ĐẠI HỌC BÁCH KHOA HÀ NỘI

TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

****

BÁO CÁO

Bài thi giữa kì 20212

Học phần: Thực hành kiến trúc máy tính

Giảng viên hướng dẫn: Lê Bá Vui

Sinh viên thực hiện: Phạm Thị Ánh - 20194715

Mã lớp: 130938

Hà Nội, tháng 5 năm 2022

Phần A bài 2:

Code:

```
# 2. Nhập 2 số nguyên dương M và N từ bàn phím. In ra màn hình bội số chung nhỏ nhất của M và N
.data
    mess: .ascii "BCNN la: "
.text
    li $v0, 5
    syscall
    add $s0, $v0, 0     # M

    li $v0, 5
    syscall
    add $s1, $v0, 0     #N

    li $t0, 0           #max
    li $t1, 0           #step
    li $t2, 0           #BCNN

if:
    ble $s0, $s1,else   #neu M <= N thi nhay
    add $t0, $s0, 0     # max = M
    add $t1, $s0, 0     # step = M
    j loop

else:
    add $t0, $s1, 0     # max = N
    add $t1, $s1, 0     # step = N

loop:
    div $t0, $s0
    mfhi $a0            #neu max chia M du 0 thi tiep tuc lap
    bne $a0, $zero, endloop

    div $t0, $s1
    mfhi $a1            #neu max chia N du 0 thi tiep tuc lap
    bne $a1, $zero, endloop

    add $t2, $t0, 0     # BCNN = max
    j print

endloop:
    add $t0, $t0, $t1   # max += step
    j loop

print:
    li  $v0, 4
    la  $a0, mess       #in chuoi
    syscall

    li  $v0, 1
    add $a0, $t2, 0     #in BCNN
    syscall
```

Kết quả chạy được :

```
12
5
BCNN la: 60
-- program is finished running (dropped off bottom) --
```

```
37
28
BCNN la: 1036
-- program is finished running (dropped off bottom) --
```

Cách thực hiện: Đầu tiên tìm số lớn nhất trong 2 số, gán giá trị lớn nhất cho biến tạm và kiểm tra xem số này có chia hết cho số còn lại không, nếu chia hết thì BCNN chính là số lớn nhất đó. Nếu không chia hết, tiếp tục cộng giá trị của số lớn hơn vào biến tạm và thực hiện vòng lặp loop. Cho đến khi tìm thấy BCNN và in ra màn hình.

Phần B bài 10:

Code:

```
#10. Nh?p m?ng s? nguyên t? bàn phím. In ra màn hình s? l? l?n nh?t nh? h?n m?i s? ch?n trong
m?ng

.data
    A:word  0:100
    mess1:.asciiz "So phan tu cua mang: "
    error:.asciiz "phan tu phai lon hon 0\n"
    mess2:.asciiz "phan tu thu "
    mess3:.asciiz " la:"
    mess4:.asciiz "Phan tu le lon nhat nho hon tat ca cac so chan trong mang la: "
    mess5:.asciiz " Mang khong co phan tu chan"
.text
    Nhap_so_phan_tu:
    li $v0,4
    la $a0,mess1        # print string
    syscall

    li $v0,5            # read integer
    syscall

    add $t5,$t5,$v0     # luu so phan tu cua mang vào $t5 (n phan tu)
    slt $t9,$t5,$zero   # kiem tra n < 0 thì lenh duoi se nhay den nhan loi ($t5<0 thi $t9=1)
    bne $t9,$zero,loi   # $t9 = 0, khong co loi
    j ketthuc

loi:
    li $v0,4
    la $a0,error        #in chuoi
    syscall
```

```
    j Nhap_so_phan_tu   #nhap lai

ketthuc:
    li $t1,0            # i = 0
nhap_mang:
    beq $t1,$t5,end_nhapmang # i = n thi ket thuc vong lap

    li $v0,4
    la $a0,mess2        #in chuoi
    syscall

    li $v0,1
    add $a0,$t1,$zero   # in i
    syscall

    li $v0,4
    la $a0,mess3        #in chuoi
    syscall

    li $v0,5            # nhan 1 so nguyen nhap tu ban phim
    syscall

    sll $t2,$t1,2       # dich trai i sang 2 bit ($t1*4 = $t2)
    sw  $v0,A($t2)      # luu gia tri so vua nhap vao A[i]

    addi $t1,$t1,1      # i++
    j nhap_mang

end_nhapmang:
    li $t9,0    #sum
    li $t1,0    # i=0
    la $a0,A    #load dia chi mang A ( A[0] )
    li $t8,3

khaibao:
    li $s0, 10000000        #max
    add $s2, $s0, 0     # min = max
    li  $t1, 0          # i=0
    li  $s3, 2
    j loop
loop_up:
    addi $t1,$t1,1

loop:
    beq $t1,$t5,if  # i=n => endloop
    sll $t2,$t1,2           # $t2 = i*4
    add $t3,$t2,$a0         # lay dia chi A[i] luu vao $t3
    lw  $t2,0($t3)          # lay gia tri A[i] luu vao $t2
    blt $s2, $t2, loop_up   # nhay neu min < A[i]
    div $t2, $s3            #A[i] / 2
    mfhi $v1                #lay du
    bne $v1, $zero, loop_up     # du khac khong thi nhay
    add $s2, $t2, 0         # min = A[i]
    j loop_up
if:
    beq $s2, $s0,else
```

```
    add $s4, $s2, -1        # return min - 1
    j main
else:
    li $v0,4
    la $a0,mess5        # print string
    syscall
    j end
main:
    li $v0,4
    la $a0,mess4        # print string
    syscall

    li $v0,1
    add $a0,$s4,$zero   # in so
    syscall

end:
```

Kết quả chạy được :

```
So phan tu cua mang: 5
phan tu thu 0 la:13
phan tu thu 1 la:3
phan tu thu 2 la:4
phan tu thu 3 la:6
phan tu thu 4 la:2
Phan tu le lon nhat nho hon tat ca cac so chan trong mang la: 1
-- program is finished running (dropped off bottom) --
```

```
So phan tu cua mang: 3
phan tu thu 0 la:21
phan tu thu 1 la:5
phan tu thu 2 la:7
 Mang khong co phan tu chan
-- program is finished running (dropped off bottom) --
```

Cách thực hiện: Đầu tiên, khởi tạo biến min có giá trị thật lớn để có thể tìm được giá trị lẻ nhỏ nhất trong mảng (min khác 0). Sau đó, tiến hành duyệt tất cả các phần tử trong mảng nếu A[i] vừa mang giá trị chẵn vừa bé hơn min thì ta gán min = A[i].

- Sau khi duyệt, nếu min bằng giá trị khởi tạo có nghĩa là mảng không có phần tử chẵn. In ra màn hình mảng không có phần tử chẵn và kết thúc chương trình
- Nếu khác giá trị khởi tạo ta in ra giá trị (min – 1) là phần tử lẻ lớn nhất nhỏ hơn tất cả các số chẵn trong mảng

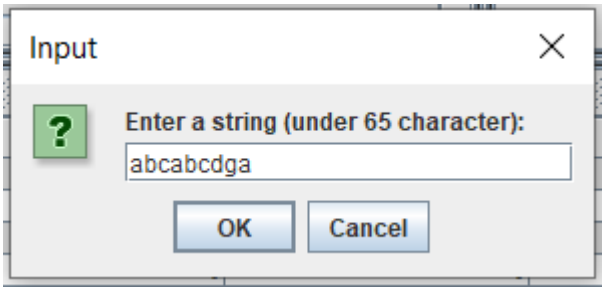
Phần C bài 2:

Code:

```
# 2. Nhập vào xâu ký tự. In ra màn hình số ký tự khác nhau có trong xâu.
.data
    tmp: .space 65              # chuoi tmp dung de luu tru cac ky tu khac nhau trong xau nguoi
dung nhap vao
    inputString: .space 65      # xau nguoi dung nhap vao duoc luu trong input string
    INPUT_STRING_MESSAGE: .asciiz "Enter a string (under 65 character): "
    RESULT:   .asciiz "SO KY TU KHAC NHAU TRONG CHUOI: "
.text
main:
#-----------------------------------------------------------------------------------------
# @brief    Nhan vao 1 xau tu nguoi dung
#-----------------------------------------------------------------------------------------
getString:
    li $v0, 54                      # hien thi thong bao nhap xau tu nguoi dung
    la $a0, INPUT_STRING_MESSAGE
    la $a1, inputString             # inputString luu tru xau nhap vao
    la $a2, 51
    syscall

    li $t1, 0                       # do dai hien tai cua chuoi tmp
    li $s2, 0                       # dem so ky tu khac nhau
    la $s0, inputString
    la $s1, tmp

    li $a0, '\0'
    li $a1, '\n'
loop1:
    la $t5, tmp                 # $t5 la con tro index tro den dia chi cua cac ki tu trong chuoi
tmp
    la $t6, tmp($t1)            # $t6 la dia chi sau ki tu cuoi cung cua chuoi tmp
    lb $t2, 0($s0)              # lay ki tu hien tai trong chuoi inputString de xem xet
    beq $t2, $a1, end_loop1
loop2:
    lb $t3, 0($t5)
    beq $t2, $t3, continue_loop1# xem ki tu dang xet cua chuoi inputString da ton tai trong chuoi
tmp hay chua?
    add $t5, $t5, 1             # tang con tro index $t5 len 1
    bgt $t5, $t6, end_loop2     # con tro index da tro den ki tu cuoi cung cua chuoi tmp hay
chua?
                                # neu chua thi cho vao cuoi cung cua chuoi tmp
    j loop2
end_loop2:
    sb $t2, 0($t6)              # luu ki tu vao cuoi cua chuoi tmp
    add $t1, $t1, 1             # tang do dai cua chuoi tmp len 1
    j loop1
continue_loop1:
    add $s0, $s0, 1
    j loop1
end_loop1:
    add $t6, $s1, $t1
    sb $a0, 0($t6)              # luu ki tu '\0' vao cuoi chuoi tmp


dem_ki_tu_khac_nhau:
```

```
    la $t0, tmp                 # address(tmp[0])
    la $t2, tmp($t1)            # address(tmp[length])
loop3:
    beq $t0, $t2, print         # neu $t0 == address(tmp[length]) (tuc la $t0 dan tro den ki tu
'\0' trong chuoi tmp
                                # thi se in ra so cac ky tu khac nhau)
    add $s2, $s2, 1             # count++
    add $t0, $t0, 1             # tang con tro index len 1
    j loop3
print:
    li  $v0, 4                  # in ra
    la $a0, RESULT
    syscall

    li  $v0, 1
    add $a0, $s2, 0
    syscall
```

Kết quả chạy được :




```
SO KY TU KHAC NHAU TRONG CHUOI: 5
-- program is finished running (dropped off bottom) --
```


```
SO KY TU KHAC NHAU TRONG CHUOI: 13
-- program is finished running (dropped off bottom) --
```

Cách thực hiện: Duyệt qua kí tự của chuỗi nhập vào xem tồn tại trong chuỗi lưu các kí tự khác nhau hay chưa.