BÁO CÁO THỰC HÀNH KIẾN TRÚC MÁY TÍNH (IT3280) GIỮA KỲ

Họ và tên: Phạm Đức Long

MSSV: 20225737

Assignment A: Nhập số nguyên dương N từ bàn phím, in ra màn hình các số chia hết cho 3 hoặc cho 5 nhỏ hơn N.

- Code:

```
.data
        Input:          .asciiz "Nhap so N: "
        Message:        .asciiz "Cac so chia het cho 3 hoac 5 nho hon N la: "
        Message2:       .asciiz ", "
        Error:          .asciiz "N phai lon hon 0, nhap lai N: "

.text
input:
        # Nhap N:
        li      $v0, 4
        la      $a0, Input
        syscall

        li      $v0, 5
        syscall

        move    $s0, $v0        # gan $s0 = N
        blt     $s0, 0, error   # N < 0 thi error
```

```
message:
        li      $v0, 4
        la      $a0, Message
        syscall

init:
        li      $t0, 0          # $t0 = i = 0
        li      $t9, 0          # $t9 = j = 0, dem xem co bao nhieu output, de xoa dau phay o
cuoi
        li      $t1, 3          # $t1 = 3
        li      $t2, 5          # $t2 = 5

count_output:
        beq     $t0, $s0, restart       # neu i = N thi dung chuong trinh
        div     $t0, $t1                # i / 3
        mfhi    $t3                     # $t3 la so du cua phep chia tren
        beq     $t3, 0, count           # $t3 = 0 nghia la 3|i => count++
        div     $t0, $t2                # i / 5
        mfhi    $t4                     # $t4 la so du cua phep chia tren
        beq     $t4, 0, count           # $t4 = 0 nghia la 5|i => count++
        addi    $t0, $t0, 1             # i++
        j       count_output

count:
        addi    $t9, $t9, 1     # j++
        addi    $t0, $t0, 1     # i++
        j       count_output
```

```
restart:
        li      $t0, 0          # gan lai i = 0
        sub     $t9, $t9, 1     # j -= 1 vi so dau phay luon it hon output 1 cai
        j       check

dauphay:
        li      $v0, 4
        la      $a0, Message2
        syscall
        sub     $t9, $t9, 1     # j--

check:
        beq     $t0, $s0, exit          # neu i = N thi dung chuong trinh
        div     $t0, $t1                # i / 3
        mfhi    $t3                     # $t3 la so du cua phep chia tren
        beq     $t3, 0, print           # $t3 = 0 nghia la 3|i => in ra ket qua
        div     $t0, $t2                # i / 5
        mfhi    $t4                     # $t4 la so du cua phep chia tren
        beq     $t4, 0, print           # $t4 = 0 nghia la 5|i => in ra ket qua
        addi    $t0, $t0, 1             # i++
        j       check

print:
        li      $v0, 1
        move    $a0, $t0
        syscall
```

```
update:
        addi    $t0, $t0, 1     # i++
        beq     $t9, 0, check
        j       dauphay

error:
        li      $v0, 4
        la      $a0, Error
        syscall
        j       input

exit:
        li      $v0, 10
        syscall
```

- Phân tích cách thực hiện:
    - Đầu tiên chương trình thực hiện yêu cầu nhập số N từ bàn phím, sau đó ta set hai biến i = 0 với mục đích kiểm soát vòng lặp, va j = 0 với mục đích kiểm soát dấu phẩy (tức là loại bỏ dấu phẩy ở cuối, đơn giản hơn là dấu phẩy sẽ luôn nhỏ hơn các giá trị output 1 đơn vị).
    - Sau đó ta chạy i từ 0 đến N – 1, thực hiện kiểm tra xem i có chia hết cho 3 hoặc 5 hay không, nếu có thì ngay lập tức in ra giá trị đó, đồng thời tăng i lên 1 đơn vị và tiếp tục thực hiện kiểm tra, nếu không thì ta tăng i lên 1 đơn vị và quay lại kiểm tra.

- Kết quả thu được:

```
Nhap so N: 23
Cac so chia het cho 3 hoac 5 nho hon N la: 0, 3, 5, 6, 9, 10, 12, 15, 18, 20, 21
-- program is finished running --

Nhap so N: 46
Cac so chia het cho 3 hoac 5 nho hon N la: 0, 3, 5, 6, 9, 10, 12, 15, 18, 20, 21, 24, 25, 27, 30, 33, 35, 36, 39,
-- program is finished running --

Nhap so N: 19
Cac so chia het cho 3 hoac 5 nho hon N la: 0, 3, 5, 6, 9, 10, 12, 15, 18
-- program is finished running --
```

Assignment B: Nhập mảng số nguyên từ bàn phím. In ra tổng các phần tử lẻ và tổng các phần tử âm trong mảng.

- Code:

```
.data
        A:          .space 100
        Message:    .asciiz "Nhap so phan tu mang: "
        Message1:   .asciiz "Nhap so: "
        Message2:   .asciiz "Tong cac phan tu le la: "
        Message3:   .asciiz "Tong cac phan tu am la: "
        NewLine:    .asciiz "\n"
        Error:      .asciiz "So phan tu mang phai lon hon bang 1!\n"

.text
main:
        li      $v0, 4
        la      $a0, Message
        syscall

        # Nhap N:
        li      $v0, 5
```

```
        syscall

        move    $s0, $v0            # $s0 = N
        blt     $s0, 1, PrintError  # N < 1 => error
        la      $a1, A              # $a1 = address A[0]

input_array:
        # Nhap cac phan tu cua array:
        beq     $t2, $s0, CheckSum
        add     $t0, $a1, $t1
        li      $v0, 4
        la      $a0, Message1
        syscall

        li      $v0, 5
        syscall

        move    $s1, $v0
        sw      $s1, 0($t0)
        addi    $t2, $t2, 1
        mul     $t1, $t2, 4
        j       input_array

CheckSum:
        li      $s1, 0          # $s1 = SumOdd = tong phan tu le
        li      $s2, 0          # $s2 = SumNega = tong phan tu am
        li      $s3, 2
```

```
        li      $t0, 0          # $t0 = i = 0

loop:
        beq     $t0, $s0, PrintSum      # $t0 = N => Print branch
        lw      $t1, 0($a1)             # $t1 = A[i]

        # kiem tra phan tu le:
        div     $t1, $s3                # A[i] / 2
        mfhi    $t9                     # lay so du phep chia tren
        jal     CheckOdd

        # kiem tra phan tu am:
        jal     CheckNega

continue:
        addi    $t0, $t0, 1             # i++
        addi    $a1, $a1, 4
        j       loop

CheckOdd:
        beq     $t9, 1, SumOdd              # neu $t9 = 1 nghia la A[i] le => tinh tong
        beq     $t9, -1, SumOdd             # neu $t9 = -1 nghia la A[i] vua am vua le => tinh
tong
        jr      $ra                     # neu khong thi kiem tra tiep co am khong?

SumOdd:
        add     $s1, $s1, $t1           # update SumPosi
```

```
        jr      $ra                     # neu co thi kiem tra tiep co am khong?

CheckNega:
        blt     $t1, 0, SumNega             # neu A[i] < 0 => tinh tong
        jr      $ra                     # neu khong thi tiep vong lap moi

SumNega:
        add     $s2, $s2, $t1           # update SumNega
        j       continue

PrintSum:
        # in ket qua:
        li      $v0, 4
        la      $a0, Message2
        syscall

        li      $v0, 1
        addi    $a0, $s1, 0
        syscall

        li      $v0, 4
        la      $a0, NewLine
        syscall

        la      $a0, Message3
        syscall
```

```
        li      $v0, 1
        addi    $a0, $s2, 0
        syscall
        j       exit

PrintError:
        # in ra loi:
        li      $v0, 4
        la      $a0, Error
        syscall
        j       main

exit:
        # ket thuc chuong trinh
        li      $v0, 10
        syscall
```

- Giải thích chương trình:
  - Đầu tiên ta nhập số phần tử của mảng N, và các phần tử của mảng. Khởi tạo thanh ghi $s1 là tổng phần tử lẻ, $s2 là tổng phần tử âm, $s3 = 2 để kiểm tra chẵn lẻ và $t0 = i = 0.
  - Chú ý điều kiện rằng: một số lẻ có thể là số âm, nhưng số âm chưa chắc là số lẻ nên ta phải kiểm tra cả hai điều kiện của một phần tử. Do đó, CheckOdd sẽ kiểm tra phần tử đó lẻ không, có thì cộng vào SumOdd và quay lui về CheckNega xem phần tử đó có âm không, nếu có thì cộng phần tử đó vào SumNega, rồi tăng i lên 1 đơn vị, chuyển con trỏ sang phần tử tiếp theo để kiểm tra phần tử đó; lặp lại cho đến phần tử cuối cùng của mảng. Cuối cùng ta sẽ in ra tổng các phần tử lẻ trong mảng, và tổng các phần tử âm trong mảng.

- Kết quả thu được:

```
Nhap so phan tu mang: 5
Nhap so: -3
Nhap so: -4
Nhap so: 7
Nhap so: 8
Nhap so: 13
Tong cac phan tu le la: 17
Tong cac phan tu am la: -7
-- program is finished running --

Nhap so phan tu mang: 3
Nhap so: -2
Nhap so: -3
Nhap so: 7
Tong cac phan tu le la: 4
Tong cac phan tu am la: -5
-- program is finished running --

Nhap so phan tu mang: 7
Nhap so: 6
Nhap so: -2
Nhap so: 5
Nhap so: -7
Nhap so: -3
Nhap so: 9
Nhap so: 1
Tong cac phan tu le la: 5
Tong cac phan tu am la: -12
-- program is finished running --
```

Assignment C: Nhập vào xâu ký tự và ký tự C. In ra số lần xuất hiện ký tự C trong xâu (không phân biệt chữ hoa hay chữ thường).

- Code:

```
.data
        String:         .space 100
        Message1:               .asciiz "Nhap xau: "
        Message2:               .asciiz "Nhap ky tu: "
        Message3:               .asciiz "\nSo lan ky tu nay xuat hien trong xau la: "
```

```
.text
input:
        # nhap xau:
        li      $v0, 4
        la      $a0, Message1
        syscall

        li      $v0, 8
        li      $a1, 100
        syscall

        li      $v0, 4
        la      $a0, Message2
        syscall

        # nhap ky tu:
        li      $v0, 12
        syscall

        move    $s1, $v0        # luu ky tu vao $s1

set:
        li      $s2, 0          # $s2 = bien dem = count
        li      $t2, 10         # \n
        la      $t0, String     # dia chi cua String[0]
```

```
check:
        # kiem tra neu ky tu in thuong
        slti    $s3, $s1, 123
        sgtu    $s4, $s1, 96
        add     $s5, $s3, $s4
        beq     $s5, 2, compare1
        # kiem tra neu ky tu in hoa
        slti    $s3, $s1, 91
        sgtu    $s4, $s1, 64
        add     $s5, $s3, $s4
        beq     $s5, 2, compare2

compare1:
        # lay tung ky tu cua xau de so sanh neu C in thuong
        lb      $s0, 0($t0)             # lay ky tu String[i]
        beq     $s0, $t2, print  # duyet het xau thi in ket qua
        add     $t0, $t0, 1             # tang dia chi len 1
        beq     $s0, $s1, update1       # neu String[i] = C thi update count
        sub     $s3, $s1, 32            # $s3 = $s1 - 32 = ky tu C o dang in hoa
        beq     $s0, $s3, update1       # neu String[i] = C thi update count
        j       compare1

compare2:
        # lay tung ky tu cua xau de so sanh neu C in hoa
        lb      $s0, 0($t0)             # lay ky tu String[i]
        beq     $s0, $t2, print         # duyet het xau thi in ket qua
        add     $t0, $t0, 1             # tang dia chi len 1
```

```
        beq     $s0, $s1, update2       # neu String[i] = C thi update count
        addi    $s3, $s1, 32            # $s3 = $s1 + 32 = ky tu C o dang in thuong
        beq     $s0, $s3, update2       # neu String[i] = C thi update count
        j       compare2

update1:
        add     $s2, $s2, 1     # count++
        j       compare1

update2:
        add     $s2, $s2, 1     # count++
        j       compare2

print:
        # in ket qua:
        li      $v0, 4
        la      $a0, Message3
        syscall

        li      $v0, 1
        addi    $a0, $s2, 0
        syscall

end:
        li      $v0, 10
        syscall
```

- Giải thích chương trình:

- Đầu tiên, ta sẽ nhập vào xâu ký tự, và ký tự C. Đặt $s2 = biến đếm = count
- Sau đó hàm check dùng để kiểm tra xem ký tự C là in thường hay in hoa. Nếu là in thường thì ta sẽ xuống hàm compare1, kiểm tra ký tự hiện tại trong xâu có trùng với C không, sau đó lấy C – 32 để ra C để C ở dạng in hoa, rồi lại kiểm tra xem các ký tự có trong xâu có trùng với C hay không, nếu có thì ta sẽ tăn biến đếm thêm một đơn vị rồi tăng địa chỉ lên một, kiểm tra ký tự tiếp theo. Cuối cùng, in kết quả ra màn hình.
- Tương tự, nếu C là in hoa thì xuống hàm compare2, kiểm tra ký tự hiện tại trong xâu có trùng với C không, sau đó lấy C + 32 để ra C để C ở dạng in thường, rồi lại kiểm tra xem các ký tự có trong xâu có trùng với C hay không, nếu có thì ta sẽ tăn biến đếm thêm một đơn vị rồi tăng địa chỉ lên một, kiểm tra ký tự tiếp theo. Cuối cùng, in kết quả ra màn hình.

- Kết quả thu được:

```
Nhap xau: cCcCCc
Nhap ky tu: C
So lan ky tu nay xuat hien trong xau la: 6
-- program is finished running --

Nhap xau: phan long
Nhap ky tu: N
So lan ky tu nay xuat hien trong xau la: 2
-- program is finished running --

Nhap xau: yoimiya
Nhap ky tu: I
So lan ky tu nay xuat hien trong xau la: 2
-- program is finished running --

Nhap xau: tagAshiRa
Nhap ky tu: a
So lan ky tu nay xuat hien trong xau la: 3
-- program is finished running --
```