ĐẠI HỌC QUỐC GIA TP.HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

TỔ CHỨC VÀ CẤU TRÚC MÁY TÍNH II

LỚP: IT012.N21.2

BÁO CÁO THỰC HÀNH SỐ 4 (LAB 04)

Giảng viên hướng dẫn: Nguyễn Thành Nhân

Sinh viên: Hồ Trọng Hiển

MSSV: 22520414

MỤC LỤC

Phần 2: Thực hành

```
1
2  .text
3  li $s0, 7
4  li $s1, 6
5  li $t0, 8
6  li $t1, 7
7
8  bne $s0, $s1, else
9  add $s2, $t0, $t1
10 j exit
11
12 else:
13 sub $s2, $t0, $t1
14 exit:
15
16 li $v0, 1
17 la $a0, ($s2)
18 syscall
19
20
```

```
if (i == j)
    f = g + h;
else
    f = g – h;
```

```
1  .text
2  li $s1, 5
3  li $s2, 0
4  li $s0, 1
5  loop:
6  sle $t0, $s0, $
7  beq $t0, $0, ex
8  add $s2, $s2, $
9  addi $s0, $s0,
10 j loop
11 exit:
12
13 li $v0, 1
14 la $a0, ($s2)
15 syscall
16
```

```
int Sum = 0
for (int i = 1; i <=N; ++i){
    Sum = Sum + i;
}
```

Phần 3: Bài tập

Câu 1: Nhập vào một ký tự, xuất ra cửa sổ I/O của MARS theo từng yêu cầu:

- Ký tự liền trước và sau của ký tự nhập vào:

```
.data
        string1: .asciiz "Nhap ky tu (chi mot ky tu):
        string2: .asciiz "\nKy tu truoc: "
        string3: .asciiz "\nKy tu sau: "
        s1: .asciiz "\nKy tu la so\n"
        s2: .asciiz "\nKy tu la chu\n"
        s3: .asciiz "\ninvalid type\n"
        s4: .asciiz "\nKy tu la chu in hoa\n"
.text
        #print string1
        li $v0, 4
        la $a0, string1
        syscall

        #read char
        li $v0, 12
        syscall
        move $s0, $v0
        move $s1, $v0 #$s0=$s1=$v0
```

- Khai báo các chuỗi và tiến hành đọc ký tự vào.

- Lưu ký tự được đọc vào hai thanh ghi $s0 và $s1.

```
res:
li $v0, 4
la $a0, string2
syscall
addi $s0, $s0, -1 #Giam di 1 don vi trong bang ma ascii
li $v0, 11 #print_char
la $a0, ($s0)
syscall

li $v0, 4
la $a0, string3
syscall
addi $s1, $s1, 1 #Tang them 1 don vi trong bang ma ascii
li $v0, 11 #print_char
la $a0, ($s1)
syscall

exit:
```

- In ký tự trước bằng cách giảm mã ASCII đi 1 đơn vị, sau đó gán $v0 = 11 và load $s0 lên $a0 để in ký tự đó ra.

- Có ký tự đằng sau bằng cách tăng mã ASCII lên 1 đơn vị, in ký tự đó ra như trên.

- Thông báo kiểu dữ liệu được nhập:

```
sle $t0, $s0, 57
sge $t1, $s0, 48 #48 <= s0 <= 57: number
beq $t0, $t1, number #Thoa man thi nhay den number

sle $t0, $s0, 122
sge $t1, $s0, 97 #97 <= s0 <= 122: char in hoa
beq $t0, $t1, char #Thoa man thi nhay den char

sle $t0, $s0, 90
sge $t1, $s0, 65 #65 <= s0 <= 90: char thuong
beq $t0, $t1, char_hoa #Thoa man thi nhay den char in hoa



#neu khong roi vao cac truong hop tren thi in invalid type roi exit
li $v0, 4
la $a0, s3
syscall
j exit

number:
li $v0, 4
la $a0, s1 #In ra s1
syscall
j res

char:
li $v0, 4
la $a0, s2  #In ra s2
syscall
j res

char_hoa:
li $v0, 4
la $a0, s4  #In ra s4
syscall
```

So sánh mã ASCII của ký tự nhập vào.

- Dùng lệnh sle để so sánh bé hơn bằng, sge để so sánh lớn hơn bằng.

+ TH1: mã ASCII nằm trong [48, 57] thì nhảy đến number rồi in ra “Ky tu la so.”

+ TH2: mã ASCII nằm trong [97, 122] thì nhảy đến char rồi in ra “Ky tu la chu.”

+ TH3: mã ASCII nằm trong [65, 90] thì nhảy đến char_hoa rồi in ra “Ky tu la chu in hoa.”

+ Nếu không nằm trong các TH trên thì in ra “Invalid type” rồi kết thúc chương trình.

Câu 2: Nhập vào hai số nguyên, in ra cửa sổ I/O các yêu cầu:

- Số lớn hơn:

```
.data
        string1: .asciiz "\nHai so bang nhau"
        string2: .asciiz "\nSo lon hon: "
        string3: .asciiz "\nNhap so dau tien: "
        string4: .asciiz "\nNhap so thu hai: "
```

- Khai báo các chuỗi cần thiết.

```
#Nhap 2 so nguyen
li $v0, 4
la $a0, string3
syscall
li $v0, 5
syscall
move $s0, $v0 #Gan so dau tien cho s0

li $v0, 4
la $a0, string4
syscall
li $v0, 5
syscall
move $s1, $v0 #Gan so thu hai cho s1
```

- Nhập số thứ nhất:

+ In ra màn hình chuỗi: “Nhap so dau tien: ” bằng cách gán $v0 = 4, load string3 lên $a0 và dùng syscall.

+ Đọc số thứ nhất vào thanh ghi $v0 và gán vào $s0.

- Nhập số thứ hai tương tự, gán vào $s1.

```
#s1 = s0
beq $s1, $s0, equal

#s0 < s1
slt $t1, $s0, $s1
bne $t1, $0, less

#s1 < s0
li $v0, 4
la $a0, string2
syscall
li $v0, 1
la $a0, ($s0)
syscall
j res

equal:
li $v0, 4
la $a0, string1
syscall
j res

less:
li $v0, 4
la $a0, string2
syscall
li $v0, 1
la $a0, ($s1)
syscall
j res
```

- Kiểm tra hai giá trị trong hai thanh ghi $s0, $s1 có bằng nhau không? Nếu có, nhảy vào equal rồi in ra “Hai so bang nhau.” và nhảy đến phần tiếp.

- Nếu không bằng nhau, kiểm tra giá trị trong thanh ghi nào lớn hơn.

+ Đầu tiên so sánh $s0 < $s1 bằng lệnh sle, gán kết quả vào $t1.

+ So sánh kết quả đó với $0, nếu khác nhau chứng tỏ $s0 < $s1 thì nhảy vào less in “So lon hon: ” và số lớn hơn trong trường hợp này là $s1 được in ra.

+ Nếu trường hợp kia không xảy ra thì tiến hành in $s0 là số lớn hơn.

- Tính tổng, hiệu:

```
#Sum
li $v0, 4
la $a0, s1
syscall
add $s2, $s1, $s0
li $v0, 1
la $a0, ($s2)
syscall
```

*Tính tổng:

+ In ra dòng “Tong hai so: ”.

+ Dùng lệnh add cộng hai giá trị trong thanh ghi $s0 và $s1 lưu vào thanh ghi $s2.

+ Đọc giá trị trong thanh ghi $s2 lên thì đó là tổng cần tìm.

```
#Sub
li $v0, 4
la $a0, s2
syscall
sub $s2, $s0, $s1
li $v0, 1
la $a0, ($s2)
syscall
```

*Tính hiệu:

+ In ra dòng “Hieu hai so: ”.

+ Dùng lệnh sub trừ hai giá trị trong thanh ghi $s0 và $s1 lưu vào thanh ghi $s2.

+ Đọc giá trị trong thanh ghi $s2 lên thì đó là hiệu cần tìm.

```
#Mul
li $v0, 4
la $a0, s3
syscall
mult $s0, $s1 #s0*s1: Ket qua la so nguyen luu trong $lo
mflo $s2
li $v0, 1
la $a0, ($s2)
syscall

#Div
div $s0, $s1 #s0 / s1
mflo $s2 #Phan nguyen luu trong thanh ghi $lo
mfhi $s3 #Phan du luu trong thanh ghi $hi

li $v0, 4
la $a0, s4
syscall
li $v0, 1
la $a0, ($s2)
syscall

li $v0, 4
la $a0, s5
syscall
li $v0, 1
la $a0, ($s3)
syscall
```

- Tính tích, thương:

* Tính tích:

+ Thực hiện lệnh mult để tiến hành nhân hai giá trị trong thanh ghi $s0 và $s1, kết quả lưu vào thanh ghi $lo, nếu thanh ghi $lo bị tràn thì lưu qua thanh ghi $hi.

+ Lấy giá trị từ thanh ghi $lo qua lệnh mflo gán vào $s2 là kết quả.

+ In giá trị trong $s2 là kết quả cần tìm.

*Tính thương:

+ Lấy div $s0, $s1 là lấy $s0/$s1.

+ Phần nguyên lưu trong $lo.

+ Phần dư lưu trong $hi.

+ Dùng lệnh mflo và mfhi để lấy phần nguyên và phần dư gán vào thanh ghi $s2, $s3.

+ In giá trị $s2 là phần nguyên của phép chia, $s3 là phần dư.