

Bài 1(4):

➤ Phân tích cách thực hiện

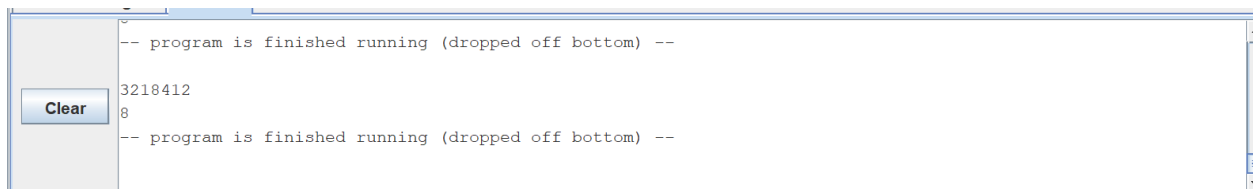
Thực hiện phép chia để kiểm tra số có độ dài bao nhiêu

Thực hiện vòng lặp với số lần lặp là độ dài của số

Thực hiện phép chia lấy số dư để so sánh với số lớn nhất hiện tại

Thực hiện chia số ban đầu cho 10 để xóa đi chữ số bên phải

➤ Ảnh chụp màn hình kết quả chạy



```
-- program is finished running (dropped off bottom) --
3218412
8
-- program is finished running (dropped off bottom) --
```

➤ Mã nguồn

```
1 .text
2 main:
3     addi    $t1, $0, 0    #max = 0
4     addi    $v0, $0, 5    #scan n=?
5     syscall
6     add     $s0, $v0, $0   #n = ?
7     addi    $a0, $0, 10
8     div     $s0, $a0
9     mflo    $a0
10    beq     $a0, $0, end_main #Exit if n < 10
11 while:    addi $a0, $0, 10   #while(
12    div     $s0, $a0         #n/10
13    mfhi    $t0             #a = n%10
14    mflo    $s0             #n=n/10
15    beq     $s0, $0, end_while #if n==0 break
16 if:      bge $t1, $t0, end_if #if max < a then max = a
17    add     $t1, $0, $t0
18 end_if:
19    j       while           #while(n>0)
20 end_while:
21    addi    $v0, $0, 1       #print max
22    add     $a0, $t1, $0
23    syscall
24 end_main:
```

Bài 2(3):

- Phân tích cách thực hiện

Tạo một mảng integer có độ dài tối đa là 100

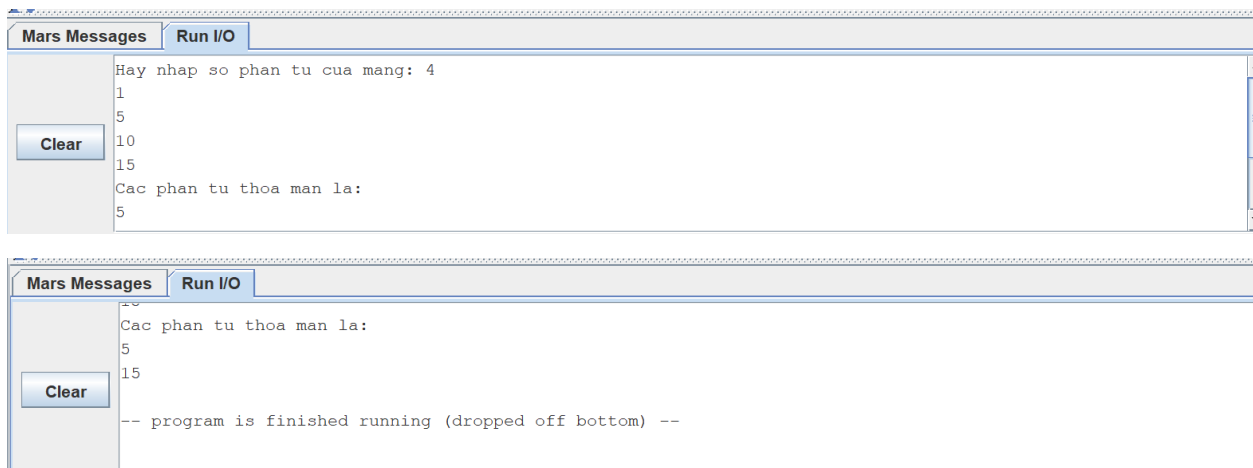
Sau khi nhập số phần tử của mảng

Thực hiện nhập lần lượt các phần tử của mảng và lưu vào mảng

Sau đó kiểm tra mỗi phần tử, để kiểm tra có thỏa mãn hay không

Chia phần dư phần tử này cho 5 và cho 2 để kiểm tra

- Ảnh chụp màn hình kết quả chạy



- Mã nguồn

```

1 .data
2 arr: .space 400
3 str: .asciiz "Hay nhap so phan tu cua mang: "
4 str1: .asciiz "Cac phan tu thoa man la: \n"
5 str2: .asciiz "\n"
6 .text
7     la    $s0, arr
8     la    $a0, str
9     addi $v0, $0, 4
10    syscall
11    addi $v0, $0, 5
12    syscall          #scan leng of array
13    add $t1, $v0, $v0
14    add $t1, $t1, $t1 #end of array
15    addi $t0, $0, 0   #i=0
16 for:
17     bge $t0, $t1, end_for #if i >= end exit
18     add $s1, $s0, $t0     #arr[i]
19     addi $v0, $0, 5       #scan integer
20     syscall
21     sw   $v0, 0($s1)      #arr[i] = scan integer
22 count:
23     addi $t0, $t0, 4      #i++
24     j     for             #
25 end_for:

```

```

26     addi $t0, $0, 0           #i=0
27     la   $a0, str1
28     addi $v0, $0, 4
29     syscall                   #print str1
30 for1:
31     bge $t0, $t1, end_for1    #if i >= end exit
32     add $s1, $s0, $t0         #arr[i]
33     lw  $v0, 0($s1)           #arr[i]
34     addi $v1, $0, 5
35     div $v0, $v1
36     mfhi $v1                  #$v1 = arr[i] % 5
37     bne $v1, $0, count1       #if arr[i] % 5 != 0 continue
38     addi $v1, $0, 2
39     div $v0, $v1
40     mfhi $v1                  #$v1 = arr[i] % 10
41     beq $v1, $0, count1       #if arr[i] % 2 == 0 continue
42     addi $a0, $v0, 0
43     addi $v0, $0, 1           #print arr[i]
44     syscall
45     la   $a0, str2
46     addi $v0, $0, 4
47     syscall                   #print "\n"
48 count1:
49     addi $t0, $t0, 4
50     j    for1

```

Bài 3(5):

- Phân tích cách thực hiện
- Ảnh chụp màn hình kết quả
- Mã nguồn

```
1 .data
2 _ascii: .space 256
3 _str: .space 256
4 end_str:
5 .text
6 la $s0, _ascii #_ascii
7 la $s1, _str #str <=> end of _ascii
8 addi $v0, $0, 8
9 add $a0, $s1, $0
10 addi $a1, $0, 256
11 syscall #scan str
12 addi $t0, $0, 0
13 addi $s2, $s0, 0 #ascii
14 while_1:
15 beq $s2, $s1, end_while_1 #if _ascii == end of _ascii => exit while_1
16 sb $0, 0($s2) #_ascii[i]=0;
17 addi $s2, $s2, 1 #address _ascii ++
18 j while_1
19 end_while_1:
20 la $v1, end_str #end of _str
21 while:
22 bge $s1, $v1, end_while #if _str == end of _str => exit while
23 lb $t1, 0($s1) #_str[i]
24 beq $t1, $0, end_while #if _str[i] == '\0' => exit while
```

```

25
26     addi    $t3, $0, 97
27     beq     $t1, $t3, work      #if_str[i] = 'a'
28     addi    $t3, $0, 101
29     beq     $t1, $t3, work      #if_str[i] = 'e'
30     addi    $t3, $0, 105
31     beq     $t1, $t3, work      #if_str[i] = 'i'
32     addi    $t3, $0, 111
33     beq     $t1, $t3, work      #if_str[i] = 'o'
34     addi    $t3, $0, 117
35     beq     $t1, $t3, work      #if_str[i] = 'u'
36
37     addi    $t3, $0, 65
38     beq     $t1, $t3, work      #if_str[i] = 'A'
39     addi    $t3, $0, 69
40     beq     $t1, $t3, work      #if_str[i] = 'E'
41     addi    $t3, $0, 73
42     beq     $t1, $t3, work      #if_str[i] = 'I'
43     addi    $t3, $0, 79
44     beq     $t1, $t3, work      #if_str[i] = 'O'
45     addi    $t3, $0, 85
46     beq     $t1, $t3, work      #if_str[i] = 'U'
47     j       continue
48

```

```

49 work:
50     add     $t2, $s0, $t1      #_ascii[_str[i]]
51     lb      $t1, 0($t2)
52     addi    $t1, $t1, 1        #_ascii[_str[i]]++
53     sb      $t1, 0($t2)
54 continue:
55     addi    $s1, $s1, 1        #address_str++
56     j       while
57 end_while:
58 while_:
59     bge     $s0, $v1, end_while_
60     lb      $t1, 0($s0)
61     beq     $t1, $0, while_
62     add     $a0, $0, $t1
63     addi    $s0, $s0, 1
64 end_while_:
65     addi    $v0, $0, 1
66     syscall

```