BÁO CÁO GIỮA KỲ BỘ MÔN THỰC HÀNH KTMT

Họ và tên: Nguyễn Tùng Lâm

MSSV: 20204998

A – Bài 5:

Cách thực hiện:

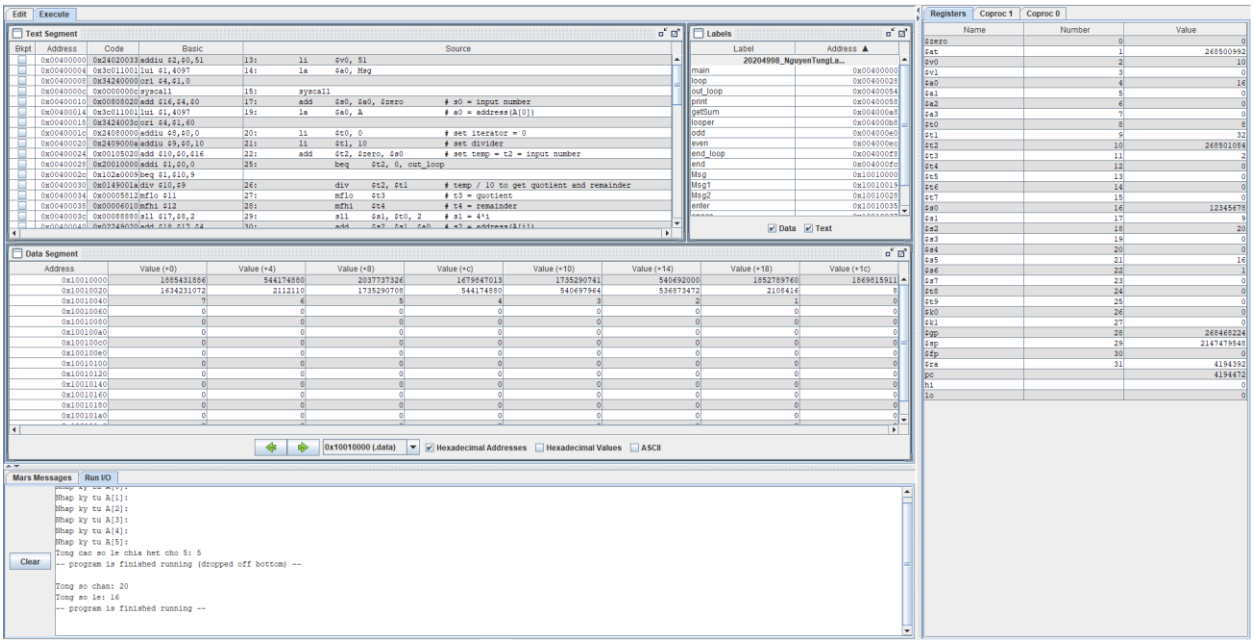
- Nhập số nguyên dương N sau đó cho từng số 1 của số nguyên dương N vào 1 mảng, VD: 123 -> [1 2 3]. Sau đó duyệt mảng và kiểm tra số nào lẻ số nào chẵn thì cộng vào tổng tương ứng

Ý nghĩa chương trình con:

- Hàm loop để tách vào khởi tạo mảng
- Hàm print để in và getSum để xử lí yêu cầu

Ảnh chụp màn hình:

- Case đã chạy để test: N = 12345678 => tổng lẻ = 1 + 3 + 5 + 7 = 16, tổng chẵn = 2 + 4 + 6 + 8 = 20



Mã nguồn:

```
.data
Msg:    .asciiz    "Nhap so nguyen duong N: "
Msg1:   .asciiz    "Tong so chan: "
Msg2:   .asciiz    "Tong so le: "
enter:  .asciiz    "\n"
```

```
space:          .asciiz         " "
commas:         .asciiz         ", "
A:              .word           0:100       # create a table with 100 elements = 0

.text
# read integer with pop up
main:
        li      $v0, 51
        la      $a0, Msg
        syscall

        add     $s0, $a0, $zero  # s0 = input number

        la      $a0, A                  # a0 = address(A[0])
        li      $t0, 0                  # set iterator = 0
        li      $t1, 10         # set divider
        add     $t2, $zero, $s0         # set temp = t2 = input number

        loop:
                beq     $t2, 0, out_loop
                div     $t2, $t1# temp / 10 to get quotient and remainder
                mflo    $t3             # t3 = quotient
                mfhi    $t4             # t4 = remainder
                sll     $s1, $t0, 2     # s1 = 4*i
                add     $s2, $s1, $a0   # s2 = address(A[i])
                sw      $t4, 0($s2)     # A[i] = t4 = remainder
                add     $t2, $t3, $zero  # temp = temp / 10
                addi    $t0, $t0, 1     # increase iterator
                j       loop
```

```
    out_loop:
        jal     getSum

    print:
        li      $v0, 4
        la      $a0, Msg1
        syscall

        li      $v0, 1
        add     $a0, $s2, $zero
        syscall

        li      $v0, 4
        la      $a0, enter
        syscall

        li      $v0, 4
        la      $a0, Msg2
        syscall

        li      $v0, 1
        add     $a0, $s5, $zero
        syscall

        # stop the program
        li      $v0, 10
        syscall

getSum:
```

```
        li      $s1, 0                  # i = 0
        li      $s2, 0                  # sum1 = 0 - even
        li      $s5, 0                  # sum2 = 0 - odd
        li      $t3, 2                  # divide for 2
        # subi  $t0, $t0, 1             # n = n - 1
        looper:
                slt     $s6, $t0, $s1   # if i > n => quit
                beq     $s6, 1, end_loop
                sll     $t1, $s1, 2     # t1 = 4*i
                add     $t2, $t1, $a0   # t2 = address(A[i])
                lw      $t4, 0($t2)     # t4 = A[i]
                div     $t4, $t3# A[i] / 2
                mfhi    $t5             # t5 = remainder = A[i] % 2
                beq     $t5, 0, even    # if t5 == 0 then plus to even sum
        odd:
                add     $s5, $t4, $s5   # sum2 += A[i] - odd sum
                addi    $s1, $s1, 1     # i = i + 1
                j       looper
        even:
                add     $s2, $t4, $s2   # sum1 += A[i]
                addi    $s1, $s1, 1     # i = i + 1
                j       looper
        end_loop:
                jr      $ra
end:
```

B – Bài 3:

Cách thực hiện:

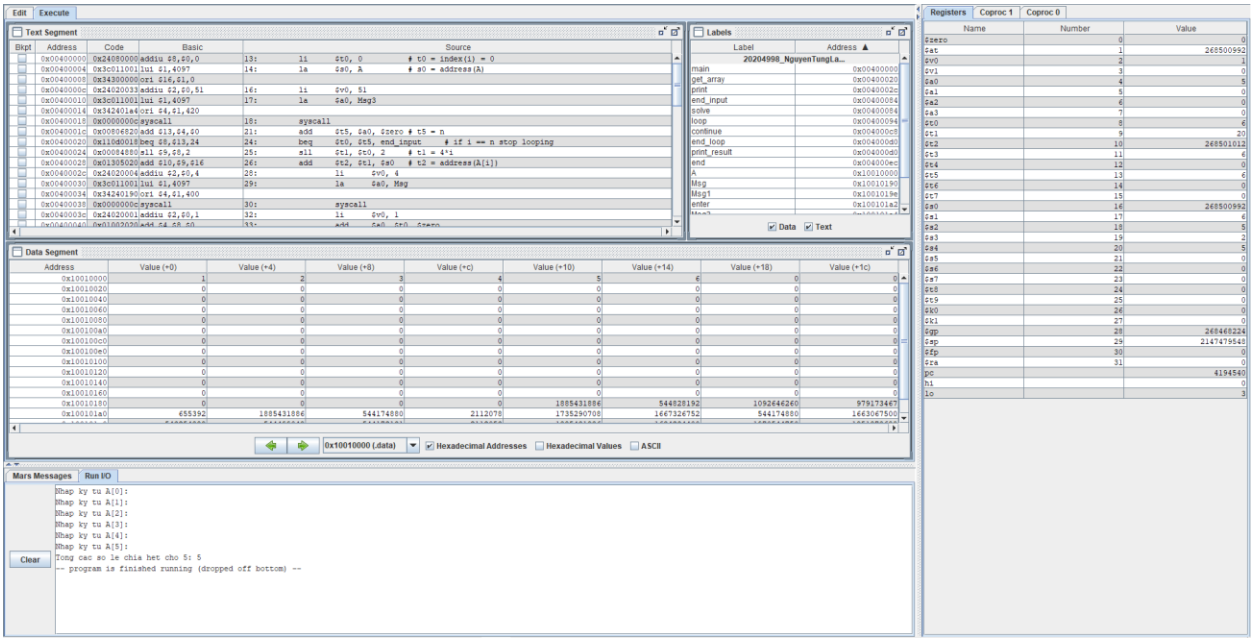
- Nhập số nguyên dương N sau đó nhập từng giá trị của mảng, sau khi có mảng chỉ việc duyệt rồi xét phần tử hiện tại có lẻ và chia hết cho 5 không? Nếu không thỏa mãn 1 trong 2 điều kiện thì duyệt phần tử tiếp và ngược lại thì cộng vào tổng rồi in ra kết quả

Ý nghĩa chương trình con:

- Chương trình con có ý nghĩa như tên đã gọi trong code

Ảnh chụp màn hình:

- Case đã chạy: n = 6, mảng [1 2 3 4 5 6] => tổng = 5 vì chỉ có 5 thỏa mãn



Mã nguồn:

```
.data
A:      .word       0:100           # declaring table
Msg:    .asciiz     "Nhap ky tu A["
Msg1:   .asciiz     "]: "
enter:  .asciiz     "\n"
Msg3:   .asciiz     "Nhap so N: "
Msg4:   .asciiz     "Tong cac so le chia het cho 5: "
Msg5:   .asciiz     "Nhap phan tu mang:"
space:  .asciiz     " "
```

```
.text
main:
        li      $t0, 0          # t0 = index(i) = 0
        la      $s0, A          # s0 = address(A)

        li      $v0, 51
        la      $a0, Msg3
        syscall

        # a0 is now number of elements of the array
        add     $t5, $a0, $zero  # t5 = n

get_array:
        beq     $t0, $t5, end_input     # if i == n stop looping
        sll     $t1, $t0, 2     # t1 = 4*i
        add     $t2, $t1, $s0   # t2 = address(A[i])
        print:
                li      $v0, 4
                la      $a0, Msg
                syscall

                li      $v0, 1
                add     $a0, $t0, $zero
                syscall

                li      $v0, 4
                la      $a0, Msg1
                syscall
```

```
        li      $v0, 51
        la      $a0, Msg5
        syscall

    sw      $a0, 0($t2)     # a0 = input number = A[i]

        li      $v0, 4
        la      $a0, enter
        syscall

    addi    $t0, $t0, 1     # i = i + 1
    j       get_array

end_input:
    # subi  $t0, $t0, 2     # last index = n - 1

solve:
    li      $s1, 0          # s1 = index(i) = 0
    li      $s2, 0          # sum = s2 = 0
    li      $s3, 2          # divide for 2
    li      $s4, 5          # divide for 5
    loop:
        beq     $s1, $t0, end_loop      # if i == n then quit
        sll     $t1, $s1, 2             # t1 = 4*i
        add     $t2, $t1, $s0           # t2 = address(A[i])
        lw      $t3, 0($t2)             # t3 = A[i]
        div     $t3, $s3        # Check A[i] odd
        mfhi    $t4                     # Get remainder
        beq     $t4, 0, continue # even => skipping
```

```
        div     $t3, $s4            # Check A[i] % 5 == 0
        mfhi    $t4                 # Get remainder
        bne     $t4, 0, continue    # A[i] % 5 !== 0 => skip
        add     $s2, $s2, $t3       # sum += A[i]
    continue:
        addi    $s1, $s1, 1         # i = i + 1
        j       loop
    end_loop:


print_result:
    li      $v0, 4
    la      $a0, Msg4
    syscall

    li      $v0, 1
    add     $a0, $zero, $s2
    syscall
end:
```

C – Bài 4:

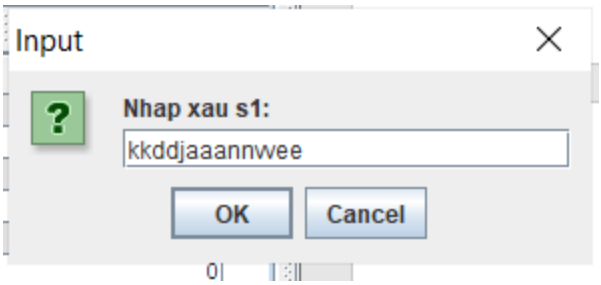
Cách thực hiện:

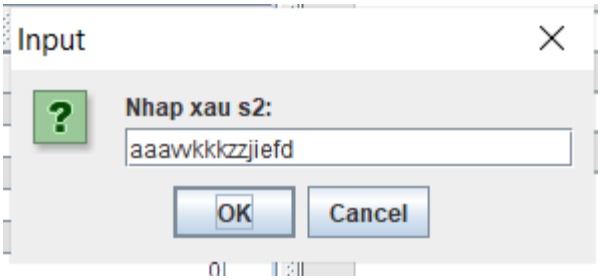
- Tạo 2 chương xâu s1 s2 và thực hiện lấy input, sau đó tạo 2 xâu temp để lưu các ký tự khác biệt (các ký tự xuất hiện trong 2 xâu) và sau đó tìm ký tự xâu temp1 trong xâu temp2, nếu xuất hiện thì in ra

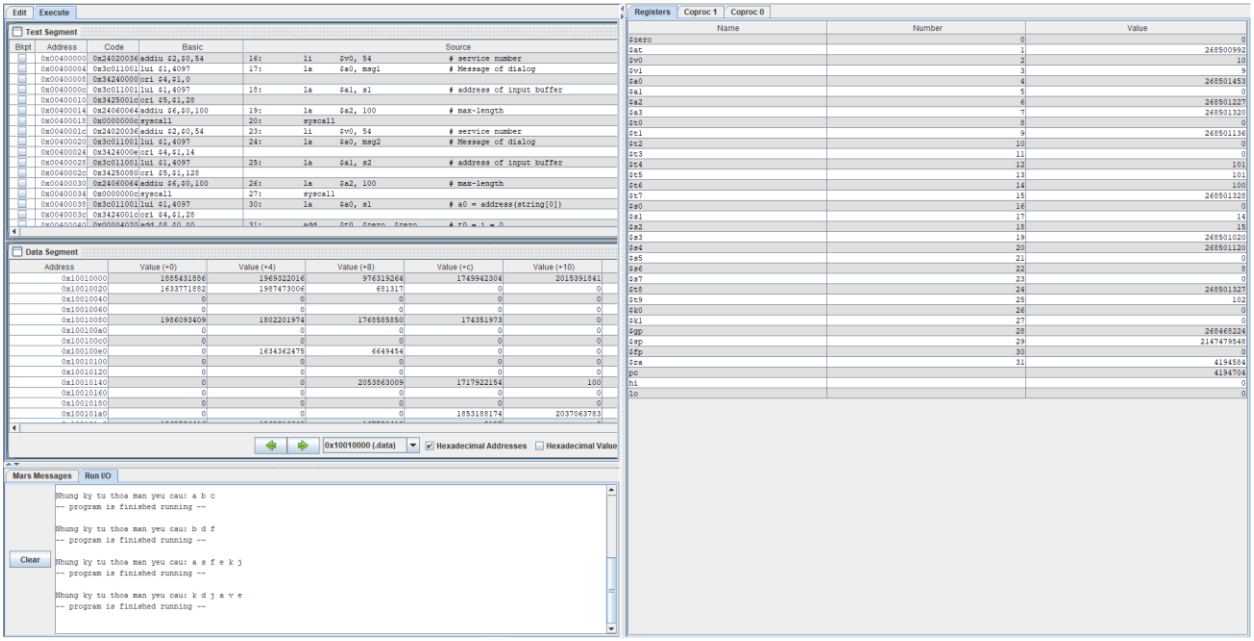
Ý nghĩa chương trình con:

- Hàm get string dùng để lấy 2 xâu mặc định
- Hàm get length để lấy độ dài mỗi xâu
- Hàm case1, case2 để khởi tạo 2 mảng temp
- Hàm find_char1, 2 dùng để tìm xem ký tự đã xuất hiện/ được thêm chưa?
- Hàm print in kết quả

Ảnh chụp kết quả:







- Ta thu được kết quả đúng

Mã nguồn:

```
.data
msg1:       .asciiz     "Nhap xau s1: "
msg2:       .asciiz     "Nhap xau s2: "
s1:         .space      100
s2:         .space      100
temp:       .space      100
temp1:      .space      100
msg3:       .asciiz     "Nhung ky tu thoa man yeu cau: "
enter:      .asciiz     "\n"
space:      .asciiz     " "
```

```
.text
main:
get_string:
        # Get string 1
        li      $v0, 54         # service number
        la      $a0, msg1               # Message of dialog
        la      $a1, s1                 # address of input buffer
        la      $a2, 100                # max-length
        syscall

        # Get string 2
        li      $v0, 54         # service number
        la      $a0, msg2               # Message of dialog
        la      $a1, s2                 # address of input buffer
        la      $a2, 100                # max-length
        syscall

get_length_s1:
        la      $a0, s1                 # a0 = address(string[0])
        add     $t0, $zero, $zero       # t0 = i = 0

check_char1:
        add     $t1, $a0, $t0           # t1 = a0 + t0 = address(s1[i])
        lb      $t2, 0($t1)             # t2 = s1[i]
        beq     $t2, $zero, exit1       # is null char ?
        addi    $t0, $t0, 1             # $t0 = $t0 + 1 <=> i = i + 1
        j       check_char1             # continue looping
exit1:
        # string 1 length = s1
```

```
        subi    $t0, $t0, 1
        add     $s1, $t0, $zero

get_length_s2:
        la      $a0, s2                 # a0 = address(s2[0])
        add     $t0, $zero, $zero       # t0 = i = 0

check_char2:
        add     $t1, $a0, $t0           # t1 = a0 + t0 = address(s2[i])
        lb      $t2, 0($t1)             # t2 = s2[i]
        beq     $t2, $zero, exit2       # is null char ?
        addi    $t0, $t0, 1             # $t0 = $t0 + 1 <=> i = i + 1
        j       check_char2             # continue looping
exit2:
        # string 2 length = s2
        subi    $t0, $t0, 1
        add     $s2, $t0, $zero

        la      $s3, s1                 # s3 = address(s1)
        la      $s4, s2                 # s4 = address(s2)

        li      $v0, 4
        la      $a0, msg3
        syscall

        la      $a2, temp               # a2 = address(temp)
        la      $a3, temp1              # a3 = address(temp1)

case1:  # iter s1
```

```
        li      $t0, 0                  # index = t0 = 0
        li      $s7, 0                  # index for temp string - j
        loop1:
                beq     $t0, $s1, end_loop1     # if index == s1.length then stop
                add     $t3, $t0, $s3           # t3 = address(s1[i])
                lb      $t5, 0($t3)             # t5 = s1[i]
                jal     find_char1              # Check if the char has been added or not
                beq     $a1, 1, continue1       # if letter has been added => skip else add
                add     $t7, $s7, $a2           # t7 = address(temp[j])
                sb      $t5, 0($t7)             # temp[j] = s1[i]
                addi    $s7, $s7, 1             # j = j + 1
        continue1:
                addi    $t0, $t0, 1             # i = i + 1
                j       loop1
        end_loop1:

        add     $v0, $s7, $zero         # v0 = temp.length
        li      $t0, 0                  # reset index = t0 = 0
        li      $s7, 0                  # reset index for temp1 string - j

case2:  # iter s2
        li      $t0, 0                  # index = t0 = 0
        li      $s7, 0                  # index for temp1 string - j
        loop2:
                beq     $t0, $s2, end_loop2     # if index == s2.length then stop
                add     $t4, $t0, $s4           # t4 = address(s2[i])
                lb      $t6, 0($t4)             # t6 = s2[i]
                jal     find_char2              # Check if the char has been added or not
                beq     $a1, 1, continue2       # if letter has been added => skip else add
```

```
        add     $t7, $s7, $a3           # t7 = address(temp1[j])
        sb      $t6, 0($t7)             # temp1[j] = s2[i]
        addi    $s7, $s7, 1             # j = j + 1
continue2:
        addi    $t0, $t0, 1             # i = i + 1
        j       loop2
end_loop2:

add     $v1, $s7, $zero         # v1 = temp1.length
li      $t0, 0                  # reset index for temp - i - s1
li      $s7, 0                  # reset index for temp1 - j - s2

# find char in 2 distinct temp array
print_char:
        lb      $t3, 0($a2)                     # t3 = temp[i])
        beq     $t3, $zero, end_print_char      # null then quit
        loop_to_find:
                lb      $t4, 0($a3)             # t4 = temp1[j])
                beq     $t4, $zero, end_print   # null then next char -> not found
                beq     $t4, $t3, print         # if 2 char equal then print
                addi    $a3, $a3, 1             # move next
                j       loop_to_find
        stop_find:
print:
        li      $v0, 11
        add     $a0, $t3, $zero
        syscall
        li      $v0, 4
        la      $a0, space
```

```
        syscall
    end_print:
        la      $a3, temp1          # reset a3 = address(temp1)
        addi    $a2, $a2, 1         # move next
        j       print_char
    end_print_char:
        # stop the program
        li      $v0, 10
        syscall

find_char1:
    li      $s6, 0          # index = s6 = 0
    li      $a1, 0          # Check = 0
    loop_find1:
        beq     $s6, $s7, end_find1     # if i == current temp.length => quit
        add     $t8, $s6, $a2           # t8 = address(temp)
        lb      $t9, 0($t8)             # t9 = temp[i]
        beq     $t9, $t5, found1        # if temp[i] = current equal char in s1/s2
        addi    $s6, $s6, 1             # i = i + 1
        j       loop_find1
    found1:
        li      $a1, 1                  # letter found => check = 1
    end_find1:
    jr      $ra

find_char2:
    li      $s6, 0          # index = s6 = 0
    li      $a1, 0          # Check = 0
    loop_find2:
```

```
        beq     $s6, $s7, end_find2     # if i == current temp1.length => quit
        add     $t8, $s6, $a3           # t8 = address(temp1[i])
        lb      $t9, 0($t8)             # t9 = temp1[i]
        beq     $t9, $t6, found2        # if temp1[i] = current char in s2
        addi    $s6, $s6, 1             # i = i + 1
        j       loop_find2
found2:
        li      $a1, 1                  # letter found => check = 1
end_find2:
jr      $ra
```