IT3280 –THỰC HÀNH KIẾN TRÚC MÁY TÍNH
Tuần 10:

Họ và tên	Phạm Quốc Cường
Mã số sinh viên	20225604

Assignment 3:

Code:

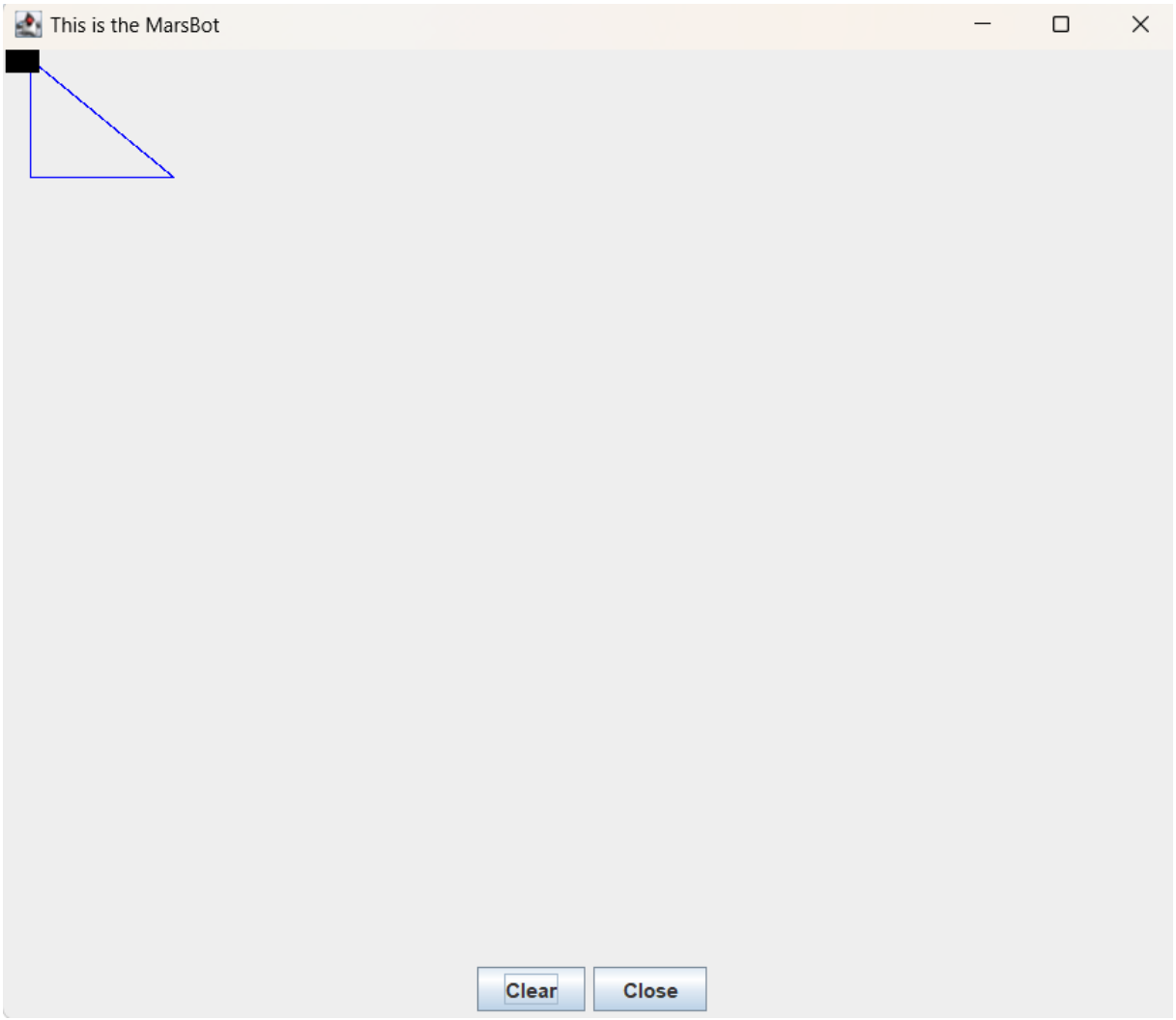
```
.eqv HEADING 0xffff8010          # integer: an angle between 0 and 359
                                 # 0 : North (up)
                                 # 90: East (right)
                                 # 180: South (down)
                                 # 270: West (left)
.eqv MOVING 0xffff8050           # boolean: whether or not to move
.eqv LEAVETRACK 0xffff8020       # boolean: (0 or non-0)
                                 # whether or not to leave a track
.eqv WHEREX 0xffff8030           # integer: current x-location of Marsbot
.eqv WHEREY 0xffff8040           # integer: current y-location of marsbot
.text
#chuyển bot về nơi nhìn rõ được tam giác
main:   addi $a0, $0, 130 # Marsbot rotates 180 * running and start running
        jal ROTATE
        nop
        jal GO
        nop
sleep:  addi $v0, $0, 32 # keep running by sleeping in 1000ms
        li $a0, 1000
        syscall
#---------------------------------
        jal STOP
goline1: addi $a0, $0, 180 # Marsbot rotates 90 * running and start running
        jal ROTATE
        nop
        jal TRACK
```

```
        nop
        jal GO
        nop
sleep1: addi $v0, $0, 32 # keep running by sleeping in 3000ms
        li $a0, 3000
        syscall
#-------------------------------------
        jal STOP
goline2: addi $a0, $0, 90 # Marsbot rotates 90 * running and start running
jal ROTATE
        nop
        jal UNTRACK
        nop
        jal TRACK
        nop
        jal GO
        nop
sleep2: addi $v0, $0, 32 # keep running by sleeping in 4000ms
        li $a0, 4000
        syscall
#-------------------------------------
        jal STOP
goline3: addi $a0, $0, 310 # Marsbot rotates 90 * running and start running
        jal ROTATE
        nop
        jal UNTRACK
        nop
        jal TRACK
        nop
        jal GO
        nop
sleep3: addi $v0, $0, 32 # keep running by sleeping in 6100ms
```

```
        li $a0, 6100
        syscall
#--------------------------------
end_main: jal STOP
        nop
        j end
#------------------------------------------------------
# GO procedure, to start running
# param[in] none
#------------------------------------------------------
GO: li $at, MOVING # change MOVING port
        addi $k0, $zero,1 # to logic 1,
        sb $k0, 0($at) # to start running
        nop
        jr $ra
        nop
#------------------------------------------------------
# STOP procedure, to stop running
# param[in] none
#------------------------------------------------------
STOP: li $at, MOVING # change MOVING port to 0
        sb $zero, 0($at) # to stop
        nop
        jr $ra
        nop
#------------------------------------------------------
# TRACK procedure, to start drawing line
# param[in] none
#------------------------------------------------------
TRACK: li $at, LEAVETRACK # change LEAVETRACK port
        addi $k0, $zero,1 # to logic 1,
        sb $k0, 0($at) # to start tracking
```

```
        nop
        jr $ra
        nop
#-----------------------------------------------------------
# UNTRACK procedure, to stop drawing line
# param[in] none
#-----------------------------------------------------------
UNTRACK:li $at, LEAVETRACK # change LEAVETRACK port to 0
        sb $zero, 0($at) # to stop drawing tail
        nop
        jr $ra
        nop
#-----------------------------------------------------------
# ROTATE procedure, to rotate the robot
# param[in] $a0, An angle between 0 and 359
# 0 : North (up)
# 90: East (right)
# 180: South (down)
# 270: West (left)
#-----------------------------------------------------------
ROTATE: li $at, HEADING # change HEADING port
        sw $a0, 0($at) # to rotate robot
        nop
        jr $ra
        nop
end:
```

Result:



Assignment 4:

Code:

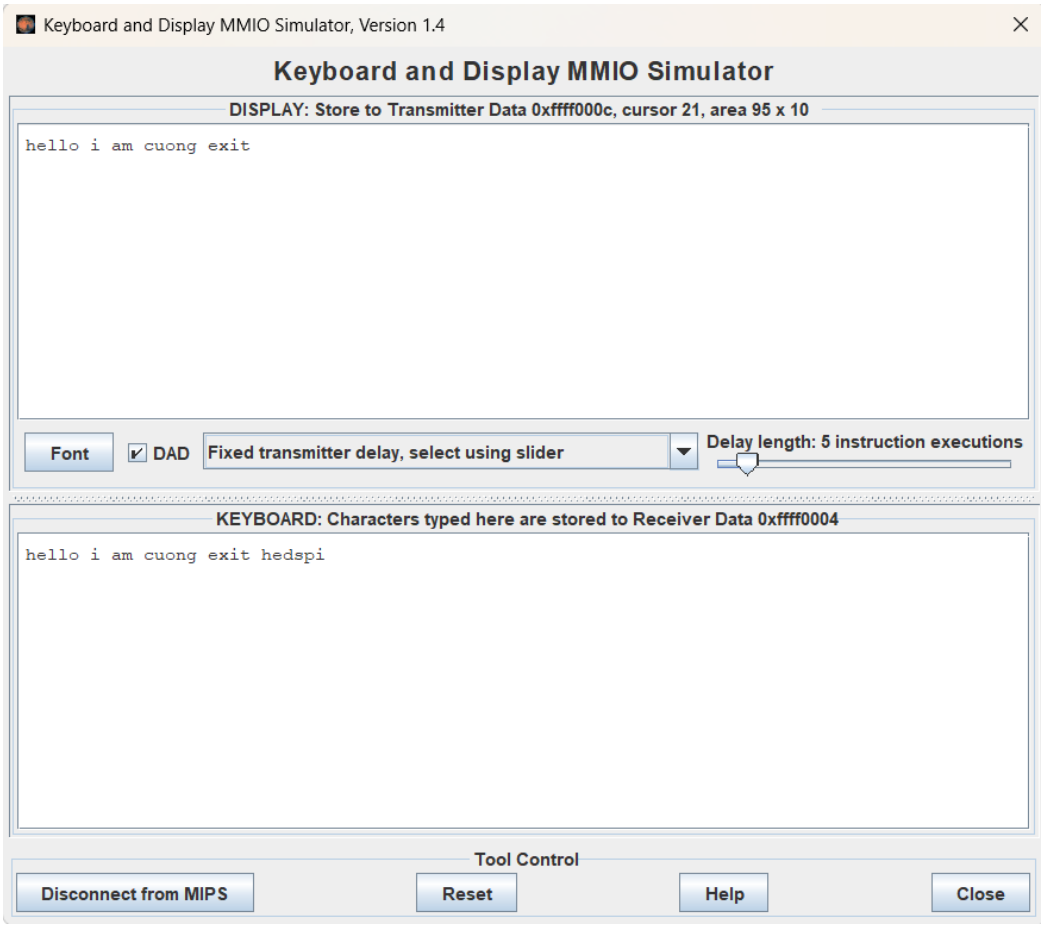
```
.eqv KEY_CODE 0xFFFF0004 # ASCII code from keyboard, 1 byte
.eqv KEY_READY 0xFFFF0000 # =1 if has a new keycode ?
# Auto clear after lw
.eqv DISPLAY_CODE 0xFFFF000C # ASCII code to show, 1 byte
.eqv DISPLAY_READY 0xFFFF0008 # =1 if the display has already to do
# Auto clear after sw
.text
        li $k0, KEY_CODE
        li $k1, KEY_READY
        li $s0, DISPLAY_CODE
```

```
        li $s1, DISPLAY_READY
        li $t6, 0
        li $t5, 4
loop: nop
WaitForKey: lw $t1, 0($k1) # $t1 = [$k1] = KEY_READY
        nop
        beq $t1, $zero, WaitForKey # if $t1 == 0 then Polling
        nop
#----------------------------------------------------
ReadKey: lw $t0, 0($k0) # $t0 = [$k0] = KEY_CODE
        nop
#----------------------------------------------------
WaitForDis: lw $t2, 0($s1) # $t2 = [$s1] = DISPLAY_READY
        nop
        beq $t2, $zero, WaitForDis # if $t2 == 0 then Polling
        nop
#----------------------------------------------------
ShowKey: sw $t0, 0($s0) # show key
        nop
#----------------------------------------------------
 #check exit
        beq $t0, 'e', continue_fake
        beq $t0, 'x', checkx
        beq $t0, 'i', checki
        beq $t0, 't', checkt
        j continue
continue_fake2: addi $t6, $t6, 1
        j continue
        continue_fake: addi $t6, $zero, 1
        continue: beq $t6, $t5, exit
        j loop
        nop
```

```
checkx:
        li $t3, 1
        beq $t6, $t3, continue_fake2
        j continue
checki:
        li $t3, 2
        beq $t6, $t3 continue_fake2
        j continue
checkt:
        li $t3, 3
        beq $t6, $t3, continue_fake2
        j continue
exit:
```

Result:

Chương trình chạy bình thường và đưa kết quả input từ bàn phím vào mục DISPLAY cho đến khi gặp exit chương trình tự động ngắt và không còn nhận input nữa.



Explain:

- Kiểm tra các kí tự nhập vào, mỗi lần đọc được kí tự ‘e’ thì bộ đém $t6 tăng thành 1 và tương tự sẽ tăng thêm 1 nếu gặp thêm các kí tự ‘x’, ‘i’, ‘t’.

- Nếu bộ đếm đạt đến 4, chỉ ra rằng chuỗi "exit" đã được nhập chính xác, chương trình nhảy tới nhãn exit. Nếu không nó tiếp tục đợi phím bấm tiếp theo.

```
#check exit
        beq $t0, 'e', continue_fake
        beq $t0, 'x', checkx
        beq $t0, 'i', checki
        beq $t0, 't', checkt
        j continue
continue_fake2: addi $t6, $t6, 1
        j continue
continue_fake: addi $t6, $zero, 1
continue: beq $t6, $t5, exit
        j loop
        nop
checkx:
        li $t3, 1
        beq $t6, $t3, continue_fake2
        j continue
checki:
        li $t3, 2
        beq $t6, $t3 continue_fake2
        j continue
checkt:
        li $t3, 3
        beq $t6, $t3, continue_fake2
        j continue

exit:
```