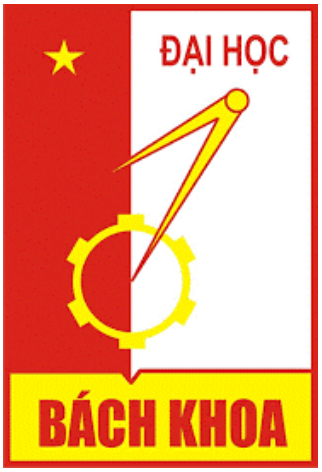
TRƯỜNG ĐẠI HỌC BÁCH KHOA HÀ NỘI

TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG



BÁO CÁO PROJECT
THỰC HÀNH KIẾN TRÚC MÁY TÍNH

GVHD: Lê Bá Vui

Nhóm: 12

Thành Viên: Phùng Ngọc Vinh - 20194719

Ngô Thị Huyền Diệu - 20194506

Mã Lớp: 130939

MỤC LỤC

LỜI MỞ ĐẦU

Nhóm gồm 2 thành viên Phùng Ngọc Vinh và Ngô Thị Huyền Diệu. Bản báo cáo khái quát quá trình thực hiện 2 Project cuối kỳ là bài 8 (Phùng Ngọc Vinh) và bài 10 (Ngô Thị Huyền Diệu) gồm các nội dung chính:

- ❖ Đề bài.
- ❖ Phân tích đề bài.
- ❖ Cách làm.
- ❖ Mã nguồn.
- ❖ Hình ảnh kết quả.

Bản báo cáo sẽ không tránh khỏi những sai sót, vì vậy nhóm rất mong nhận được ý kiến góp ý của thầy giáo và các bạn.

Chúng em chân thành cảm ơn.

I. Bài 8:

Chủ đề: Mô phỏng ổ đĩa RAID5

1. Đề bài

8. Mô phỏng ổ đĩa RAID 5

Hệ thống ổ đĩa RAID5 cần tối thiểu 3 ổ đĩa cứng, trong đó phần dữ liệu parity sẽ được chứa lần lượt lên 3 ổ đĩa như trong hình bên. Hãy viết chương trình mô phỏng hoạt động của RAID 5 với 3 ổ đĩa, với giả định rằng, mỗi block dữ liệu có 4 kí tự. Giao diện như trong minh họa dưới. *Giới hạn chuỗi kí tự nhập vào có độ dài là bội của 8.*

Trong ví dụ sau, chuỗi kí tự nhập vào từ bàn phím *(DCE.****ABCD1234HUSTHUST*) sẽ được chia thành các block 4 byte. Block 4 byte đầu tiên "DCE." sẽ được lưu trên Disk 1, Block 4 byte tiếp theo "****" sẽ lưu trên Disk 2, dữ liệu trên Disk 3 sẽ là 4 byte parity được tính từ 2 block đầu tiên với mã ASCII là 6e='D' xor '*' ; 69='C' xor '*'; 6f='E' xor '*' ; 04='.' xor '*'

```
Nhap chuoi ki tu : DCE.****ABCD1234HUSTHUST
     Disk 1                Disk 2                Disk 3
 --------------        --------------        --------------
|      DCE.      |      |      ****      |      [[ 6e,69,6f,04]]
|      ABCD      |      [[ 70,70,70,70]]      |      1234      |
[[ 00,00,00,00]]      |      HUST      |      |      HUST      |
 --------------        --------------        --------------
```

2. Phân tích đề bài

- Mô phỏng ổ đĩa RAID 5 bằng cách nhập vào từ bàn phím một chuỗi kí tự với độ dài là một số chia hết cho 8.
- Chia chuỗi vừa nhập thành các block 4 byte. Và sau đó , ta lấy từng kí tự của block 4 byte này 'xor' với từng kí tự của block 4 byte kế tiếp.
- Sau khi tính toán xong, ta hiển thị các khối block 4 byte và block parity tương ứng theo ví dụ trong đề bài đưa ra.

3. Phân tích cách làm bài

- Quá trình dưới đây được lặp lại vô hạn cho tới khi người dùng không muốn thực hiện nữa:
  - Bước 1: Nhập chuỗi đầu vào từ bàn phím và lưu vào bộ nhớ.
  - Bước 2: Đếm độ dài chuỗi và kiểm tra độ dài chuỗi có chia hết cho 8 không. Nếu không chia hết cho 8

thì thông báo lỗi và quay lại bước 1. Nếu chia hết cho 8 thì đến bước 3.

- Bước 3: Lấy từ chuỗi kí tự nhập vào từ bàn phím, ta lấy lần lượt 2 dãy - mỗi dãy gồm 4 kí tự - và lưu vào 2 biến chứa mảng khác nhau. Kết quả sau khi tính toán sẽ được lưu trong 1 biến chứaa mảng riêng biệt.
- Bước 4: Ta dùng 1 biến đếm để làm cờ hiệu cho số thứ tự dòng mà ta muốn in ra.
  + Nếu cờ hiệu bằng 1: (line1) Block1,Block2,Block Parity
  + Nếu cờ hiệu bằng 2: (line2) Block1,Block Parity,Block2
  + Nếu cờ hiệu bằng 3: (line3) Block Parity,Block1,Block2

4. Mã nguồn

```
.data
d1:         .space 4
d2:         .space 4
array:      .space 32
string:     .space 1000
hex_result:     .space 8
start:      .asciiz "Nhap chuoi ky tu : "
enter:      .asciiz "\n"
error_length:   .asciiz "Do dai chuoi khong hop le! Ban co muon nhap lai 1 lan nua ? \n"
m1:         .asciiz "      Disk 1                Disk 2 Disk 3\n"
m2:         .asciiz "----------------       ----------------       ----------------\n"
slash:      .asciiz "|"
space:      .asciiz "     "
```

```
space2:     .asciiz "       "
m3:         .asciiz "[[ "
m4:         .asciiz "]]"
comma:      .asciiz ","
ms_again :  .asciiz "Ban co muon tiep tuc chuong trinh
\n ?"


.text
main:
   jal input
   nop
   jal output
   nop


   li $v0, 50         # service 50 is ConfirmDialog . a0
= 0 :  yes
   la $a0, ms_again   # the message to user
   syscall       # execute
   beq $a0,0,main     # if a0 = 0 => jump to main
   nop
exit:
   li $v0, 10         # service 10 is exit
   syscall            # execute



#====================================INPUT==============
================================================
# t3 = length ; $s0 = input_string
input: li $v0, 54         # service 54 is input diaglog
string
```

```
   la $a0, start      # the message to user
   la $a1, string          # $a1 = address of input
buffer. $a1 contains status value : 0 : ok
   la $a2, 1000       # maximum number of characters to
read is 1000
   syscall       # execute
   bne $a1,$0,error   # if a1 !=0 => goi den ham error1
   la $s0,string      # s0 chua dia chi xau moi nhap



#-----------------------kiem tra do dai co chia het cho
8 khong--------------------------
length: addi $t3, $zero, 0     # t3 = length


check_char: add $t1, $s0, $t3  # t1 = address of
string[i]
   lb $t2, 0($t1)          # t2 = string[i]
   nop
   beq $t2, 10, check_length   # t2 = '\n' ket thuc xau
   nop
   addi $t3, $t3, 1   # length++
   j check_char       # tiep tuc check_char
   nop
check_length:
   move $t5, $t3      # t5 = t3 = length of input string
   addi $t4,$0,8      # t4 = 8
   div $t3, $t4       # Lo = $t3 / $t4, Hi = $t3 mod $t4
   mfhi $t4      # $t4 = Hi =  $t3 mod $t4 (t4 la so
du )
```

```
  bne $t4,0,error         # if $t4 != 0 (length khong
chia het cho 8) => goi ham error
  jr $ra              # nhay ve main
  nop


error:
  li $v0, 50          # service 50 is ConfirmDialog
  la $a0, error_length    # set $a0 to error_length's
address. The message to user .
  syscall       # execute
  beq $a0,0,input         # $a0 contains value of user-
chosen option . 0: Yes 1: No 2: Cancel. If a0 = 0 =>
turn back to input
  j exit              # jump to exit
  nop
#==================================OUTPUT==========
==============================
output:
  addi $sp, $sp, -4  # sp = sp - 4
  sw $ra, 0($sp)          # store $ra contents into
effective memory word address . Luu dia chi tro ve main
vao main


  li $v0, 4     # service 4 is print string
  la $a0, m1          # the string to be printed is
string "   Disk 1                Disk 2
Disk 3\n"
  syscall             # execute
  li $v0, 4     # service 4 is print string
  la $a0, m2          # the string to be printed is
string "----------------       ----------------       --
--------------\n"
```

```
   syscall            # execute


   li $s7,1      # $t7 = 1 (dat flag)
output_loop:
   addi $t0, $zero, 0 # t0 = 0 (t0 la so thu tu cac byte
da duoc xu li trong 1 block)
   la $s1, d1         # s1 : mang chua cac ki tu o block
4 byte dau
   la $s2, d2         # s2 : mang chua cac ki tu o block
4 byte tiep theo
   la $a2, array      # a2 : mang chua cac byte partity
a1:                   #block 4 byte k
   lb $t1, ($s0)      #t1 = string[i]
   addi $t3, $t3, -1  # t3 = t3 - 1
   sb $t1, ($s1)      # $s1[] = $t1
a2:                   #block 4 byte k+1
   add $s5, $s0, 4         # $s5 = $s0 + 4
   lb $t2, ($s5)      # t2 = string[i+4]
   addi $t3, $t3, -1  # t3 = t3 - 1
   sb $t2,($s2)
pitity_block:
   xor $a3, $t1, $t2  # set $a3 to bitwise XOR of $t1 and
$t2 . $a3 = $t1 XOR $t2
   sw $a3, ($a2)      # store $a3 contents into effective
memory word address
   addi $a2, $a2, 4   # a2 = a2 + 4
   addi $t0, $t0, 1   # t0 = t0 +1 (them 1 byte da duoc
xu ly)
   addi $s0, $s0, 1   # s0 = s0 + 1
   addi $s1, $s1, 1   # s1 = s1 + 1
```

```
   addi $s2, $s2, 1   # s2 = s2 + 1
   beq $t0, 4, reset  # if t0 = 4 => goi ham reset
   j a1          # else => jump to a1 : tiep tuc xu li
cac cap ki tu
   nop
reset: la $s1, d1
   la $s2, d2
   la $a2, array


case1:
   addi  $t6, $0, 1        # $t6 = 1
   bne   $s7, $t6, case2       # s7 == 1? if not, skip
to case 2
   jal   print_a1          # goi toi thu tuc print_a1
   nop
   jal   print_a2          # goi toi thu tuc print_a2
   nop
   jal   print_parity_array    # goi toi thu tuc
print_parity_array
   nop
   j     done              # jump to done
case2:
   addi  $t6, $0, 2   # $t6 = 2
   bne   $s7, $t6, case3    # t7 == 2? if not, skip to
case 3
   jal   print_a1
   nop
   jal   print_parity_array
   nop
   jal   print_a2
```

```
  nop
  j     done
case3:
  jal   print_parity_array
  nop
  jal   print_a1
  nop
  jal   print_a2
  nop
  j     done
done:
  li $v0, 4          # service 4 is print string
  la $a0, enter           # the string to be printed is
'\n'
  syscall                 # execute
  beq $t3, 0, exit1       # if t3 = 0 (da xu li het tat
ca cac ky tu) => jump to exit1
  addi $s0, $s0, 4        # s0 = s0 + 4 ; (tang i len
4 )
  beq $s7,3,else          # if s7 = 3 => jump to else
  addi $s7,$s7,1          # else s7 = s7 + 1;
  j output_loop           # jump to output_loop. Tiep
tuc vong lap
  nop
else:
  addi $s7,$0,1           # s7 = 0
  j output_loop           # # jump to output_loop. Tiep
tuc vong lap
  nop
print_a1:
```

```
  li $v0, 4          # service 4 is print string
  la $a0, slash          # string to be printed is "|"
  syscall                # execute
  li $v0, 4          # service 4 is print string
  la $a0, space          # string to be printed is "
"
  syscall                # execute
  addi $t9, $zero, 0          # t9 = 0  (index)
print_a1_loop:
  lb $a0, ($s1)          # load byte : set $a0 to sign-
extended 8bit value from $s1 memory byte address . ao
chua ky tu dau tien trong s1
  li $v0, 11             # service 11 is print
character
  syscall                # execute
  addi $t9, $t9, 1       # t9 = t9 + 1
  addi $s1, $s1, 1       # s1 = s1 + 1 (chuyen toi ky
tu tiep theo)
  beq $t9, 4, end_print_a1    # if t9 = 4(nhom 4 ky tu
da dc in ra) => jump to end_print_a1
  j print_a1_loop             # else jump to
print_a1_loop . Tiep tuc vong lap in ky tu trong block


end_print_a1:
  li $v0, 4          # service 4 is print string
  la $a0, space          # string to be printed is "
"
  syscall                # execute
  li $v0, 4          # service 4 is print string
  la $a0, slash          # string to be printed is "|"
  syscall                # execute
```

```
   li $v0, 4          # service 4 is print string
   la $a0, space2          # string to be printed is "
"
   syscall                 # execute
   jr   $ra           # nhay ve cac case goi toi tuong
ung
   nop
print_a2:
   li $v0, 4
   la $a0, slash
   syscall
   li $v0, 4
   la $a0, space
   syscall
   addi $t9, $zero, 0
print_a2_loop:
   lb $a0, ($s2)           # load byte : set $a0 to sign-
extended 8bit value from $s1 memory byte address . ao
chua ky tu dau tien trong s2
   li $v0, 11
   syscall
   addi $t9, $t9, 1
   addi $s2, $s2, 1
   beq $t9, 4, end_print_a2
   j print_a2_loop


end_print_a2:
   li $v0, 4
   la $a0, space
   syscall
```

```
   li $v0, 4
   la $a0, slash
   syscall
   li $v0, 4
   la $a0, space2
   syscall
   jr $ra
   nop
print_parity_array:
   addi $sp, $sp, -4       # sp = sp -4
   sw $ra, 0($sp)          # luu lai dia chi tro ve case
goi tuong tuong ung vao dau ngan xep
   li $v0, 4          # service 4 is print string
   la $a0, m3              # string to be printed is "[["
   syscall                 # execute
   addi $t9, $zero, 0          # t9 = 0 (index)
print_parity_array_loop:
   lb $t8, ($a2)           # load byte
   jal hex_convert             # goi toi thu tuc
hex_convert
   nop
   li $v0, 4          # service 4 is print string
   la $a0, comma           # string to be printed is ","
   syscall                 # execute
   addi $t9, $t9, 1        # t9 ++
   addi $a2, $a2, 4        # tro toi gia tri tiep theo
trong mang
   beq $t9, 3, end_print_parity_array    # if t9 = 3 =>
jump to  end_print_parity_array .  in ra 3 parity dau co
dau ",", parity cuoi cung k co
```

```
   j print_parity_array_loop
end_print_parity_array:
   lb $t8, ($a2)
   jal  hex_convert
   nop
   li $v0, 4          # service 4 is print string
   la $a0, m4              # string to be printed is "]]"
   syscall                 # execute
   li $v0, 4
   la $a0, space2
   syscall
   lw $ra, 0($sp)          # load lai dia chi tro ve case
tuong ung da dc luu o dinh sp .
   addi $sp, $sp, 4        # sp = sp + 4
   jr $ra                  # tro ve case tuong ung goi
toi
   nop


#==============Hex==========================
#convert to hex and print
hex_convert:
   add $t2,$t8,$zero       #Lay gia tri so input gan vao
$t2
   li $t5,0           #flag $t5 = 0
   li $t0,8           #Gan gia tri i = 8 (So vong lap)
   la $t7,hex_result       #Lay dia chi string result


Loop_hex:
   beqz $t0,hex_convert_end    #Dieu kien thoat vong lap
i = 0
```

```
   nop
   rol $t2,$t2,4
            #Xoay vong, lan luot lay ra 4 bit (Tuong ung
voi 1 vi tri trong he co so 16)
   and $t4,$t2,0xf             #$t4 chi luu gia tri 4
bit lay ra
   ble $t4,9,Sum_hex       #Neu $t4 <= 9 gia tri se cong
them 48 (gia tri tu 0 - > 9 trong ascii)
   nop
   addi $t4,$t4,87                  #Neu $t4 >9 gia tri
cong them 87 (gia tri a, b, c, d, e, f)
   j End_hex          #Chuyen huong xuong end_hex
   nop


Sum_hex:
   addi $t4, $t4, 48       #Neu $t4 <= 9 gia tri se cong
them 48 (gia tri tu 0 - > 9 trong ascii)
End_hex:
   bgt $t0,2,End_hex2
   nop
   sb $t4, 0($t7)          #Nap vao hex_result
   addi $t7, $t7, 1        #Lay hex_result[j]
End_hex2:
   addi $t0, $t0, -1       #i = i - 1
   j Loop_hex              #Quay lai vong lap
   nop
hex_convert_end:
   la $a0,hex_result       #In ra gia tri hex
   li $v0,4
   syscall
```

```
   jr $ra                  # tro ve
   nop


#==========Exit1================================
exit1: li $v0, 4           # service 4 is print string
   la $a0, m2              # print to be printed is "----
--- --------- -----------"
   syscall                 # execute
   lw $ra, 0($sp)          # load dia chi tro ve main da
duoc luu o dinh ngan xep
   addi $sp, $sp, 4        # sp = sp + 4
   jr $ra                  # nhay ve main
   nop
```

5. Hình ảnh kết quả mô phỏng

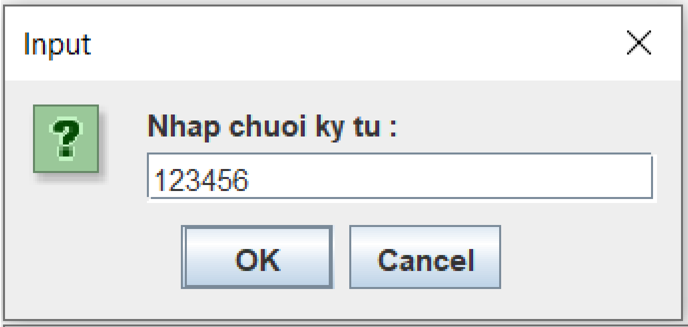
5.1 Chức năng cơ bản

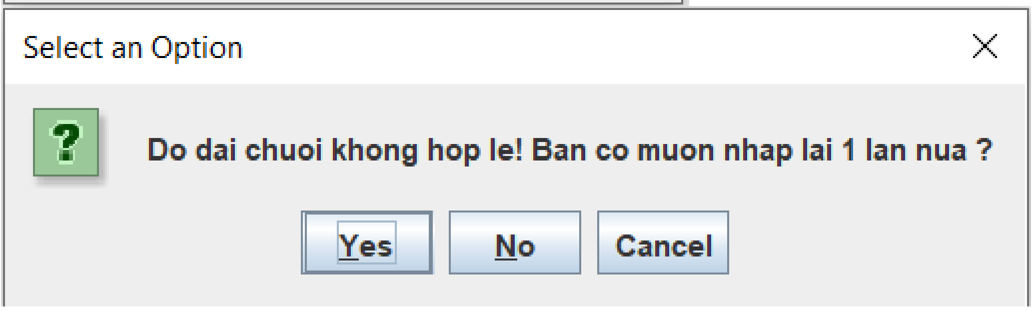
- Ta nhập xâu “*DCE.****ABCD1234HUSTHUST*” sau khi chạy chương trình

```
      Disk 1                    Disk 2                    Disk 3
----------------          ----------------          ----------------
|     DCE.     |          |     ****     |          [[ 6e,69,6f,04]]
|     ABCD     |          [[ 70,70,70,70]]          |     1234     |
[[ 00,00,00,00]]          |     HUST     |          |     HUST     |
----------------          ----------------          ----------------

-- program is finished running --
```

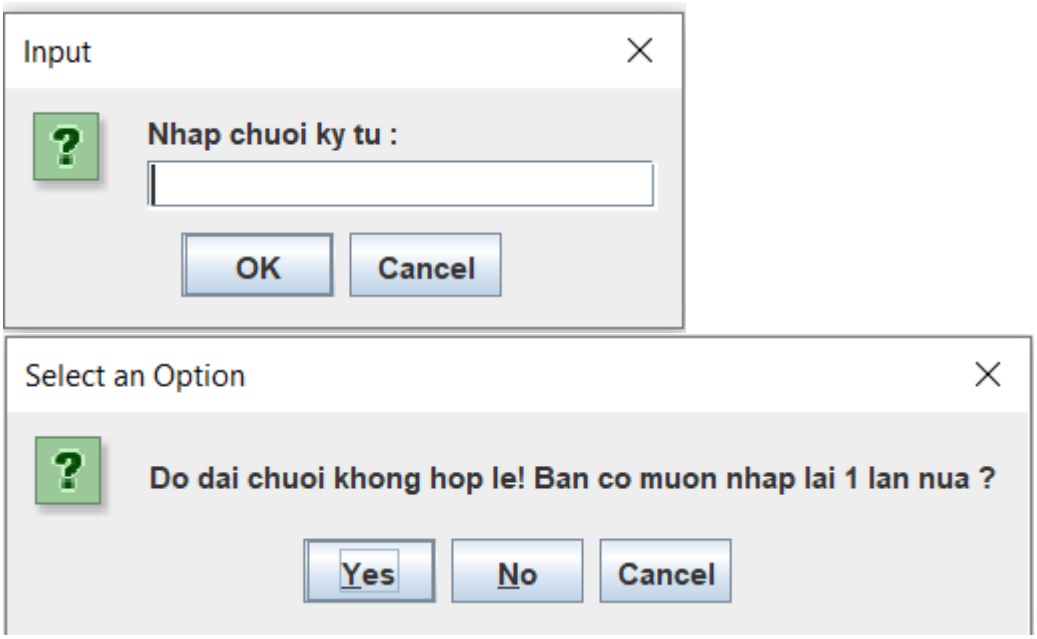
5.2 Xử lý ngoại lệ

- Xâu nhập vào có độ dài không chia hết cho 8:





- Xâu nhập vào là xâu rỗng:


Input
Nhap chuoi ky tu :
OK
Cancel
Select an Option
Do dai chuoi khong hop le! Ban co muon nhap lai 1 lan nua ?
Yes
No
Cancel

II. Bài 10:

Chủ đề: Máy tính bỏ túi

1. Đề bài

Sử dụng 2 ngoại vi là bàn phím keypad và led 7 thanh để xây dựng một máy tính bỏ túi đơn giản. Hỗ trợ các phép toán +, -, *, /, % với các toán hạng là số nguyên. Do trên bàn phím không có các phím trên nên sẽ dùng các phím:

- Bấm phím a để nhập phép tính +
- Bấm phím b để nhập phép tính –
- Bấm phím c để nhập phép tính *
- Bấm phím d để nhập phép tính /
- Bấm phím e để nhập phép tính %
- Bấm phím f để nhập phép =

Yêu cầu cụ thể như sau:

- Khi nhấn các phím số, hiển thị lên LED, do chỉ có 2 LED nên chỉ hiển thị 2 số cuối cùng. Ví dụ khi nhấn phím 1 → hiển thị 01. Khi nhấn thêm phím 2 → hiển thị 12. Khi nhấn thêm phím 3 → hiển thị 23.
- Sau khi nhập số, sẽ nhập phép tính + - * / %
- Sau khi nhấn phím f (dấu =), tính toán và hiển thị kết quả lên LED.
- Có thể thực hiện các phép tính liên tiếp (tham khảo ứng dụng Calculator trên hệ điều hành Windows)

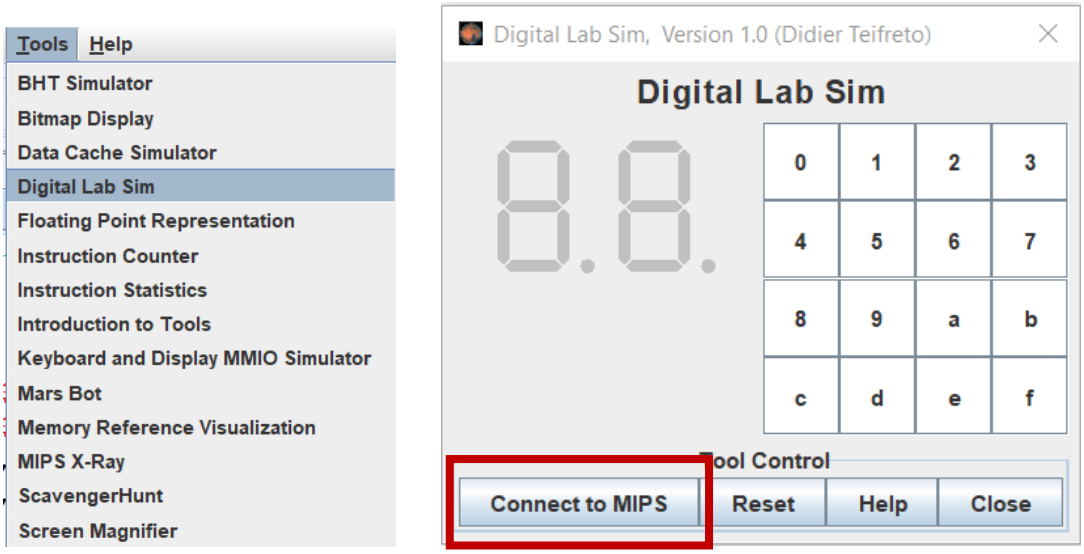
2. Phân tích đề bài

- Thực hiện các phép toán + - * / % bằng cách sử dụng bàn phím keypad và led 7 thanh.
- Yêu cầu nhập vào 2 số bất kì từ bàn phím keypad và hiển thị 2 chữ số cuối của mỗi số lên led 7 thanh (nếu số muốn thực hiện phép tính chỉ có 1 chữ số, ví dụ số 2 thì hiển thị là 02)
- Hiển thị menu lựa chọn các phép toán + - * / %

- Sau khi thực hiện phép toán thì hiển thị lên led 7 thanh 2 chữ số cuối của kết quả (nếu kết quả chỉ có 1 chữ số, ví dụ số 2 thì hiển thị là 02)
- Sau khi thực hiện xong phép tính, sẽ có thông báo cho người dùng muốn tiếp tục thực hiện chương trình hay không (nếu có), quá trình chạy chương trình diễn ra vô hạn cho tới khi người dùng không muốn dùng nữa.
- Xử lí các trường hợp ngoại lệ có trong các phép toán + - * / %
  + Trường hợp kết quả là 1 số âm thì sẽ có thông báo không thể hiển thị được kết quả âm trên led 7 thanh
  + Trường hợp phép chia cho 0 thì sẽ báo lỗi
  + Vân vân …

3. Phân tích cách làm

- Quá trình dưới đây được lặp lại vô hạn cho tới khi người dùng không muốn thực hiện nữa:
  - Project 10 được tạo ra trong phạm vi phép toán với số có 4 chữ số. Người dùng cần nhập 8 chữ số bất kì từ bàn phím keypad, tương đương với 2 số để thực hiện phép tính.
    + Sử dụng 1 biến đếm có max = 4 để đánh dấu sự kết thúc nhập của từng số.
  - Hiển thị menu lựa chọn phép toán muốn thực hiện. Các phím a, b, c, d, e tương ứng với +, -, *, /, %. Dùng 1 biến lưu giá trị từ 1 → 5 để biểu diễn các case này và show kết quả/ lỗi(nếu có)
- Cách chạy chương trình:



- Connect Digital Lab Sim và click vào ô Connect to Mips.

- Click vào biểu tượng  sau đó click vào  .

4. Mã nguồn

```
.data

  welc:          .asciiz "\nChao mung den voi May tinh bo
tui Dieu Ngo!\n"

  p_int:         .asciiz "\n Hay nhap vao so thu nhat co do
dai  4 chu so,\n VD: nhan 0021 de nhap so 21 . "

  p_int1:        .asciiz "\n Hay nhap vao so thu hai co do
dai  4 chu so,\n VD: nhan 0001 de nhap so 1 . "

  p_toantu:      .asciiz "\n nhap vao toan tu \n Nhan a de
nhap phep cong\n Nhan b de nhap phep tru \n Nhan c de nhap
phep nhan\n Nhan d de nhap phep chia \n Nhan e de nhap
phep chia lay du\n "

  co:            .asciiz "\n phat hien \n"

  ketqua:        .asciiz "\n ket qua la :  "

  xuongdong:     .asciiz "\n"

  err1:          .asciiz "\n Xay ra loi. Xin hay nhap lai
toan tu\n"
```

```
  sb_nhan:      .asciiz "\n Ban da nhap phep nhan. \n"
  sb_chia:      .asciiz "\n Ban da nhap phep chia.  \n"
  sb_cong :     .asciiz "\n Ban da nhap phep cong.\n"
  sb_tru:       .asciiz "\n Ban da nhap phep tru. \n"
  sb_chialaydu: .asciiz "\n Ban da nhap phep chia lay du.
\n"
  sb_daubang:   .asciiz "\n Nhap f de hien thi ket qua"
  rep:          .asciiz "\nNhap 0 de thoat khoi chuong
trinh \n Nhap so khac 0 de tiep tuc chuong trinh \n "
  goodbye:      .asciiz "\n Cam on ban da su dung chuong
trinh, Tam biet hen gap lai\n"
  erram:        .asciiz "\nKet qua la so am, khong hien
thi duoc\n."


.eqv ZERO                   63  # Gia tri byte hien thi so
0 tren den LED
.eqv ONE                    6  # Gia tri byte hien thi so
1 tren den LED
.eqv TWO                    91  # Gia tri byte hien thi so
2 tren den LED
.eqv THREE                  79 # Gia tri byte hien thi so
3 tren den LED
.eqv FOUR                   102  # Gia tri byte hien thi
so 4 tren den LED
.eqv FIVE                   109  #Gia tri  byte hien thi
so 6 tren den LED
.eqv SIX                    125  #Gia tri  byte hien thi
so 6 tren den LED
.eqv SEVEN                  7  # Gia tri byte hien thi so
7 tren den LED
```

```
.eqv EIGHT                  127  # Gia tri byte hien thi
so 8 tren den LED
.eqv NINE                   111  # Gia tri byte hien thi
so 9 tren den LED


.eqv IN_ADDRESS_HEXA_KEYBOARD   0xFFFF0012  # chua byte
dieu khien dong cua ban phim


.eqv OUT_ADDRESS_HEXA_KEYBOARD  0xFFFF0014  # chua byte
tra ve vi tri cua phim duoc bam


.eqv LEFT_LED            0xFFFF0010  # chua byte dieu
khien den led ben phai
.eqv RIGHT_LED          0xFFFF0011  # chua byte dieu khien
den led ben trai
.text
main:


start:
     la $a0,welc
     li $v0, 4
     syscall
     la $a0,p_int
     li $v0, 4
     syscall


    #-------------------------------------------------
-----------------
    # ######### Kich hoat interupt -----------------
```

```
    #-----------------------------------------------------
-----------------


  li $t1,IN_ADDRESS_HEXA_KEYBOARD
  li $t3,0x80     # Kich hoat interupt tu ban phim hexa
  sb $t3,0($t1)


    #-----------------------------------------------------
-----------------
    # Khai bao bien
    #------------2----------------------------------------
------------------
      li $t6,0      # $t6: Bien gia tri so cua den LED
trai
      li $t7,0      # $t7: Bien gia tri so cua den LED
phai
      li $s1,0      # $s1: Gia tri lay ra
      li $s2,0      # $s2: toan tu lay ra
      li $s3,0      # $s3:  dau =
      li $s5,0         # so thu 1
      li $s6,0      # so thu 2
      li $s4,0      # bien dem s4
    #-----------------------------------------------------
-----------------
    #-----------------------------------------------------
-----------------
    # Vong lap cho tin hieu interupt so thu nhat
    #-----------------------------------------------------
-----------------
```

```
Loop: beq $s4,4,nhapso2
      nop
      beq $s4,4,nhapso2
      nop
      beq $s4,4,nhapso2
      nop
      beq $s4,4,nhapso2
      b Loop     #Wait for interrupt

    #-------------------------------------------------------
-----------------

nhapso2:

    add $s5,$s1,$0 # luu gia tri so thu nhat vao s5
    add $s1,$0,$0 # reset s1
    addi $s4,$0,0 # reset bien dem s4
    la $a0,xuongdong
    li $v0, 4
    syscall
    move $a0,$s5  #IN SO THU 1
    li $v0,1
    syscall



    li $t6,0      # reset den
    li $t7,0
```

```
    la $a0,p_int1
    li $v0,4
    syscall
Loop1: # VOng lap cho interupt so thu 2
     beq $s4,4,nhaptoantu
     nop
     beq $s4,4,nhaptoantu
     nop
     beq $s4,4,nhaptoantu
     nop
     beq $s4,4,nhaptoantu
     b Loop1      #Wait for interrupt

 nhaptoantu:

  add $s6,$s1,$0 # LUU SO THU 2 VAO S6
  add $s1,$0,$0  #RESET S1
  addi $s4,$0,0 #RESET BIEN DEM S4
  la $a0,xuongdong
  li $v0, 4
  syscall

   move $a0,$s6  # IN SO THU 2
  li $v0,1
  syscall
```

```
  li $t6,0  # reset den led
  li $t7,0

  la $a0,p_toantu
  li $v0,4
  syscall

Loop2:

    beq $s4,1,nhaptoantu2
    nop
    b Loop2

nhaptoantu2:
     add $a3,$0,$0
     add $s1,$0,$0 # reset s1
     addi $s4,$0,0 # reset bien dem s4

   cong1: bne $s2,1,tru1
       la $a0,sb_cong
       li $v0, 4
       syscall
       j baonhapdaubang
   tru1:  bne $s2,2,nhan1
       la $a0,sb_tru
          li $v0, 4
          syscall
```

```
        j baonhapdaubang
  nhan1: bne $s2,3,chia1
        la $a0,sb_nhan
        li $v0,4
        syscall
        j baonhapdaubang
  chia1: bne $s2,4,chialaydu1
        la $a0,sb_chia
        li $v0, 4
        syscall
        j baonhapdaubang
  chialaydu1: bne $s2,5,baonhapdaubang
        la $a0,sb_chialaydu
        li $v0, 4
        syscall
        j baonhapdaubang
baonhapdaubang:
        la $a0,sb_daubang
           li $v0, 4
              syscall

Loop3: # Cho nhap vao dau = de hien thi ket qua
   beq $s3,6, show
   nop
   beq $s3,6,show
   nop
   beq $s3,6,show
```

```
    nop
    beq $s3,6,show
    b Loop2    #Wait for interrupt
    beq $s3,6,show
    nop
    beq $s3,6,show
    b Loop3

      show:

    case_cong:  bne $s2,1,case_tru  # neu la phep cong
                addu $s7,$s5,$s6 # thuc hien phep cong
                la $a0,ketqua
                li $v0, 4
                syscall
                move $a0,$s7 # in ket qua ra console
                li $v0,1
                syscall
                j showketqua

    case_tru:   bne $s2,2,case_nhan
                la $a0,ketqua
                li $v0, 4
                syscall
                sub $s7,$s5,$s6
                move $a0,$s7
                li $v0,1
```

```
                syscall
                j showketqua
    case_nhan:  bne $s2,3,case_chia
                la $a0,ketqua
                li $v0,4
                syscall
                mul $s7,$s5,$s6
                move $a0,$s7
                li $v0,1
                syscall
                j showketqua
    case_chia:  bne $s2,4,case_chialaydu
          beq $s6, $0, pheptinhdf
                la $a0,ketqua
                li $v0,4
                syscall
                div $s7,$s5,$s6
                move $a0,$s7
                li $v0,1
                syscall
          j showketqua
    case_chialaydu: bne $s2,5,pheptinhdf
          beq $s6, $0, pheptinhdf
                la $a0,ketqua
                li $v0,4
                syscall
                div $s5,$s6
```

```
                mfhi $s7
                move $a0,$s7
                li $v0,1
                syscall
         j showketqua
 pheptinhdf: # bao loi
  la $a0, err1
  li $v0, 4
  syscall

showketqua: li $t9,0
            li $t8,0
            slt $k1,$s7,$0 # kiem tra ket  qua la so am
hay khong
            bne $k1,$0,loisoam
            div $t8,$s7,10
            mfhi $t9
            beq $t8,0,napgiatricholed

            div $t8,$t8,10
            mfhi $t8
napgiatricholed:
                 li $t2,LEFT_LED   # hien thi den LED trai
                 add $s0,$zero,$t9 # truyen bien left
                 jal hienthi
                 nop
                 li $t2,RIGHT_LED  # hien thi den LED phai
```

```
                add $s0,$zero,$t8 # truyen bien right
                jal hienthi
                nop
                j endmain


endmain:
       li $v0, 51
       la $a0,rep
       syscall
       beq $a0, $0,END  #thong bao ket thuc chuong trinh
hay la tiep tuc
       nop
       j start
       nop
END:
      la $a0, goodbye
      li $v0, 4
      syscall

      la $v0, 10
      syscall
     loisoam: # bao loi ket qua am
              la $a0, erram
              li $v0, 4
              syscall
              li $v0, 51
              la $a0,rep #thong bao ket thuc chuong trinh
hay la tiep tuc
```

```
                syscall
                beq $a0, $0,END
                nop
                j start
                nop
                la $v0, 10
                syscall



   hienthi:
showleds_0: bne $s0,0,showleds_1  # case $s0 = 0
      li $t4,ZERO
      j napgiatri
showleds_1: bne $s0,1,showleds_2  # case $s0 = 1
      li $t4,ONE
      j napgiatri
showleds_2: bne $s0,2,showleds_3  # case $s0 = 2
      li $t4,TWO
      j napgiatri
showleds_3: bne $s0,3,showleds_4  # case $s0 = 3
      li $t4,THREE
      j napgiatri
showleds_4: bne $s0,4,showleds_5  # case $s0 = 4
      li $t4,FOUR
      j napgiatri
showleds_5: bne $s0,5,showleds_6  # case $s0 = 5
      li $t4,FIVE
```

```
      j napgiatri
showleds_6: bne $s0,6,showleds_7  # case $s0 = 6
      li $t4,SIX
      j napgiatri
showleds_7: bne $s0,7,showleds_8  # case $s0 = 7
      li $t4,SEVEN
      j napgiatri
showleds_8: bne $s0,8,showleds_9  # case $s0 = 8
      li $t4,EIGHT
      j napgiatri
showleds_9: bne $s0,9,showleds_df # case $s0 = 9
      li $t4,NINE
      j napgiatri
showleds_df:  jr $ra
napgiatri:    sb $t4,0($t2)
           jr $ra


#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
# Xu ly khi xay ra interupt
# Hien thi so vua bam len den led 7 doan
#~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
~~~~~~~~~~~~~~~
.ktext 0x80000180
  #-----------------------------------------------------
------------------
  # SAVE the current REG FILE to stack
```

```
  #-------------------------------------------------------
------------------


IntSR:addi $sp,$sp,4    # Save $ra because we may change
it later
      sw $ra,0($sp)
      addi $sp,$sp,4    # Save $ra because we may change
it later
      sw  $at,0($sp)
      addi $sp,$sp,4    # Save $ra because we may change
it later
      sw $v0,0($sp)
      addi $sp,$sp,4    # Save $a0, because we may change
it later
      sw $a0,0($sp)
      addi $sp,$sp,4    # Save $t1, because we may change
it later
      sw $t1,0($sp)
      addi $sp,$sp,4    # Save $t3, because we may change
it later
      sw $t3,0($sp)
      addi $sp,$sp,4
      sw $s4,0($sp)
  # -----------------------------------------------------
----------------
  # Processing
  # -----------------------------------------------------
----------------
      addi $s4,$s4,1
      jal getInt1
```

```
      nop
      jal getInt2
      nop
      jal getInt3
      nop
     jal getInt4
      nop
next_pc:    mfc0 $at,$14    # $at <= Copro0.$14 =
Coproc0.epc
      addi $at,$at,4    # $at = $at + 4
      mtc0 $at,$14    #Coproc0.$14 = Coproc0.epc <= $at
  #-----------------------------------------------------
----------------
  #-----------------------------------------------------
----------------
  # RESTORE the REG FILE from STACK
  #-----------------------------------------------------
----------------
restore:
    #  lw $s4,0($sp)
     # addi $sp,$sp,-4
      lw $t3,0($sp)
      addi $sp,$sp,-4
      lw $t1,0($sp)
      addi $sp,$sp,-4
      lw $a0,0($sp)
      addi $sp,$sp,-4
      lw $v0,0($sp)
```

```
      addi $sp,$sp,-4
      lw $ra,0($sp)
      addi $sp,$sp,-4
      lw $t4,0($sp)
      addi $sp,$sp,-4
back_main:  eret

  #-----------------------------------------------------
-----------------
  # Thu tuc quet cac phim o hang 1 va xu ly
  # Tham so truyen vao:
  # Tra ve:
  #-----------------------------------------------------
-----------------
getInt1: addi $sp,$sp,4
      sw $ra,0($sp)
      li $t1,IN_ADDRESS_HEXA_KEYBOARD
      li $t3,0x81     # Kich hoat interrupt, cho phep bam
phim o hang 1
      sb $t3,0($t1)
      li $t1,OUT_ADDRESS_HEXA_KEYBOARD
      lb $t3,0($t1)   # Nhan byte the hien vi tri cua phim
duoc bam trong hang 1
case_0:   li $t5,0x11
      bne $t3,$t5,case_1  # case 0x11
      addi $t7,$t6,0    # left=right
      addi $t6,$zero,0  # left = 0
      mul $s1,$s1,10
```

```
      add $s1,$s1,$t6   # factor=factor*10+left
      j show1
case_1:   li $t5,0x21
      bne $t3,$t5,case_2  # case 0x21
      addi $t7,$t6,0    # left=right
      addi $t6,$zero,1  # left = 1
      mul $s1,$s1,10
      add $s1,$s1,$t6 # factor=factor*10+left
      j show1
case_2:   li $t5,0x41
      bne $t3,$t5,case_3  # case 0x41
      addi $t7,$t6,0    # left=right
      addi $t6,$zero,2  # left = 2
      mul $s1,$s1,10
      add $s1,$s1,$t6 # factor=factor*10+left
      j show1
case_3:   li $t5,0xffffff81
      bne $t3,$t5,case_default1  # case 0xffffff81
      addi $t7,$t6,0    # left=right
      addi $t6,$zero,3  # left = 3
      mul $s1,$s1,10
      add $s1,$s1,$t6 # factor=factor*10+left
      j show1
show1:    li $t2,LEFT_LED   # hien thi den LED trai
      add $s0,$zero,$t6 # truyen bien left
      jal displayLED
      nop
```

```
      li $t2,RIGHT_LED  # hien thi den LED phai
      add $s0,$zero,$t7 # truyen bien right
      jal displayLED
      nop
case_default1:   j getInt1rt
getInt1rt: lw $ra,0($sp)
      addi $sp,$sp,-4
      jr $ra
  #-----------------------------------------------------
------------------


    #-----------------------------------------------------
-------------------
  # Thu tuc quet cac phim o hang 2 va xu ly
  # Tham so truyen vao:
  # Tra ve:
  #-----------------------------------------------------
-----------------
getInt2: addi $sp,$sp,4
      sw $ra,0($sp)
      li $t1,IN_ADDRESS_HEXA_KEYBOARD
      li $t3,0x82     # Kich hoat interrupt, cho phep bam
phim o hang 1
      sb $t3,0($t1)
      li $t1,OUT_ADDRESS_HEXA_KEYBOARD
      lb $t3,0($t1)   # Nhan byte the hien vi tri cua phim
duoc bam trong hang 1
case_4:   li $t5,0x12
      bne $t3,$t5,case_5  # case 0x12
```

```
      addi $t7,$t6,0    # left=right
      addi $t6,$zero,4  # left = 4
      mul $s1,$s1,10
      add $s1,$s1,$t6   # factor=factor*10+left
      j show2
case_5:   li $t5,0x22
      bne $t3,$t5,case_6  # case 0x22
      addi $t7,$t6,0    # left=right
      addi $t6,$zero,5  # left = 5
      mul $s1,$s1,10
      add $s1,$s1,$t6   # factor=factor*10+left
      j show2
case_6:   li $t5,0x42
      bne $t3,$t5,case_7  # case 0x42
      addi $t7,$t6,0    # left=right
      addi $t6,$zero,6  # left = 6
      mul $s1,$s1,10
      add $s1,$s1,$t6   # factor=factor*10+left
      j show2
case_7:   li $t5,0xffffff82
      bne $t3,$t5,case_default2  # case 0xffffff82
      addi $t7,$t6,0    # left=right
      addi $t6,$zero,7  # left = 7
      mul $s1,$s1,10
      add $s1,$s1,$t6 # factor=factor*10+left
      j show2
show2:    li $t2,LEFT_LED   # hien thi den LED trai
```

```
      add $s0,$zero,$t6 # truyen bien left
      jal displayLED
      nop
      li $t2,RIGHT_LED  # hien thi den LED phai
      add $s0,$zero,$t7 # truyen bien right
      jal displayLED
      nop
case_default2:   j getInt2rt
getInt2rt: lw $ra,0($sp)
      addi $sp,$sp,-4
      jr $ra
 ###################### GETCODE 3 ##########
 getInt3: addi $sp,$sp,4
      sw $ra,0($sp)
      li $t1,IN_ADDRESS_HEXA_KEYBOARD
      li $t3,0x84     # Kich hoat interrupt, cho phep bam
phim o hang 3
      sb $t3,0($t1)
      li $t1,OUT_ADDRESS_HEXA_KEYBOARD
      lb $t3,0($t1)   # Nhan byte the hien vi tri cua phim
duoc bam trong hang 3
case_8:   li $t5,0x00000014
      bne $t3,$t5,case_9  # case 0x14
      addi $t7,$t6,0    # left=right
      addi $t6,$zero,8  # left = 8
      mul $s1,$s1,10
      add $s1,$s1,$t6   # factor=factor*10+left
      j show3
```

```
case_9:   li $t5,0x00000024
      bne $t3,$t5,case_a  # case 0x24
      addi $t7,$t6,0    # left=right
      addi $t6,$zero,9  # left = 9
      mul $s1,$s1,10
      add $s1,$s1,$t6   # factor=factor*10+left
      j show3
case_a:   li $t5,0x44
         bne $t3,$t5,case_b # case 0x44
         addi $s2,$0,1
         j case_default3

case_b:   li $t5,0xffffff84
    bne $t3,$t5,case_default3 # case 0xffffff84
    addi $s2,$0,2
  j  case_default3

show3:    li $t2,LEFT_LED   # hien thi den LED trai
      add $s0,$zero,$t6 # truyen bien left
      jal displayLED
      nop
      li $t2,RIGHT_LED  # hien thi den LED phai
      add $s0,$zero,$t7 # truyen bien right
      jal displayLED
      nop
case_default3:   j getInt3rt
getInt3rt: lw $ra,0($sp)
```

```
      addi $sp,$sp,-4
      jr $ra
#----------------------------------------getcode 4
 getInt4: addi $sp,$sp,4
      sw $ra,0($sp)
      li $t1,IN_ADDRESS_HEXA_KEYBOARD
      li $t3,0x88     # Kich hoat interrupt, cho phep bam
phim o hang 4
      sb $t3,0($t1)
      li $t1,OUT_ADDRESS_HEXA_KEYBOARD
      lb $t3,0($t1)   # Nhan byte the hien vi tri cua phim
duoc bam trong hang 4

case_c:   li $t5,0x18
          bne $t3,$t5,case_d # case 0x18
          addi $s2,$0,3
          j case_default4

case_d:   li $t5,0x28
          bne $t3,$t5,case_e # case 0x28
          addi $s2,$0,4
          j  case_default4

case_e:   li $t5,0x48
          bne $t3,$t5,case_f # case 0x48
          addi $s2,$0,5
          j  case_default4
```

```
case_f:   li $t5,0xffffff88
          bne $t3,$t5,case_default4 # case 0xffffff88
          addi $s3,$0,6
          j case_default4



case_default4:   j getInt4rt
getInt4rt:       lw $ra,0($sp)
                 addi $sp,$sp,-4
                 jr $ra


   #-------------------------------------------------------
-------------------


  #-------------------------------------------------------
------------------
  # Thu tuc hien thi den LED
  # Tham so truyen vao: $t2 (dia chi cua LEFT_LED hoac
RIGHT_LED), $s0 : bien kieu int
  # Den LED $t2 se hien thi so $s0
  #-------------------------------------------------------
------------------
displayLED: addi $sp,$sp,4
      sw $ra,0($sp)   # save $ra
display:
hienthi_0: bne $s0,0,hienthi_1  # case $s0 = 0
      li $t4,ZERO
      j assign
```

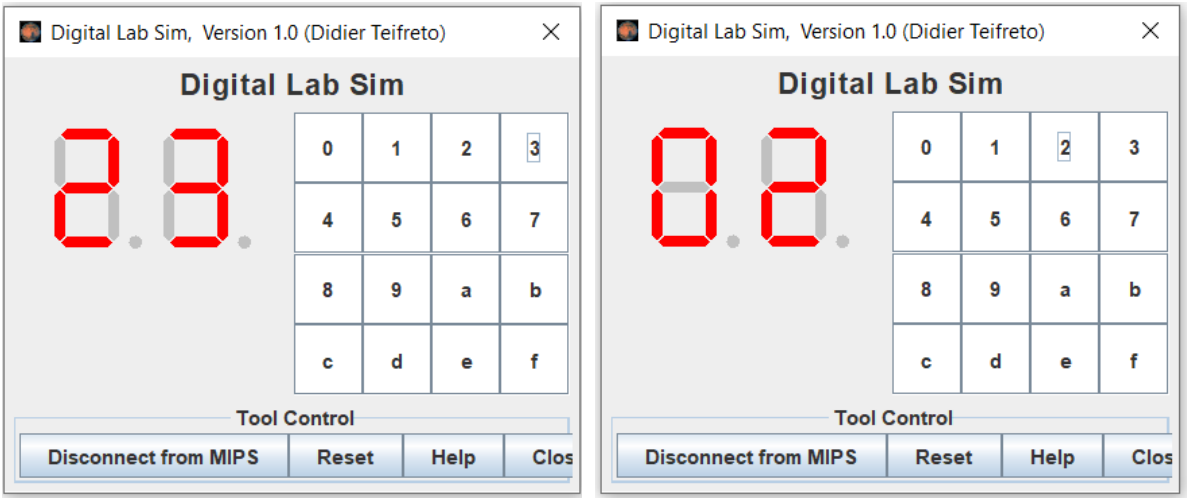
```
hienthi_1: bne $s0,1,hienthi_2  # case $s0 = 1
      li $t4,ONE
      j assign
hienthi_2: bne $s0,2,hienthi_3  # case $s0 = 2
      li $t4,TWO
      j assign
hienthi_3: bne $s0,3,hienthi_4  # case $s0 = 3
      li $t4,THREE
      j assign
hienthi_4: bne $s0,4,hienthi_5  # case $s0 = 4
      li $t4,FOUR
      j assign
hienthi_5: bne $s0,5,hienthi_6  # case $s0 = 5
      li $t4,FIVE
      j assign
hienthi_6: bne $s0,6,hienthi_7  # case $s0 = 6
      li $t4,SIX
      j assign
hienthi_7: bne $s0,7,hienthi_8  # case $s0 = 7
      li $t4,SEVEN
      j assign
hienthi_8: bne $s0,8,hienthi_9  # case $s0 = 8
      li $t4,EIGHT
      j assign
hienthi_9: bne $s0,9,hienthi_df # case $s0 = 9
      li $t4,NINE
      j assign
```

```
hienthi_df:  j displayLEDrt
assign:   sb $t4,0($t2)
displayLEDrt: lw $ra,0($sp)
      addi $sp,$sp,-4
      jr $ra
  #-----------------------------------------------------
```

5. Hình ảnh kết quả mô phỏng

5.1. Chức năng cơ bản

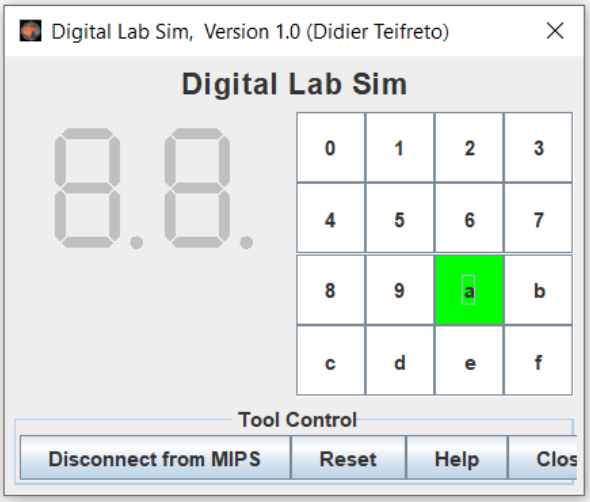
Giả sử 2 số nhập vào là 123 và 2.



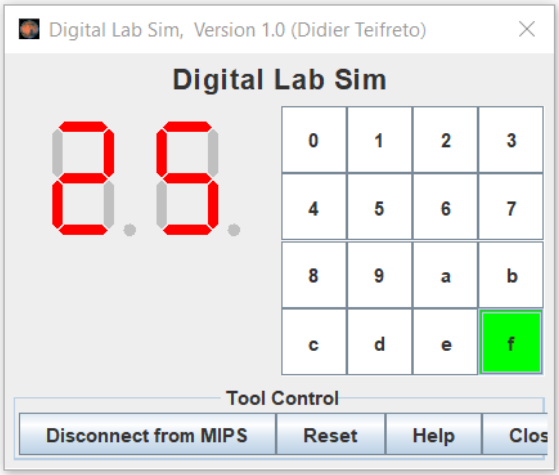

```
Chao mung den voi May tinh bo tui Dieu Ngo!

 Hay nhap vao so thu nhat co do dai  4 chu so,
 VD: nhan 0021 de nhap so 21 .
123
 Hay nhap vao so thu hai co do dai  4 chu so,
 VD: nhan 0001 de nhap so 1 .
2
 nhap vao toan tu
 Nhan a de nhap phep cong
 Nhan b de nhap phep tru
 Nhan c de nhap phep nhan
 Nhan d de nhap phep chia
 Nhan e de nhap phep chia lay du
```

- Phép cộng:

```
Chao mung den voi May tinh bo tui Dieu Ngo!

 Hay nhap vao so thu nhat co do dai  4 chu so,
 VD: nhan 0021 de nhap so 21 .
123
 Hay nhap vao so thu hai co do dai  4 chu so,
 VD: nhan 0001 de nhap so 1 .
2
 nhap vao toan tu
 Nhan a de nhap phep cong
 Nhan b de nhap phep tru
 Nhan c de nhap phep nhan
 Nhan d de nhap phep chia
 Nhan e de nhap phep chia lay du

 Ban da nhap phep cong.

 Nhap f de hien thi ket qua
```

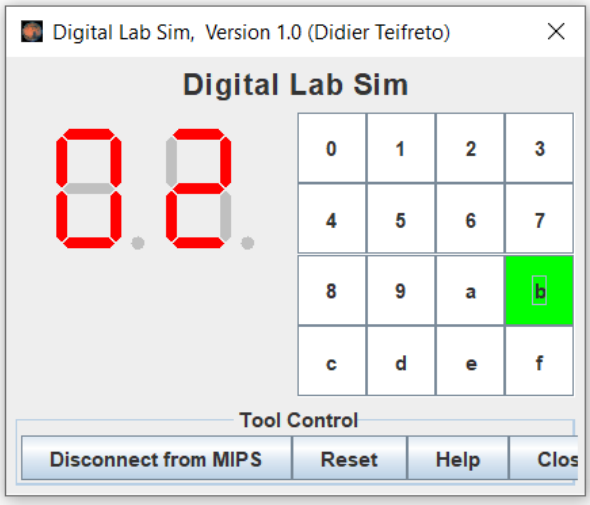


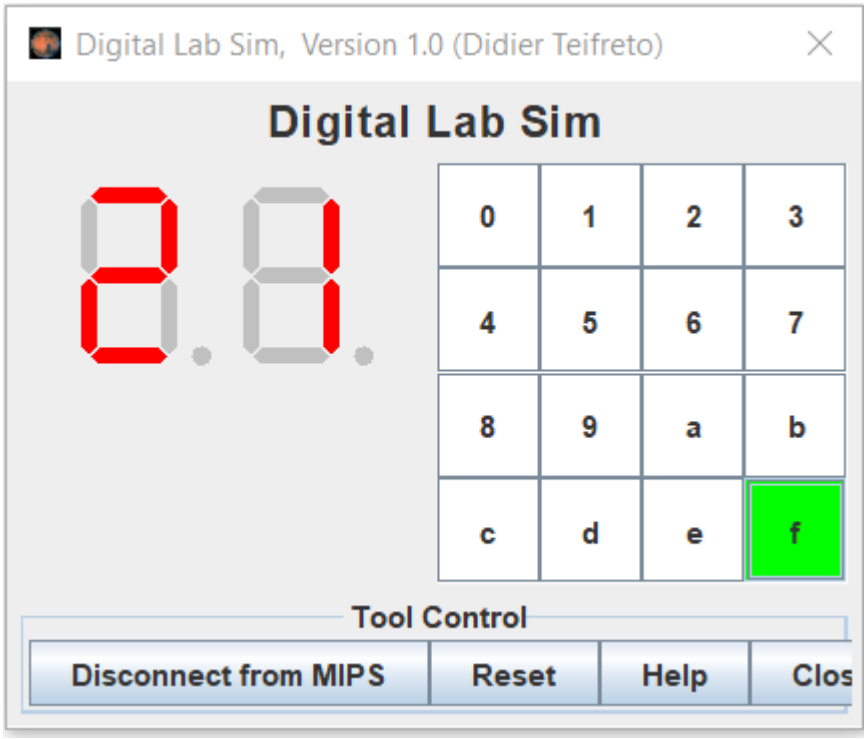

```
Chao mung den voi May tinh bo tui Dieu Ngo!

 Hay nhap vao so thu nhat co do dai  4 chu so,
 VD: nhan 0021 de nhap so 21 .
123
 Hay nhap vao so thu hai co do dai  4 chu so,
 VD: nhan 0001 de nhap so 1 .
2
 nhap vao toan tu
 Nhan a de nhap phep cong
 Nhan b de nhap phep tru
 Nhan c de nhap phep nhan
 Nhan d de nhap phep chia
 Nhan e de nhap phep chia lay du

 Ban da nhap phep cong.

 Nhap f de hien thi ket qua
 Ban da nhap phep cong.

 Nhap f de hien thi ket qua
 ket qua la :  125
```



- Phép trừ:

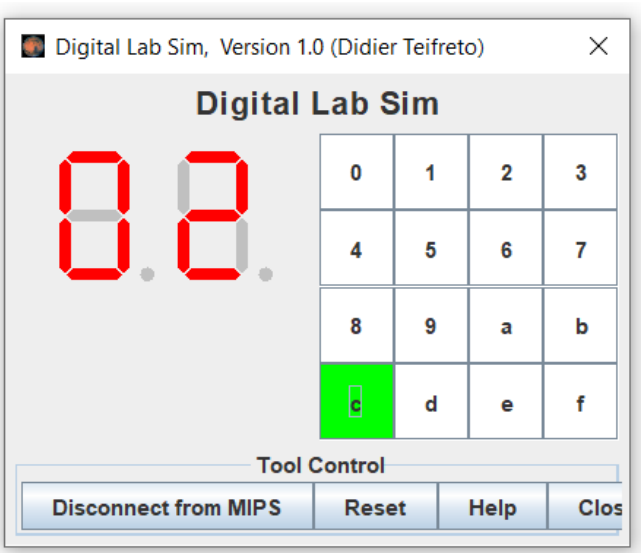
```
Chao mung den voi May tinh bo tui Dieu Ngo!

 Hay nhap vao so thu nhat co do dai  4 chu so,
 VD: nhan 0021 de nhap so 21 .
123
 Hay nhap vao so thu hai co do dai  4 chu so,
 VD: nhan 0001 de nhap so 1 .
2
 nhap vao toan tu
 Nhan a de nhap phep cong
 Nhan b de nhap phep tru
 Nhan c de nhap phep nhan
 Nhan d de nhap phep chia
 Nhan e de nhap phep chia lay du

 Ban da nhap phep tru.

 Nhap f de hien thi ket qua
```

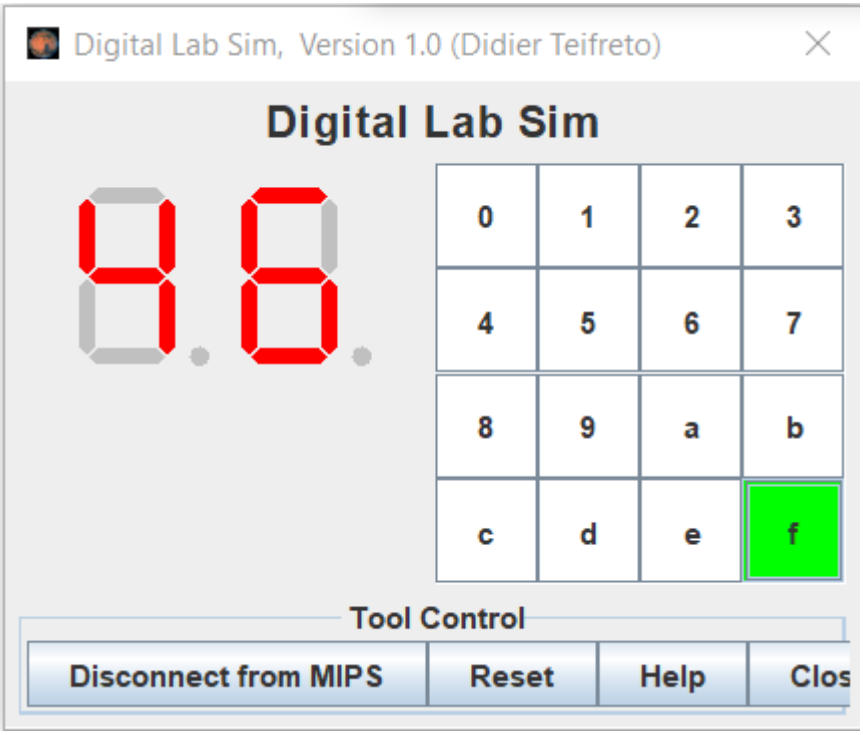




- Phép nhân:

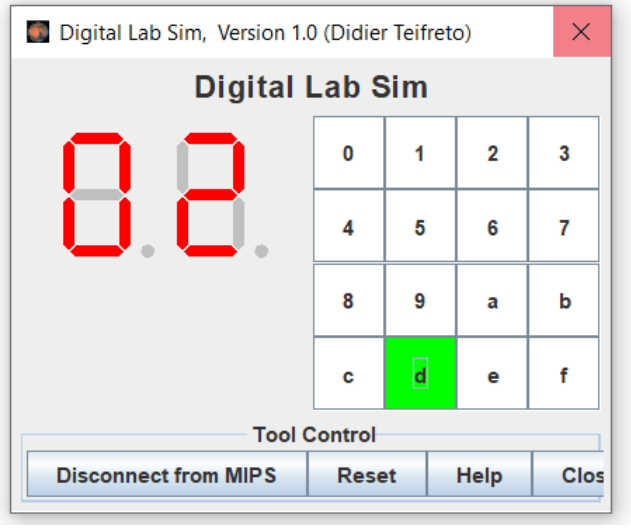
```
Chao mung den voi May tinh bo tui Dieu Ngo!

 Hay nhap vao so thu nhat co do dai  4 chu so,
 VD: nhan 0021 de nhap so 21 .
123
 Hay nhap vao so thu hai co do dai  4 chu so,
 VD: nhan 0001 de nhap so 1 .
2
 nhap vao toan tu
 Nhan a de nhap phep cong
 Nhan b de nhap phep tru
 Nhan c de nhap phep nhan
 Nhan d de nhap phep chia
 Nhan e de nhap phep chia lay du

 Ban da nhap phep nhan.

 Nhap f de hien thi ket qua
```

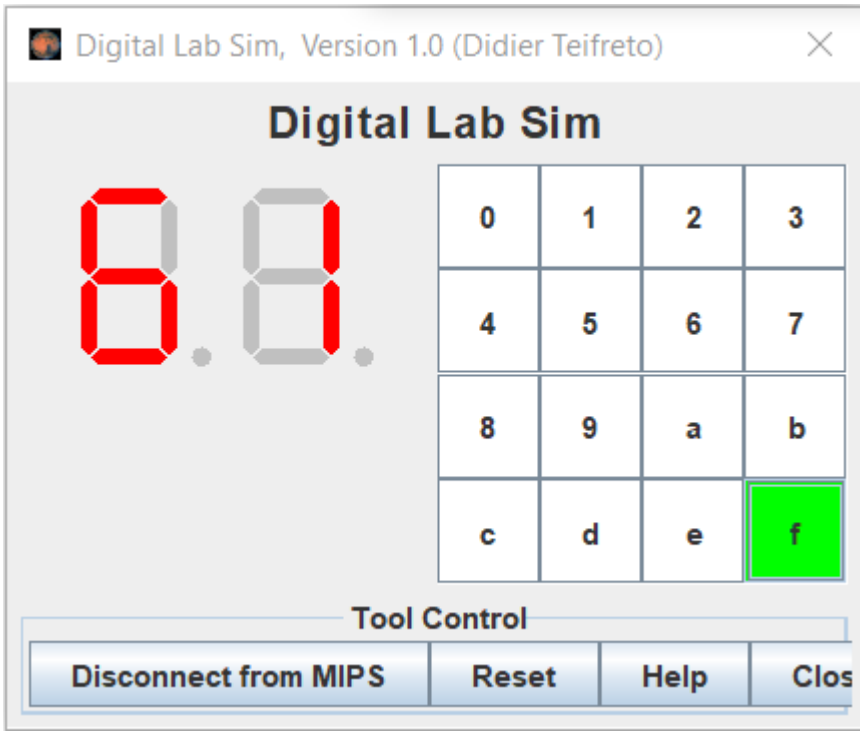




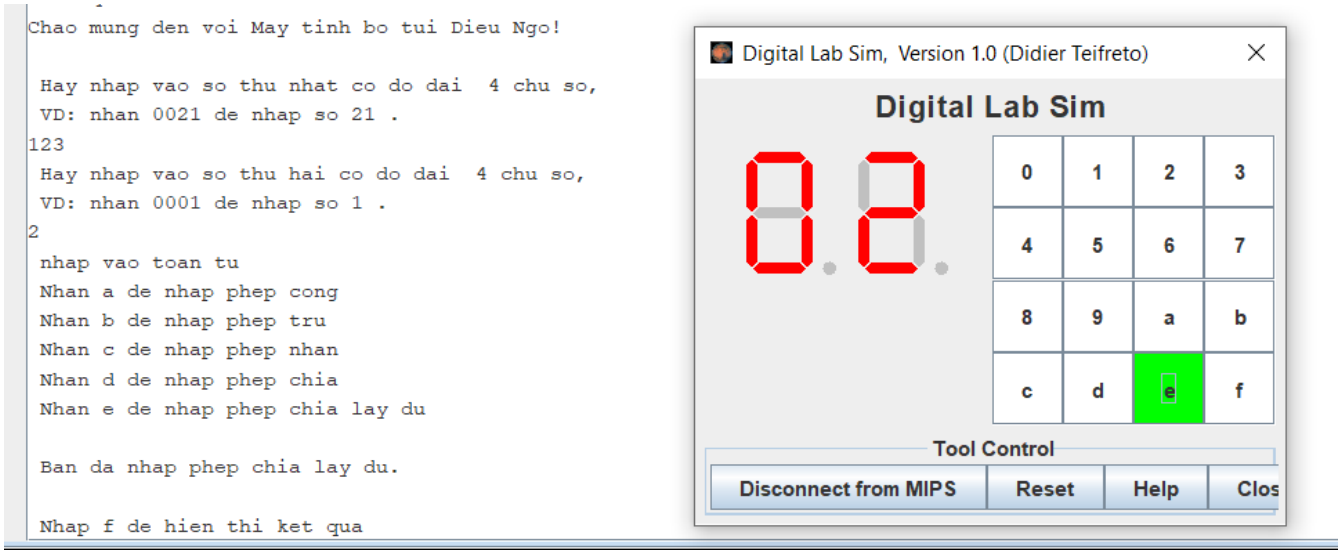
- Phép chia:

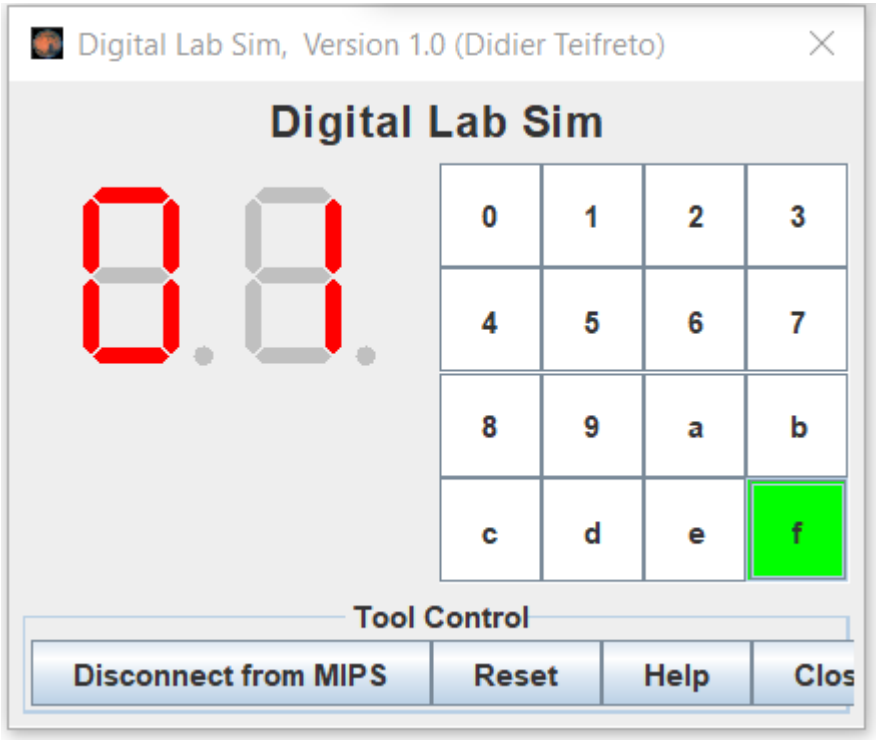
```
Chao mung den voi May tinh bo tui Dieu Ngo!

 Hay nhap vao so thu nhat co do dai  4 chu so,
 VD: nhan 0021 de nhap so 21 .
123
 Hay nhap vao so thu hai co do dai  4 chu so,
 VD: nhan 0001 de nhap so 1 .
2
 nhap vao toan tu
 Nhan a de nhap phep cong
 Nhan b de nhap phep tru
 Nhan c de nhap phep nhan
 Nhan d de nhap phep chia
 Nhan e de nhap phep chia lay du

 Ban da nhap phep chia.

 Nhap f de hien thi ket qua
```





- Phép chia lấy dư:

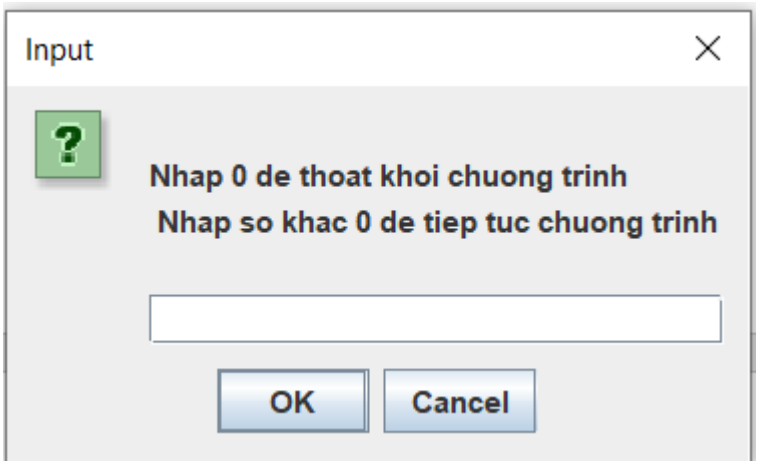




5.2. Xử lý ngoại lệ

5.2.1. Lặp đi lặp lại menu

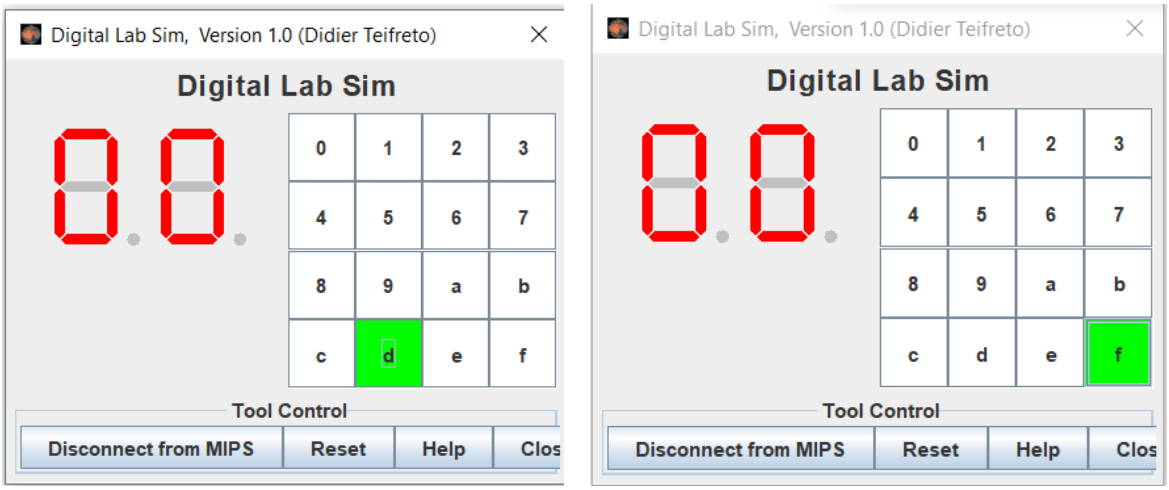
Sau khi thực hiện xong 1 phép toán, người dùng có thể quyết định ở lại thực hiện tiếp các phép toán khác hoặc thoát khỏi chương trình. Nhấn số 0 từ bàn phím để thoát khỏi chương trình hoặc nhấn số bất kì khác 0 để ở lại.



5.2.2. Phép chia cho 0

Giả sử 2 số nhập vào là 1 và 0.

```
 Hay nhap vao so thu nhat co do dai  4 chu so,
 VD: nhan 0021 de nhap so 21 .
1
 Hay nhap vao so thu hai co do dai  4 chu so,
 VD: nhan 0001 de nhap so 1 .
0
```



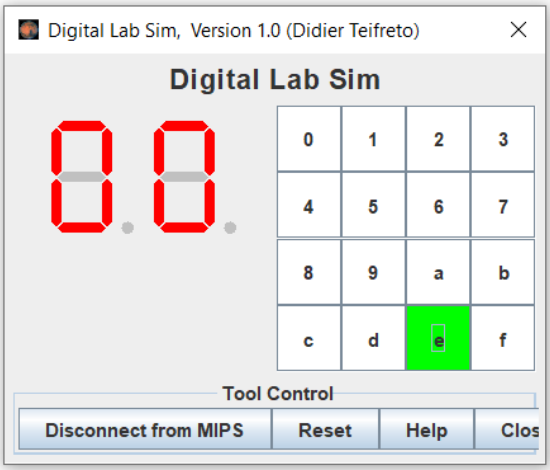

```
Nhap f de hien thi ket qua
Xay ra loi. Xin hay nhap lai toan tu
```

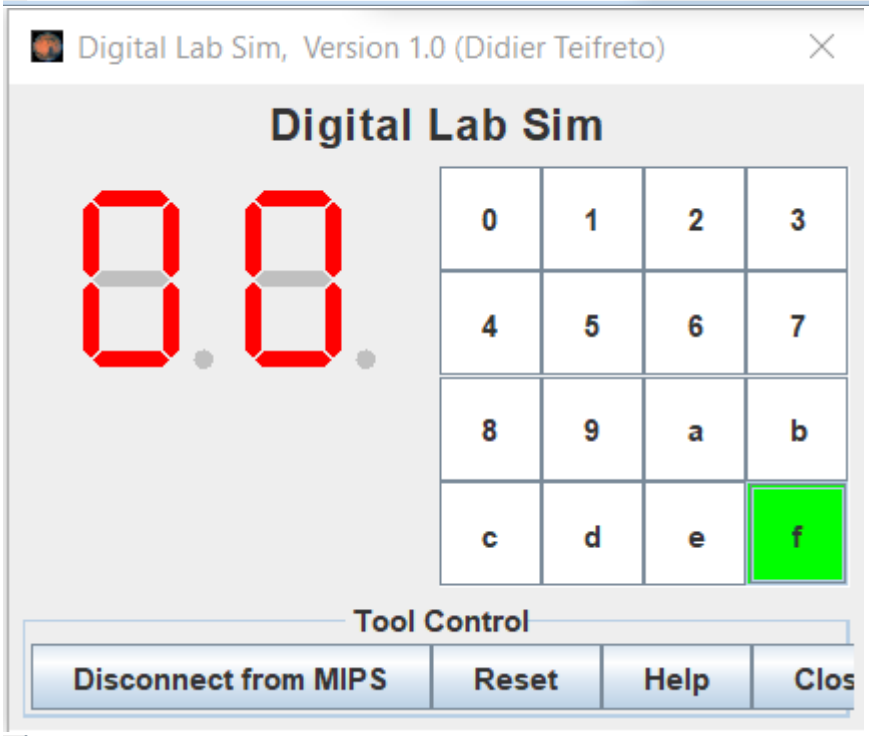
5.2.3. Phép chia lấy dư trong đó số chia là 0

Giả sử 2 số nhập vào là 1 và 0.

```
 Hay nhap vao so thu nhat co do dai  4 chu so,
 VD: nhan 0021 de nhap so 21 .
1
 Hay nhap vao so thu hai co do dai  4 chu so,
 VD: nhan 0001 de nhap so 1 .
0
```

```
Chao mung den voi May tinh bo tui Dieu Ngo!

 Hay nhap vao so thu nhat co do dai  4 chu so,
 VD: nhan 0021 de nhap so 21 .
1
 Hay nhap vao so thu hai co do dai  4 chu so,
 VD: nhan 0001 de nhap so 1 .
0
 nhap vao toan tu
 Nhan a de nhap phep cong
 Nhan b de nhap phep tru
 Nhan c de nhap phep nhan
 Nhan d de nhap phep chia
 Nhan e de nhap phep chia lay du

 Ban da nhap phep chia lay du.

 Nhap f de hien thi ket qua
```

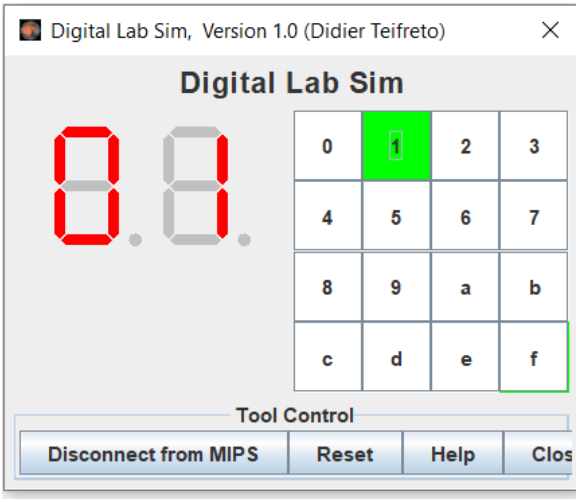


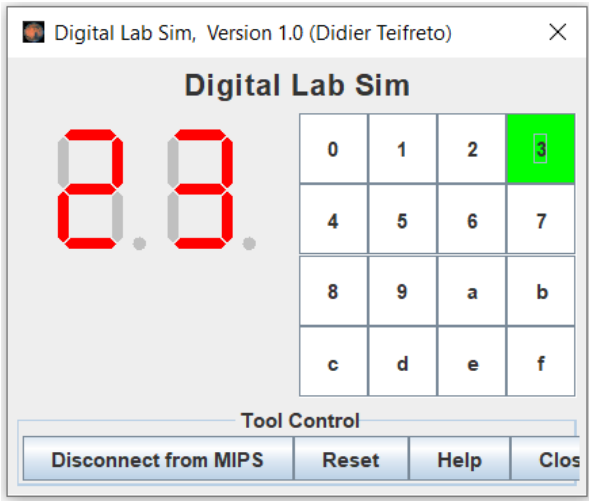



```
Nhap f de hien thi ket qua
Xay ra loi. Xin hay nhap lai toan tu
```

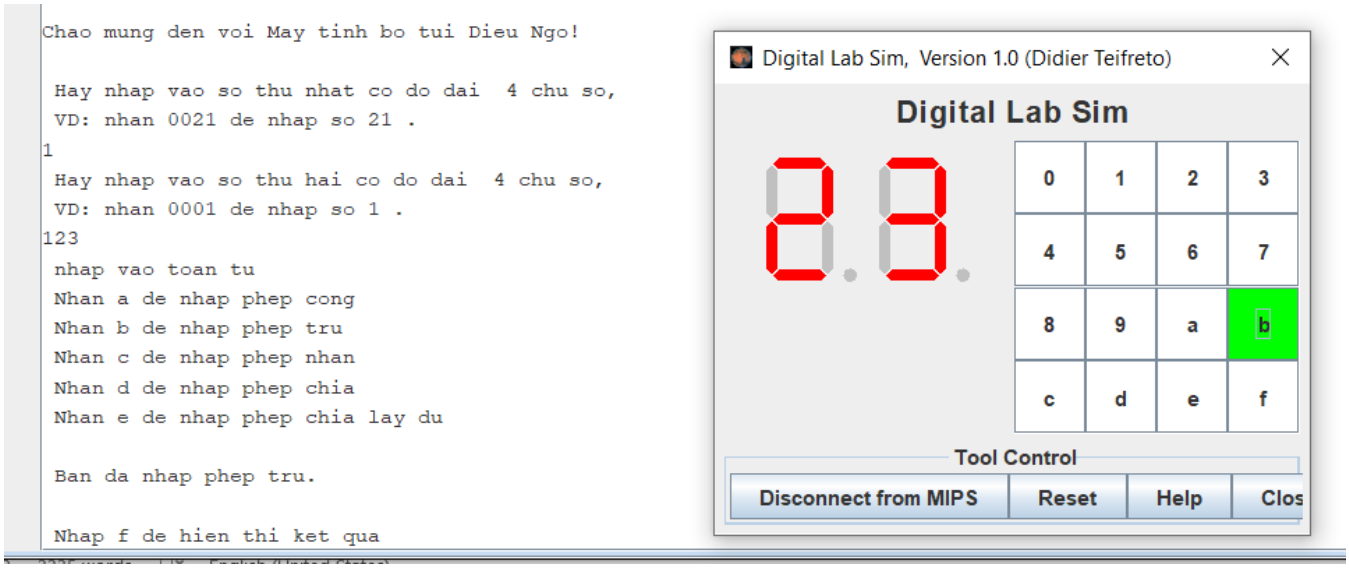
5.2.4. Phép trừ trong đó kết quả là 1 số nguyên âm

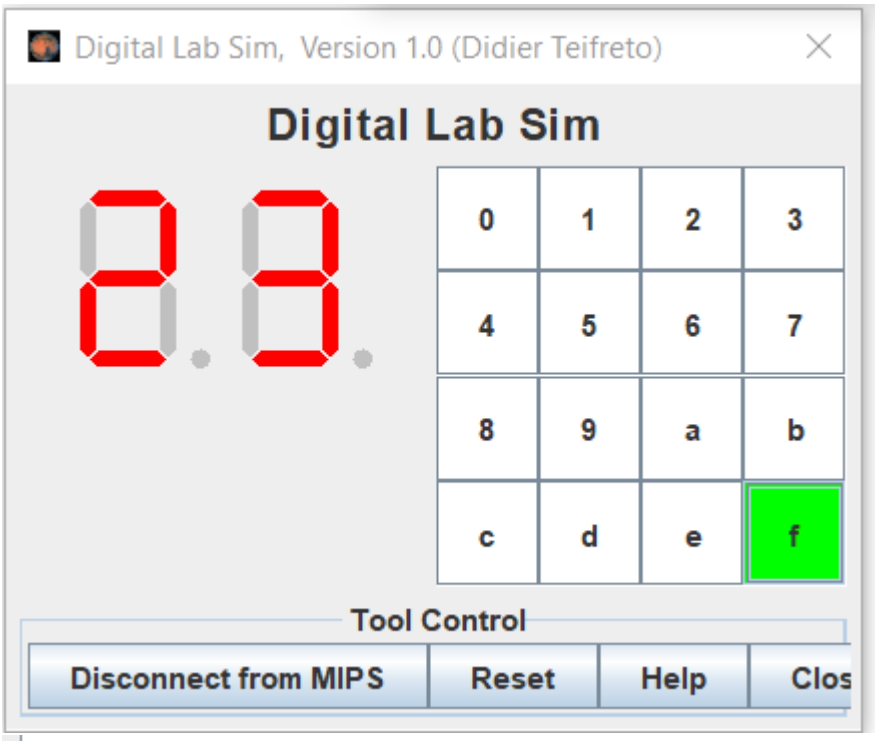
Giả sử 2 số nhập vào là 1 và 123.






```
Chao mung den voi May tinh bo tui Dieu Ngo!

 Hay nhap vao so thu nhat co do dai  4 chu so,
 VD: nhan 0021 de nhap so 21 .
1
 Hay nhap vao so thu hai co do dai  4 chu so,
 VD: nhan 0001 de nhap so 1 .
123
 nhap vao toan tu
 Nhan a de nhap phep cong
 Nhan b de nhap phep tru
 Nhan c de nhap phep nhan
 Nhan d de nhap phep chia
 Nhan e de nhap phep chia lay du
```






```
Nhap f de hien thi ket qua
ket qua la :  -122
Ket qua la so am, khong hien thi duoc
.
```

----------Hết----------