out1
        beq $t1, 57, out2
        beq $t1, 90, out2
        beq $t1, 122, out2
        li $v0, 4
        la $a0, string2
        syscall
        li $v0, 11
        move $a0, $t2
        syscall
out1:   li $v0, 4
        la $a0, string3
        syscall
        li $v0, 11
        move $a0, $t3
        syscall
        j End

out2:   li $v0, 4
        la $a0, string2
        syscall
        li $v0, 11
        move $a0, $t2
        syscall
        j End
End:
```

Sau khi kiểm tra dữ liệu, nếu hợp lệ thì nhảy đến label Print để xuất dữ liệu:
- addi $t2, $t1, -1 và addi $t3, $t1, 1 để lấy kí tự ascii liền trước và liền sau của kí tự hiện tại.
- 6 dòng tiếp theo để kiểm tra các kí tự đầu và cuối. Nếu kí tự đầu tiên (‘0’, ‘a’, ‘A’) thì chỉ print ra kí tự liền sau (nhảy tới label out1). Nếu kí tự cuối (‘9’, ‘z’, ‘Z’) thì chỉ print ra kí tự liền trước (nhảy tới label out2).

```
Nhap mot ki tu: Z
Ki tu truoc: Y
```

- Còn nếu chỉ là kí tự bình thường thì ta thực hiện print tuần tự cả kí tự liền trước và liền sau.
- Sau đó j End và kết thúc chương trình.

```
Nhap mot ki tu: 6
Ki tu truoc: 5
Ki tu sau: 7
```

```
Nhap mot ki tu: g
Ki tu truoc: f
Ki tu sau: h
```

```
Nhap mot ki tu: U
Ki tu truoc: T
Ki tu sau: V
```

3b. Nhập vào một số nguyên dương, xuất ra cửa sổ I/O của MARS cách đọc số đó:

```
.data
        zero:   .asciiz "Zero "
        one:    .asciiz "One "
        two:    .asciiz "Two "
        three:  .asciiz "Three "
        four:   .asciiz "Four "
        five:   .asciiz "Five "
        six:    .asciiz "Six "
        seven:  .asciiz "Seven "
        eight:  .asciiz "Eight "
        nine:   .asciiz "Nine "

        str:    .asciiz "Nhap so: "
        error:  .asciiz "Invalid Entry"
        error2: .asciiz "Number of digit is less or equal 3!"
```

Khai báo các label dữ liệu cần thiết cho chương trình.

```
.text
main:   #input integer
        li $v0, 4
        la $a0, str
        syscall
        li $v0, 5
        syscall
        move $t1, $v0

        #check positive integer
        li $t0, 0
        slt $t2, $t1, $t0
        beq $t2, 1, PrintError1

        #check number of digit
        li $t2, 0
        addi $t3, $t1, 0
loop:   beq $t3, 0, finish_loop
        div $t3, $t3, 10
        addi $t2, $t2, 1
        j loop
finish_loop:
        li $t7, 3
        slt $t6, $t7, $t2
        beq $t6, 1, PrintError2

        #plit digit and print integer
plit:   div $t1, $t1, 10
        mfhi $s1
        div $t1, $t1, 10
        mfhi $s2
        div $t1, $t1, 10
        mfhi $s3
```

* Nhập số:
- li $v0, 4 la $a0, str syscall: dùng để xuất ra dòng chữ "Nhap so: " cho user.
- li $v0, 5 syscall move $t1, $v0: dùng để nhập vào một số nguyên và chuyển nó vào $t1 để tiện sử dụng sau này.
* Kiểm tra số nguyên dương:
- li $t0, 0 slt $t2, $t1, $t0: dùng để so sánh giá trị trong $t1 với 0. Nếu $t1 < 0 thì $t2 = 1, ngược lại $t2 = 0.
- beq $t2, 1, PrintError1: Nếu kết quả $t2 = 1 thì nhảy đến PrintError. Nếu không thì tiếp tục thực hiện chương trình.
* Kiểm tra số lượng chữ số < 3:
- li $t2, 0: khởi tạo $t2 = 0 làm biến đếm.
- addi $t3, $t1, 0: Khởi tạo $t3 = $t1, để dùng cho phép vòng lặp đếm.
- beg $t3, 0, finish_loop: Đây là điều kiện dừng của vòng lặp. Nếu $t3 = 0 thì nhảy đến label finish_loop.
- div $t3, $t3, 10: thực hiện phép chia 10, để lấy ra chữ số cuối.
- addi $t2, $t2, 1: mỗi lần lặp thì tăng $t2 lên 1 để đếm số digit của integer.
- li $t7, 3 slt $t6, $t7, $t2 beq $t6, 1, PrintError2: đề kiểm tra nếu $t2 > 3 thì nhảy đến label PrintError2.
* Chia để lấy ra từng chữ số của integer:
- div $t1, $t1, 10: chia $t1 cho 10 để lấy chữ số cuối, kết quả lưu ngược lại vào $t1.
- mfhi $t1: Sau khi chia thì số dư được lưu trong thanh ghi $r, mfhi $t1 giúp lấy chữ số cuối trong $ra ra ngoài $s1.
Tương tự với các chữ số thứ 2, thứ 1 lưu trong $s2, $s3.

```
        #print integer
        beq $t2, 1, OneDigit
        beq $t2, 2, TwoDigit
        addi $a1, $s3, 0
        jal PrintNumber
TwoDigit:
        addi $a1, $s2, 0
        jal PrintNumber
OneDigit:
        addi $a1, $s1, 0
        jal PrintNumber
        j Exit

        #Function
PrintError1:
        li $v0, 4
        la $a0, error
        syscall
        j Exit

PrintError2:
        li $v0, 4
        la $a0, error2
        syscall
        j Exit
```

Sau khi tách các chữ số ra thì ta tiến hành print từng chữ ra console:
- 2 dòng beq đầu để phân loại cách in dựa trên số lượng chữ số:
+ Nếu số có 1 chữ số thì nhảy tới OneDigit:
vd: 3 => 'three'
+ Nếu số có 2 chữ số thì nhảy tới TwoDigit:
vd: 34 => 'three four'
+ Nếu số có 3 chữ số thì thực hiện tuần tự từ trên xuống dưới.
- addi $a1, $s3, 0: truyền giá trị $s3 vào $a1, để làm tham số cho hàm PrintNumber. Tương tự với các $s2, $s1.
- jal PrintNumber: nhảy đến hàm PrintNumber và tự động lưu địa chỉ để lát nữa thực hiện xong hàm thì quay lại đúng vị trí này để thực hiện tiếp những câu lệnh tiếp theo.

* Các hàm PrintError1, PrintError2 dùng để print ra console các dòng chữ “Invalid Entry” và “Number of digit is less or equal 3!”. Sau đó nhảy đến Exit và kết thúc chương trình.

```
Nhap so: -34
Invalid Entry
```

```
Nhap so: 1234
Number of digit is less or equal 3!
```

```
PrintNumber:
        li $v0, 4
        beq $a1, 0, Printzero
        beq $a1, 1, Printone
        beq $a1, 2, Printtwo
        beq $a1, 3, Printthree
        beq $a1, 4, Printfour
        beq $a1, 5, Printfive
        beq $a1, 6, Printsix
        beq $a1, 7, Printseven
        beq $a1, 8, Printeight
        beq $a1, 9, Printnine
        Printzero:
                la $a0, zero
                syscall
                j out
        Printone:
                la $a0, one
                syscall
                j out
        Printtwo:
                la $a0, two
                syscall
                j out
        Printthree:
                la $a0, three
                syscall
                j out
        Printfour:
                la $a0, four
                syscall
                j out
                j out
        Printfive:
                la $a0, five
                syscall
                j out
        Printsix:
                la $a0, six
                syscall
                j out
        Printseven:
                la $a0, seven
                syscall
                j out
        Printeight:
                la $a0, eight
                syscall
                j out
        Printnine:
                la $a0, nine
                syscall
                j out
        out:    jr $ra

Exit:
```

* Hàm PrintNumber:
- li $v0, 4: load $v0 = 4 để chuẩn bị cho việc print string ra console.
- Các dòng beq tiếp theo đóng vai trò như switch-case, để so sánh và gọi vào hàm Print tương ứng với từng số.
- Khi nhảy vào hàm Print, thì tiến hành load các label tương ứng và print ra console.
- j out: nhảy đến cuối hàm PrintNumber.

- jr $ra: lệnh nhảy về vị trí gọi hàm ban đầu (jal PrintNumber) ở phía trên.
* Sau khi thực hiện print ra tất cả các digit của integer, đã thỏa yêu cầu đề bài, ta nhảy đến Exit và kết thúc chương trình.
Ta được kết quả chương trình như sau:

```
Nhap so: 379
Three Seven Nine
-- program is finished running (dropped off bottom) --
```

3c. Nhập vào 2 số nguyên, in ra cửa sổ I/O của MARS max, tổng, hiệu, tích, thương:

```
.data
        in1: .asciiz "Nhap so nguyen thu nhat vao: "
        in2: .asciiz "Nhap so nguyen thu hai vao: "
        out: .asciiz "So lon hon la: "
        sum: .asciiz "Tong: "
        dif: .asciiz "Hieu: "
        multi: .asciiz "Tich: "
        divi: .asciiz "Thuong "
        newline:.asciiz "\n"
```

Khai báo các label cần thiết cho chương trình.

```
.text
main:   #input 2 integer
        li $v0, 4
        la $a0, in1
        syscall
        li $v0, 5
        syscall
        move $t0, $v0
        li $v0, 4
        la $a0, in2
        syscall
        li $v0, 5
        syscall
        move $t1, $v0

        #Compare 2 integer
        li $v0, 4
        la $a0, out
        syscall
        li $v0, 1
        slt $t2, $t0, $t1
        beq $t2, 1, out0
        move $a0, $t0
        j out1
out0:
        move $a0, $t1
out1:
        syscall
        li $v0, 4
        la $a0, newline
        syscall
```

* Nhập 2 số nguyên:
- li $v0, 4 la $a0, in1 syscall: dùng để print ra dòng chữ “Nhap so nguyen thu nhat vao: ”.
- li $v0, 5 syscall move $t0, $v0: dùng để nhập một số và lưu số đó vào $t0.
Tương tự với số thứ hai được lưu vào $t1.

* So sánh 2 số nguyên:
- li $v0, 4 la $a0, out syscall: dùng để xuất ra dòng “So lon hon la: ”.
- li $v0, 1: chuẩn bị print số nguyên.
- slt $t2, $t0, $t1: dùng để so sánh $t0 và $t1. Nếu $t0 < $t1 thì $t2 = 1, ngược lại $t2 = 0.
- beq $t2, 1, out0: nếu $t2 = 1 thì nhảy đến out0 để $a0 nhận giá trị từ $t1 và print ra console. nếu $t2 = 0 thì thực hiện câu lệnh tiếp theo, để $a0 nhận giá trị từ $t0 sau đó nhảy đến out1 và print ra console.
- li $v0, 4 la $a0, newline syscall: để viết xuống dòng.

```
#Sum, dif, mul, div
#Sum
add $t3, $t0, $t1
li $v0, 4
la $a0, sum
syscall
li $v0, 1
move $a0, $t3
syscall
li $v0, 4
la $a0, newline
syscall

#Dif
sub $t4, $t0, $t1
li $v0, 4
la $a0, dif
syscall
li $v0, 1
move $a0, $t4
syscall
li $v0, 4
la $a0, newline
syscall
```

* Tính tổng, hiệu
- add $t3, $t0, $t1: cộng $t0 và $t1 sau đó lưu vào $t3.
- li $v0, 4 la $a0, sum syscall: dùng để print dòng chữ “Tong: ”.
- li $v0, 1: load $v0 = 1 để chuẩn bị print số nguyên.
- move $a0, $t3: chuyển giá trị $t3 cho $a0, là giá trị print ra màn hình.
- syscall: dữ liệu được print ra màn hình console.
- li $v0, 4 la $a0, newline syscall: dòng này để tạo một ‘\n’ như đã giải thích bên trên.
Từ đó ta đã có được kết quả Tổng trên màn hình console.

```
#Mul
mul $t5, $t0, $t1
li $v0, 4
la $a0, multi
syscall
li $v0, 1
move $a0, $t5
syscall
li $v0, 4
la $a0, newline
syscall

#Div
div $t6, $t0, $t1
li $v0, 4
la $a0, divi
syscall
li $v0, 1
move $a0, $t6
syscall
```

Tương tự như tính tổng, hiệu, tích thương ta cũng làm tương tự chỉ thay các phép tính phù hợp.
Từ đó ta có được kết quả của bài các phép tính trên màn hình như sau:

```
Nhap so nguyen thu nhat vao: 4
Nhap so nguyen thu hai vao: 2
So lon hon la: 4
Tong: 6
Hieu: 2
Tich: 8
Thuong 2
```