BÁO CÁO BÀI THỰC HÀNH GIỮA KÌ 20232

Họ và tên: Lê Quang Khải

MSSV: 20225638

Bài 9A:

```
.data
messenger: .asciiz "Nhap n duong (2 chu so tro len): "
tongchan: .asciiz "Tong so chan trong n la: "
tongle: .asciiz "Tong so le trong n la: "
nhapsai: .asciiz "So n phai duong va co 2 chu so tro len!"
newline: .asciiz "\n"
.text
main:
    li $v0, 4
    la $a0, messenger # in ra màn hình nhập n
    syscall
    li $v0, 5
    syscall
    move $s1, $v0 # $s1 = n
    li $s2, 10    # $s2 = 10
    blt $s1, $s2, end_main_sai
    li $s3, 2     # $s3 = 2
    li $s4, 0 # sumchan=0
    li $s5, 0 # sumle=0
    j loop
loop:
    beq $s1, $0, endloop # khi nào n / 10 = 0 thì đã duyệt hết thì dừng
    div $s1, $s2 # temp chia 10
    mflo $s1 # lấy phần thương (temp = temp/10)
    mfhi $t1 # lấy dư của phép chia (chữ số cuối cùng của n)
    div $t1, $s3 # chữ số cuối chia 2
    mfhi $t2 # lưu số dư của phép chia 2 vào $t2
    beq $t2, $0, tong_chan # dư = 0 thì là chẵn thì ta nhảy tới tính tổng
    add $s5, $s5, $t1 # sumle = sumle + le
    j loop
tong_chan:
    add $s4, $s4, $t1 # sumchan = sumchan + chan
    j loop
endloop:
    li $v0, 4
```

```
    la $a0, tongchan # in ra tổng chữ số là chẵn của n
    syscall
    li $v0, 1
    move $a0, $s4
    syscall
    li $v0, 4
    la $a0, newline # xuống dòng
    syscall
    li $v0, 4
    la $a0, tongle # in ra tổng chữ số là lẻ của n
    syscall
    li $v0, 1
    move $a0, $s5
    syscall
    li $v0, 10
    syscall
end_main_sai:
    li $v0, 4
    la $a0, nhapsai # in ra thông báo nhập sai và kết thúc ctrinh
    syscall
    li $v0, 10
    syscall
```

Phân tích cách thực hiện:

1. **Nhập số nguyên dương n**:
- Sử dụng syscall để in ra thông báo "Nhập n duong (2 chu so tro len): ".
- Sử dụng syscall để nhập giá trị số nguyên dương n từ người dùng.
2. **Kiểm tra n có hợp lệ hay không**:
- Sử dụng một số lệnh so sánh và nhảy để kiểm tra xem n có ít nhất hai chữ số không.
- Nếu n không hợp lệ, chương trình sẽ in ra thông báo "So n phai duong va co 2 chu so tro len!" và kết thúc.
3. **Tính tổng chữ số chẵn và tổng chữ số lẻ**:
- Sử dụng một vòng lặp để duyệt qua từng chữ số của n.
- Trong mỗi vòng lặp:
    - Sử dụng phép chia cho 10 để lấy ra chữ số cuối cùng của n.
    - Thực hiện phép chia cho 2 để kiểm tra xem chữ số đó có phải là chẵn hay lẻ.
    - Cập nhật tổng chữ số chẵn hoặc tổng chữ số lẻ tương ứng.

4. **In ra tổng chữ số chẵn và tổng chữ số lẻ**:
- Sử dụng syscall để in ra tổng chữ số chẵn và tổng chữ số lẻ.

Ý nghĩa của các chương trình con:

- **tong_chan**: Được gọi khi một chữ số của n là số chẵn. Nhiệm vụ của nó là cộng chữ số này vào biến tổng chữ số chẵn ($s4).
- **tong_le**: Được gọi khi một chữ số của n là số lẻ. Nhiệm vụ của nó là cộng chữ số này vào biến tổng chữ số lẻ ($s5).

Chương trình sử dụng một cấu trúc lặp và các phép chia lấy phần dư để trích xuất từng chữ số của n. Sau đó, từng chữ số được kiểm tra và cộng vào tổng tương ứng. Cuối cùng, tổng của chữ số chẵn và tổng của chữ số lẻ được in ra màn hình.

Kết quả thực hiện với các trường hợp:

- Trường hợp 1: n âm

```
Nhap n duong (2 chu so tro len): -5
So n phai duong va co 2 chu so tro len!
-- program is finished running --
```

- Trường hợp 2: n là số có 1 chữ số

```
Nhap n duong (2 chu so tro len): 9
So n phai duong va co 2 chu so tro len!
-- program is finished running --
```

- Trường hợp 3 (trường hợp thỏa mãn): n là số có 2 chữ số trở lên

```
Nhap n duong (2 chu so tro len): 39087
Tong so chan trong n la: 8
Tong so le trong n la: 19
-- program is finished running --
```

Bài 2B:

```
.data
A: .word
newline: .asciiz "\n"
messenger1: .asciiz "Nhap kich thuoc cua mang: "
messenger2: .asciiz "Enter a number: "
ketqua: .asciiz "Cap phan tu lien ke co tich lon nhat la: "
comma: .asciiz ", "
.text
```

```
main:
   la $a0, messenger1
   li $v0, 4
   syscall
   li $v0, 5
   syscall
   add $a1, $zero, $v0     # length = $a1
   addi $t0, $zero, 0      # gán i = 0
read:
   beq $t0, $a1, end_read  # end loop after reaching the length of the array
   la $a0, messenger2
   li $v0, 4
   syscall
   li $v0, 5
   syscall
   la $a0, A
   add $t2, $t0, $t0    # put 2i in $t2
   add $t2, $t2, $t2    # put 4i in $t2
   add $t3, $t2, $a0    # put 4i+A (address of A[i]) in $t3
   sw $v0, 0($t3)    # store v0 in A[i]
   addi $t0, $t0, 1
   j read
end_read:
   la $s0, A      # gán $s0 = &A[0]
   move $s1, $t3  # với $t3 là address of A[n-1] đã làm ở phần read
   addi $s1, $s1, -4    # $a1 = Address(A[n-1])
   lw $t2, 0($s0)    # t2 = A[0]
   lw $t3, 4($s0)    # t3 = A[1]
   mult $t2, $t3
   mflo $s3          # s3 = A[0]*A[1]
   move $t0, $s3     # $t0 = max
   move $s4, $t2     # cập nhật cặp số mà có tích là max
   move $s5, $t3
   j find
end_main:
   li $v0, 10
   syscall
find:
   beq $s0, $s1, print  #single element list is sorted
   j find_max
find_max:
   addi $s0, $s0, 4  # trỏ tiếp lên A[i+1]
   lw $t2, 0($s0)    # s0 = A[i]
   lw $t3, 4($s0)    # s1 = A[i+1]
   mult $t2, $t3
   mflo $s3          # s3 = A[i]*A[i+1]
   blt $s3, $t0, find
   move $t0, $s3     # $t0 = max_new
   move $s4, $t2     # cập nhật cặp số mà có tích là max
   move $s5, $t3
```

```
    j find
print:
    li $v0, 4
    la $a0, ketqua    # in ra màn hình kết quả
    syscall
    li $v0, 1
    move $a0, $s4
    syscall
    li $v0, 4
    la $a0, comma
    syscall
    li $v0, 1
    move $a0, $s5
    syscall
```

Phân tích cách thực hiện:

- **Nhập kích thước của mảng:**
  - Sử dụng syscall để in ra thông báo "Nhap kich thuoc cua mang: ".
  - Sử dụng syscall để nhập giá trị kích thước của mảng từ người dùng.
- **Nhập các phần tử của mảng:**
  - Sử dụng vòng lặp để nhập các phần tử của mảng từ người dùng.
  - Sử dụng syscall để in ra thông báo "Enter a number: ".
  - Sử dụng syscall để nhập giá trị phần tử của mảng từ người dùng.
  - Lưu giá trị phần tử vào mảng A.
- **Tìm cặp phần tử có tích lớn nhất:**
  - Sử dụng một vòng lặp để duyệt qua từng cặp phần tử liên tiếp trong mảng.
  - Tại mỗi bước duyệt:
    - Tính tích của hai phần tử liên tiếp.
    - So sánh tích mới với giá trị lớn nhất đã tìm thấy trước đó.
    - Nếu tích mới lớn hơn, cập nhật giá trị lớn nhất và lưu cặp phần tử đó.
- **In ra cặp phần tử có tích lớn nhất:**
  - Sử dụng syscall để in ra thông báo "Cap phan tu lien ke co tich lon nhat la: ".
  - In ra cặp phần tử có tích lớn nhất.

Ý nghĩa của các chương trình con:

- **find_max:** Chương trình này thực hiện so sánh tích mới tính được với giá trị lớn nhất đã tìm thấy ở trước đó. Nếu tích mới lớn hơn, chương trình sẽ cập nhật giá trị lớn nhất và lưu cặp phần tử tương ứng.
- **print:** Chương trình này in ra cặp phần tử có tích lớn nhất đã tìm được.

Kết quả thực hiện với các trường hợp:

- Với mảng [1, 2, 3, 4, 5, 6, 7, 8] với max ở giữa hoặc cuối

```
Nhap kich thuoc cua mang: 8
Enter a number: 1
Enter a number: 2
Enter a number: 3
Enter a number: 4
Enter a number: 5
Enter a number: 6
Enter a number: 7
Enter a number: 8
Cap phan tu lien ke co tich lon nhat la: 7, 8
-- program is finished running (dropped off bottom) --
```

- Với mảng [7, 3, 6, 3, -2, -5] với max ở đầu phần tử

```
Nhap kich thuoc cua mang: 6
Enter a number: 7
Enter a number: 3
Enter a number: -2
Enter a number: -5
Enter a number: 3
Enter a number: 6
Cap phan tu lien ke co tich lon nhat la: 7, 3
-- program is finished running (dropped off bottom) --
```

Bài 6C:

```
.data
string: .space 100
newline: .asciiz "\n"
messenger1: .asciiz "Nhap chuoi ki tu: "
messenger2: .asciiz "Nhap ky tu C: "
messenger3: .asciiz "\nSo lan xuat hien cua ky tu C trong chuoi la: "
.text
```

```
main:
   li $v0, 4
   la $a0, messenger1         # in ra màn hình nhập chuỗi kí tự
   syscall
   li $v0, 8
   la $a0, string
   li $a1, 100
   syscall
   li $v0, 4
   la $a0, messenger2         # in ra màn hình nhập ký tự C
   syscall
   li $v0, 12
   syscall
   move $s1, $v0           # $s1 = kí tự C
   beq $s1, 10, end_main_enter         # nếu kí tự C là kí tự 'enter'
   la $s0, string
   bge $s1, 'a', toupcase_C      # Nếu kí tự C là chữ thường thì chuyển thành hoa
   j count                    # Nếu không phải chữ thường thì giữ nguyên
toupcase_C:
   bgt $s1, 'z', count        # Nếu kí tự C là chữ thường thì chuyển thành hoa
   addi $s1, $s1, -32
   j count
count:
   li $t0, 0            # Khởi tạo biến đếm $t0 = 0
count_loop:                   # Chuyển hết xâu kí tự thành chữ hoa còn các kí tự
khác thì giữ nguyên
   lb $t1, 0($s0)          # Lấy ký tự đầu tiên của chuỗi vào $t1
   beq $t1, 10, end_count        # Nếu đã đến kí tự enter kết thúc vòng lặp
   j to_upper
to_upper:
   bge $t1, 'a', toupcase
   j check
toupcase:
   bgt $t1, 'z', check        # lớn hơn 'z' thì giữ nguyên
   addi $t1, $t1, -32
check:
   beq $t1, $s1, increase_count  # Nếu hai ký tự giống nhau, tăng biến đếm lên 1
   addi $s0, $s0, 1
   j count_loop
increase_count:
   addi $t0, $t0, 1           # Tăng biến đếm lên 1
   addi $s0, $s0, 1           # Di chuyển đến ký tự tiếp theo trong chuỗi
   j count_loop
end_count:
   li $v0, 4
```

```
   la $a0, messenger3
   syscall
   li $v0, 1
   move $a0, $t0
   syscall
   li $v0, 10           # kết thúc chương trình
   syscall
end_main_enter:
   li $t0, 1            # chắc chắn có 1 lần enter
   li $v0, 4
   la $a0, messenger3
   syscall
   li $v0, 1
   move $a0, $t0
   syscall
   li $v0, 10           # kết thúc chương trình
   syscall
```

Phân tích cách thực hiện:

1. **Nhập chuỗi từ người dùng:**

- In ra màn hình thông báo "Nhap chuoi ki tu: ".
- Sử dụng syscall để nhận chuỗi từ người dùng và lưu vào vùng nhớ được chỉ định bởi string.

2. **Nhập ký tự C từ người dùng:**

- In ra màn hình thông báo "Nhap ky tu C: ".
- Sử dụng syscall để nhận một ký tự từ người dùng và lưu vào thanh ghi $s1.

3. **Nếu kí tự C là "Enter"** :

- Nếu là kí tự Enter thì chắc chắn kí tự đó xuất hiện 1 lần
- Nên ta nhảy tới hàm end_main_enter và sử dụng syscall để hiện ra giá trị của 1 lần xuất hiện kí tự "Enter" đó luôn

4. **Chuyển đổi ký tự C thành in hoa nếu cần:**

- Kiểm tra xem ký tự C có phải là ký tự thường không.
- Nếu là ký tự thường, chương trình chuyển đổi thành ký tự in hoa bằng cách giảm giá trị ASCII của nó đi 32.

5. **Đếm số lần xuất hiện của ký tự C trong chuỗi :**

- Sử dụng một vòng lặp để duyệt qua từng ký tự trong chuỗi và biến hết chữ cái thường của chuỗi thành chữ hoa để so sánh với C .
- Mỗi lần lặp, chương trình kiểm tra xem ký tự đó có bằng ký tự C không.
- Nếu bằng, biến đếm tăng lên một đơn vị.

6. **In ra số lần xuất hiện của ký tự C trong chuỗi:**

- In ra màn hình số lần xuất hiện của ký tự C trong chuỗi.

Ý nghĩa của các chương trình con (nếu có):

- **toupcase_C:** Chương trình này kiểm tra xem ký tự C có phải là ký tự thường không. Nếu là ký tự thường, nó sẽ chuyển đổi thành ký tự in hoa bằng cách giảm giá trị ASCII của nó đi 32.
- **toupcase:** Hàm này là một phần của việc chuyển đổi ký tự trong xâu kí tự sang in hoa. Nếu ký tự đang xét là ký tự in thường, nó sẽ được chuyển đổi thành ký tự in hoa bằng cách giảm giá trị ASCII của nó đi 32.
  Nếu ký tự không phải là ký tự in thường (ví dụ: nằm ngoài khoảng giữa 'a' và 'z'), thì nó giữ nguyên ký tự đó.
- **count_loop:** Chương trình này duyệt qua từng ký tự trong chuỗi và kiểm tra xem ký tự đó có bằng ký tự C không. Nếu bằng, biến đếm được tăng lên một đơn vị.

Kết quả thực hiện với các trường hợp:

- **Trường hợp nhập xâu là để kiểm tra xem chương trình có "không phân biệt hoa thường" hay không?**
  - Xâu nhập vào: LeEQuangeKhai
    Kí tự C: e

```
Nhap chuoi ki tu: LeEQuangeKhai
Nhap ky tu C: e
So lan xuat hien cua ky tu C trong chuoi la: 3
-- program is finished running (dropped off bottom) --
```

  - Xâu nhập vào: LeEQuangeKhai
    Kí tự C: E

```
Nhap chuoi ki tu: LeEQuangeKhai
Nhap ky tu C: E
So lan xuat hien cua ky tu C trong chuoi la: 3
-- program is finished running (dropped off bottom) --
```

- **Trường hợp nhập xâu để kiểm tra các kí tự đặc biệt ngoài đoạn [a,z], [A,Z]**
    - Xâu nhập vào: Le-Quang-Khai-20225638
    - Kí tự C: -

```
Nhap chuoi ki tu: Le-Quang-Khai-20225638
Nhap chuoi ki tu: Le-Quang-Khai-20225638
Nhap ky tu C: -
So lan xuat hien cua ky tu C trong chuoi la: 3
-- program is finished running (dropped off bottom) --
```

    - Kí tự C: 6

```
Nhap chuoi ki tu: le-Quang-Khai-20225638
Nhap ky tu C: 6
So lan xuat hien cua ky tu C trong chuoi la: 1
-- program is finished running (dropped off bottom) --
```

- **Trường hợp kí tự C là kí tự "Enter":**

```
Nhap chuoi ki tu: Le-Quang-Khai-20225638
Nhap ky tu C:

So lan xuat hien cua ky tu C trong chuoi la: 1
-- program is finished running --
```