

BÁO CÁO THỰC HÀNH TUẦN 11

Lương Thị Tâm-20194663

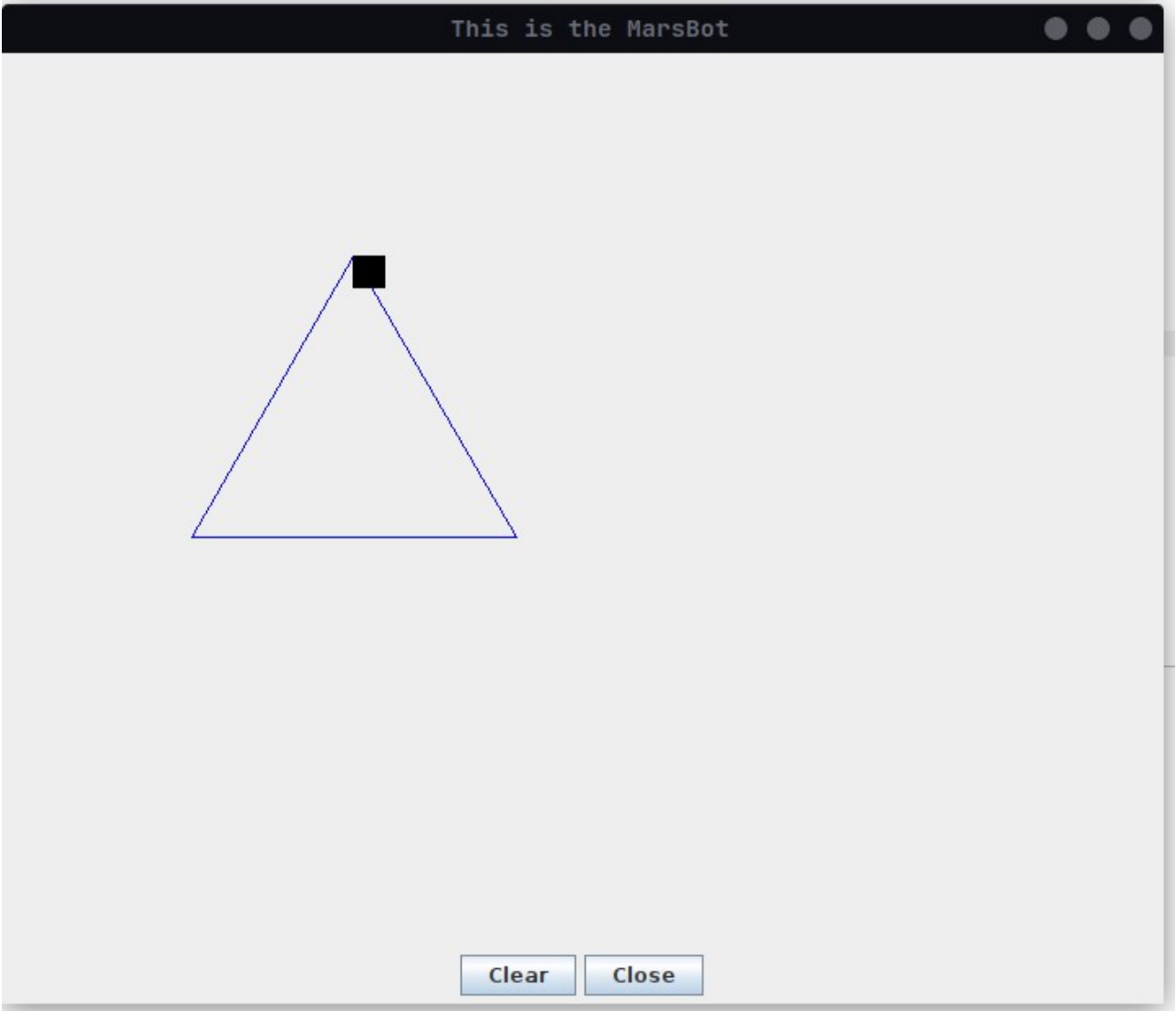
Bài 1:

- Vẽ tam giác cân

Mã nguồn:

```
1 .eqv HEADING 0xffff8010 # Integer: An angle between 0 and 359
2 # 0 : North (up)
3 # 90: East (right)
4 # 180: South (down)
5 # 270: West (left)
6 .eqv MOVING 0xffff8050 # Boolean: whether or not to move
7 .eqv LEAVETRACK 0xffff8020 # Boolean (0 or non-0):
8 # whether or not to leave a track
9 .eqv WHEREX 0xffff8030 # Integer: Current x-location of MarsBot
10 .eqv WHEREY 0xffff8040 # Integer: Current y-location of MarsBot
11
12 .text
13 main: addi $a0, $zero, 120# Marsbot rotates 120* and start running
14 jal ROTATE
15 jal GO
16 sleep1: addi $v0,$zero,32 # Keep running by sleeping in10000 ms
17 li $a0,10000
18 syscall
19 jal TRACK# and draw new track line
20 go1: addi $a0, $zero, 150# Marsbot rotates 150*
21 jal ROTATE
22 sleep2: addi $v0,$zero,32 # Keep running by sleeping in 8000 ms
23 li $a0,8000
24 syscall
25 jal UNTRACK # keep old track
26 jal TRACK # and draw new track line
27 go2: addi $a0, $zero, 270# Marsbotrotates 270*
28 jal ROTATE
29 sleep3: addi $v0,$zero,32 # Keep running by sleeping in 1000 ms
30 li $a0,8000
31 syscall
32 jal UNTRACK # keep old track
33 jal TRACK # and draw new track line
34 go3: addi $a0, $zero, 30 # Marsbot rotates 30*
35 jal ROTATE
36 sleep4: addi $v0,$zero,32 # Keep running by sleeping in 2000 ms
37 li $a0,8000
38 syscall
39 jal UNTRACK # keep old track
40
41 #jal TRACK # and draw new track line
42 jal STOP
43 li $v0, 10
44 syscall
45 end_main:
46 #-----
47 # GO procedure, to start running
48 # param[in] none
49 #-----
50 GO: li $at, MOVING # change MOVING port
51 addi $k0, $zero,1 # to logic 1,
52 sb $k0, 0($at) # to start running
53 jr $ra
54 #-----
55 # STOP procedure, to stop running
56 # param[in] none
57 #-----
58 STOP: li $at, MOVING # change MOVING port to 0
59 sb $zero, 0($at) # to stop
60 jr $ra
61 #-----
62 # TRACK procedure, to start drawing line
63 # param[in] none
64 #-----
65 TRACK: li $at, LEAVETRACK # change LEAVETRACK port
66 addi $k0, $zero,1 # to logic 1,
67 sb $k0, 0($at) # to start tracking
68 jr $ra
69 #-----
70 # UNTRACK procedure, to stop drawing line\
71 # param[in] none
72 #-----
73 UNTRACK: li $at, LEAVETRACK # change LEAVETRACK port to 0
74 sb $zero, 0($at) # to stop drawing tail
75 jr $ra
76 #-----
77 # ROTATE procedure, to rotate the robot
78 # param[in] $a0, An angle between 0 and 359
79 # 0 : North (up)
80 # 90: East (right)
81 # 180: South (down)
82 # 270: West (left)
83 #-----
84 ROTATE: li $at, HEADING # change HEADING port
85 sw $a0, 0($at) # to rotate robot
86 jr $ra
```

Kết quả chạy:

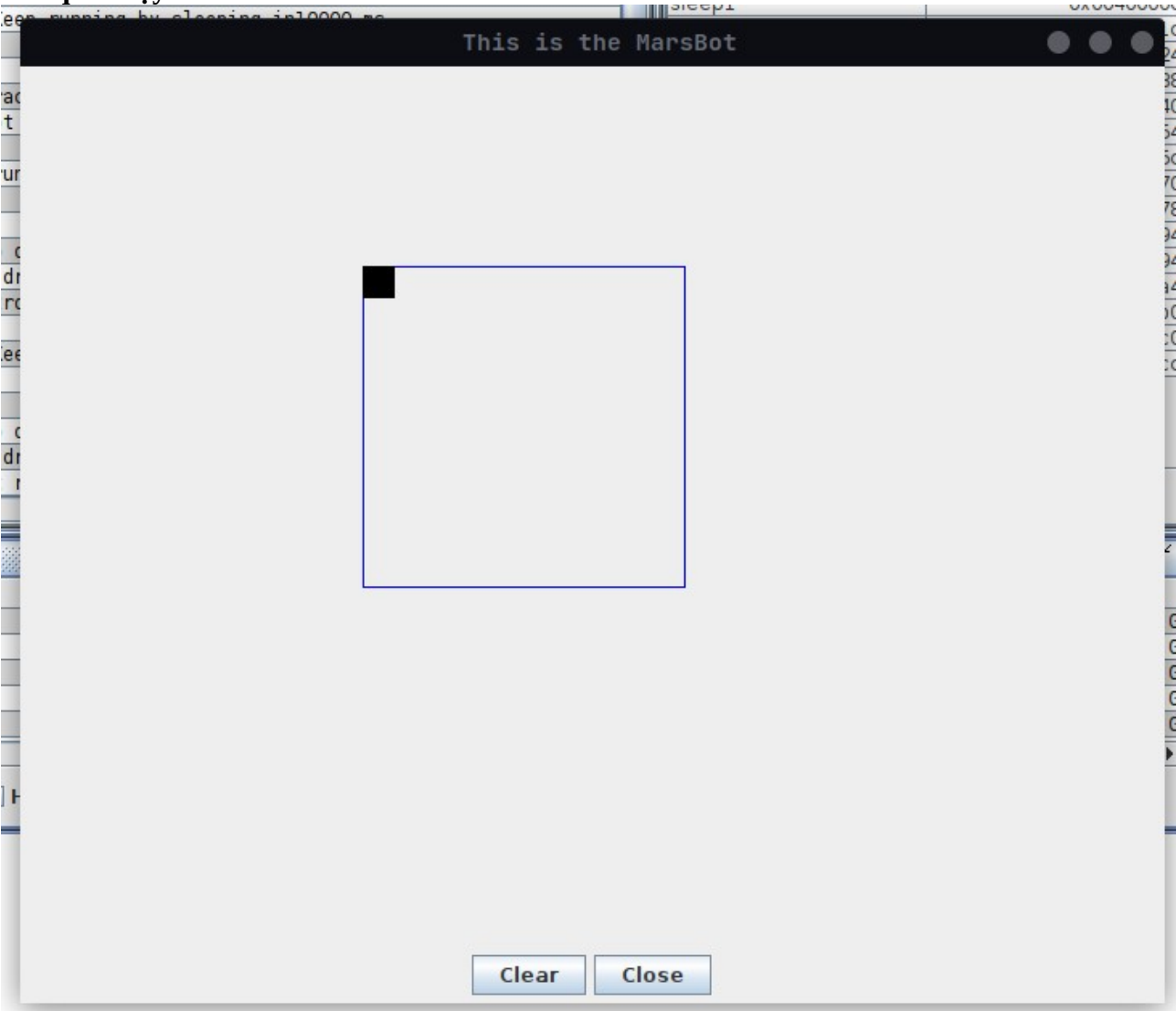


- **Vẽ hình vuông**

Mã nguồn:

```
12 .text
13 main: #jal TRACK# draw trackline
14     addi $a0, $zero, 120# Marsbot rotates 120* and start running
15     jal  ROTATE
16     jal  GO
17     sleep1: addi $v0,$zero,32    # Keep running by sleeping in 10000 ms
18             li  $a0, 10000
19             syscall
20             #jal  UNTRACK# keep old track
21             jal  TRACK# and draw new track line
22     go1: addi $a0, $zero, 90# Marsbot rotates 90*
23         jal  ROTATE
24     sleep2: addi $v0,$zero,32    # Keep running by sleeping in 8000 ms
25             li  $a0, 8000
26             syscall
27             jal  UNTRACK          # keep old track
28             jal  TRACK            # and draw new track line
29     go2: addi $a0, $zero, 180# Marsbot rotates 180*
30         jal  ROTATE
31     sleep3: addi $v0,$zero,32    # Keep running by sleeping in 8000 ms
32             li  $a0, 8000
33             syscall
34             jal  UNTRACK          # keep old track
35             jal  TRACK            # and draw new track line
36     go3: addi $a0, $zero, 270 # Marsbot rotates 270*
37         jal  ROTATE
38     sleep4: addi $v0,$zero,32    # Keep running by sleeping in 8000 ms
39             li  $a0, 8000
40             syscall
41             jal  UNTRACK          # keep old track
42             jal  TRACK            # and draw new track line
43     go4: addi $a0, $zero, 0 # Marsbot rotates 0*
44         jal  ROTATE
45     sleep5: addi $v0,$zero,32    # Keep running by sleeping in 8000 ms
46             li  $a0, 8000
47             syscall
48             jal  UNTRACK          # keep old track
49             jal  STOP
50             li  $v0, 10
51             syscall
52     end_main:
53     "
```

Kết quả chạy:



- Vẽ ngôi sao năm cánh

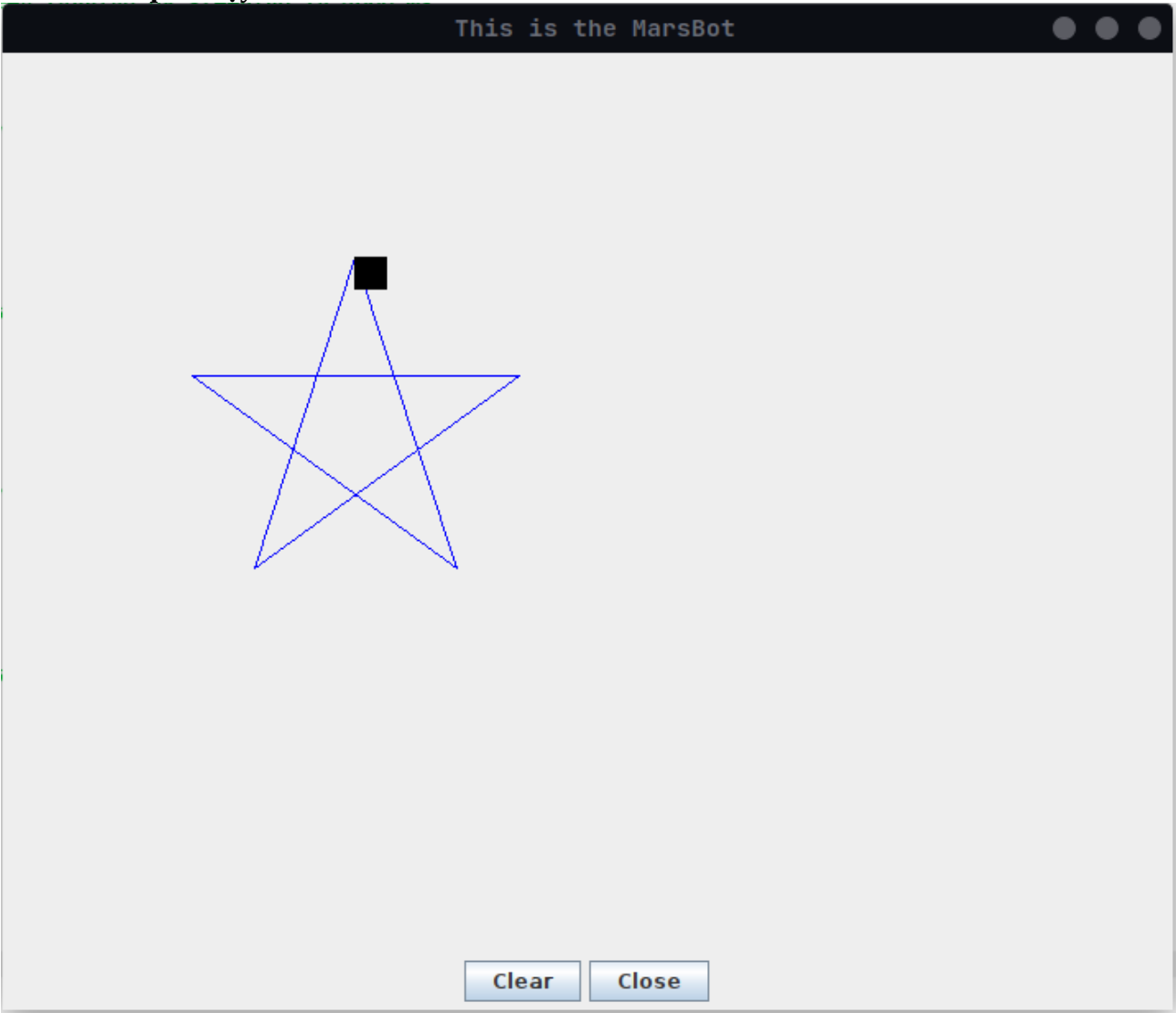
Mã nguồn:

```

12 .text
13 main: addi $a0, $zero, 120# Marsbot rotates 120* and start running
14     jal ROTATE
15     jal GO
16 sleep1: addi $v0, $zero, 32 # Keep running by sleeping in 10000 ms
17     li $a0, 10000
18     syscall
19     jal TRACK# and draw new track line
20 go1: addi $a0, $zero, 162# Marsbot rotates 162*
21     jal ROTATE
22 sleep2: addi $v0, $zero, 32 # Keep running by sleeping in 8000 ms
23     li $a0, 8000
24     syscall
25     jal UNTRACK # keep old track
26     jal TRACK # and draw new track line
27 go2: addi $a0, $zero, 306# Marsbot rotates 306*
28     jal ROTATE
29 sleep3: addi $v0, $zero, 32 # Keep running by sleeping in 8000 ms
30     li $a0, 8000
31     syscall
32     jal UNTRACK # keep old track
33     jal TRACK # and draw new track line
34 go3: addi $a0, $zero, 90 # Marsbot rotates 90*
35     jal ROTATE
36 sleep4: addi $v0, $zero, 32 # Keep running by sleeping in 8000 ms
37     li $a0, 8000
38     syscall
39     jal UNTRACK # keep old track
40     jal TRACK # and draw new track line
41 go4: addi $a0, $zero, 234 # Marsbot rotates 234*
42     jal ROTATE
43 sleep5: addi $v0, $zero, 32 # Keep running by sleeping in 8000 ms
44     li $a0, 8000
45     syscall
46     jal UNTRACK # keep old track
47     jal TRACK # and draw new track line
48 go5: addi $a0, $zero, 18 # Marsbot rotates 18*
49     jal ROTATE
50 sleep6: addi $v0, $zero, 32 # Keep running by sleeping in 8000 ms
51     li $a0, 8000
52     syscall
53     jal UNTRACK # keep old track
54     jal STOP
55     li $v0, 10
56     syscall
57 end_main:

```

Kết quả chạy:

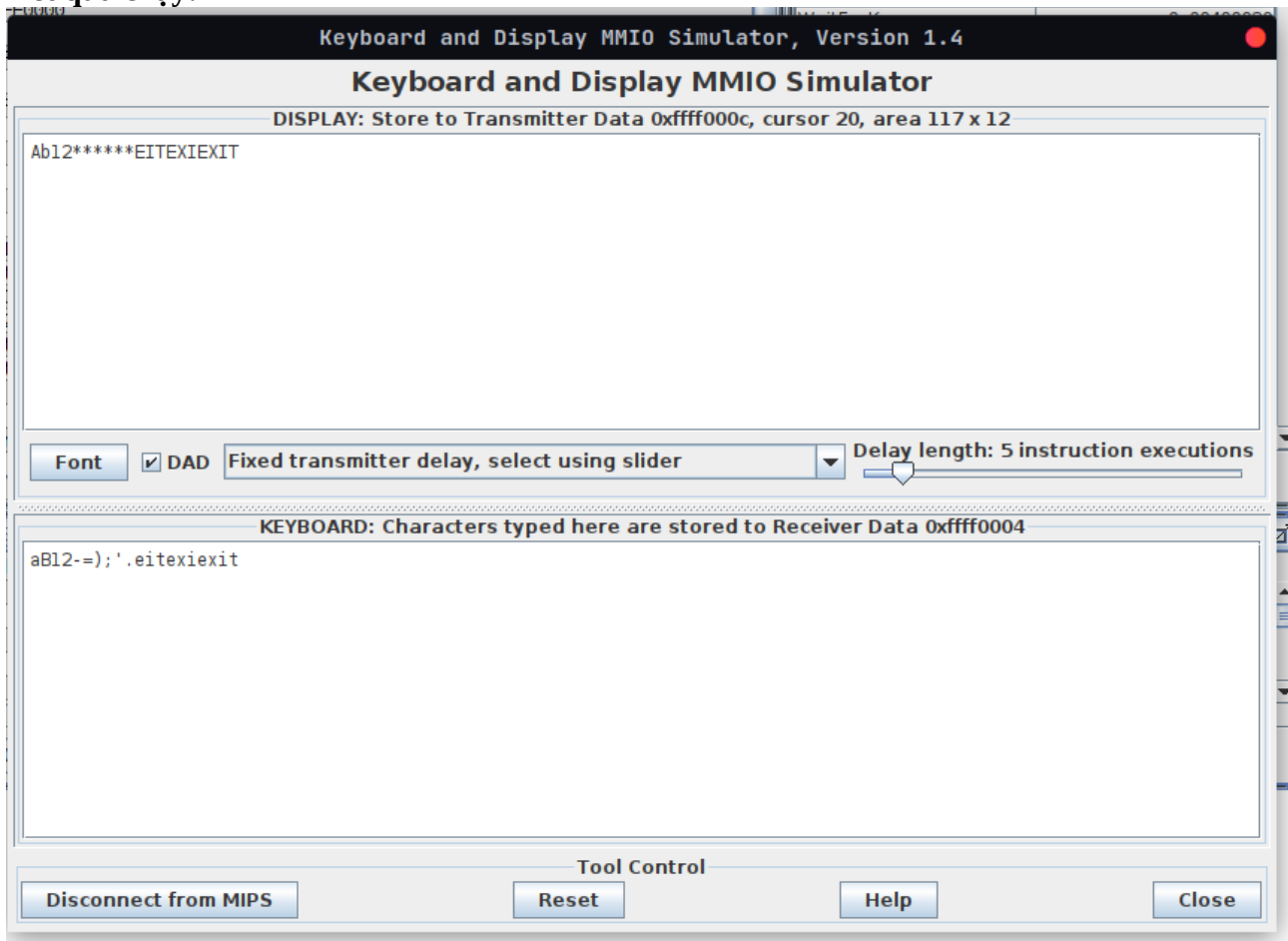


Bài 2:

Mã nguồn:

```
1 .eqv KEY_CODE    0xFFFF0004    # ASCII code from key
2 .eqv KEY_READY   0xFFFF0000    # =1 if has a new keycode ?
3                                     #Auto clear after lboard, 1 byte
4 .eqv DISPLAY_CODE 0xFFFF000C    # ASCII code to show, 1 byte
5 .eqv DISPLAY_READY 0xFFFF0008    # =1 if the display has already to do
6                                     # Auto clear after sw
7
8 .data
9 exit: .asciiz "exit"
10
11 .text
12     li $k0, KEY_CODE
13     li $k1, KEY_READY
14     li $s0, DISPLAY_CODE
15     li $s1, DISPLAY_READY
16     li $t3, 0
17     la $s3, exit
18
19 loop:    nop
20
21 WaitForKey: lw $t1, 0($k1)        # $t1 = [$k1] = KEY_READY
22             beq $t1, $zero, WaitForKey # if $t1 == 0 then Polling
23
24 ReadKey:   lw $t0, 0($k0)        # $t0 = [$k0] = KEY_CODE
25
26 WaitForDis: lw $t2, 0($s1)        # $t2 = [$s1] = DISPLAY_READY
27             beq $t2, $zero, WaitForDis # if $t2 == 0 then Polling
28             bgt $t0, 47, check1
29             j Encrypt
30             nop
31
32 check1:    bgt $t0, 57, check2
33             addi $t0, $t0, 0
34             j ShowKey
35             nop
36
37 check2:    bgt $t0, 64, check3
38             j Encrypt
39             nop
40
41 check3:    bgt $t0, 90, check4
42             addi $t0, $t0, 32
43             j ShowKey
44             nop
45
46 check4:    bgt $t0, 96, check5
47             j Encrypt
48             nop
49
50 check5:    bgt $t0, 122, Encrypt
51             add $s4, $t3, $s3
52             lb $t4, 0($s4)
53             beq $t4, $t0, next
54             beq $t0, 101, next2
55             addi $t3, $t3, 0
56             addi $t0, $t0, -32
57             j ShowKey
58             nop
59
60 next:      addi $t3, $t3, 1
61             addi $t0, $t0, -32
62             j ShowKey
63             nop
64
65 next2:     addi $t3, $t0, 1
66             addi $t0, $t0, -32
67             j ShowKey
68             nop
69
70 Encrypt:   addi $t0, $t0, 42 # change input key
71
72 ShowKey:   sw $t0, 0($s0) # show key
73             nop
74             nop
75             beq $t3, 4, end_main
76             j loop
77
78 end_main:
```

Kết quả chạy:



Bài 3:

Mã nguồn:

```
1 .eqv KEY_CODE 0xFFFF0004 # ASCII code from keyboard, 1 byte
2 .eqv KEY_READY 0xFFFF0000 # =1 if has a new keycode ?
3 # Auto clear after lw
4 .eqv DISPLAY_CODE 0xFFFF000C # ASCII code to show, 1 byte
5 .eqv DISPLAY_READY 0xFFFF0008 # =1 if the display has already to do
6 # Auto clear after sw
7 .eqv HEADING 0xffff8010 # Integer: An angle between 0 and 359
8 .eqv MOVING 0xffff8050 # Boolean: whether or not to move
9 .eqv LEAVETRACK 0xffff8020 # Boolean (0 or non-0):
10 # whether or not to leave a track
11 .eqv WHEREX 0xffff8030 # Integer: Current x-location of MarsBot
12 .eqv WHEREY 0xffff8040 # Integer: Current y-location of MarsBot
13 .data
14 .text
15     li $k0, KEY_CODE
16     li $k1, KEY_READY
17     li $s0, DISPLAY_CODE
18     li $s1, DISPLAY_READY
19
20     addi $a0, $zero, 135          #Marsbot rotates 135* and start
21     jal ROTATE
22     jal TRACK                    #Start draw
23     jal GO
24 loop: nop
25 WaitForKey: lw $t1, 0($k1) # $t1 = [$k1] = KEY_READY
26     beq $t1, $zero, WaitForKey # if $t1 == 0 then Polling
27 ReadKey: lw $t0, 0($k0) # $t0 = [$k0] = KEY_CODE
28     addi $v0, $0, 1
29     addi $a0, $t0, 0
30     syscall
31 Control:
32     jal UNTRACK                 # keep old track
33     li $a0, MOVING
34     lb $a0, 0($a0)
35 SPACE: bne $t0, ' ', UP
36     bne $a0, $0, stop
37 go:    jal GO
38     j continue
39 stop:  jal STOP
```

```

40      j    continue
41  UP:   bne $t0, 'w', DOWN
42      addi $a0, $zero, 0
43      j    rotate
44  DOWN: bne $t0, 's', LEFT
45      addi $a0, $zero, 180
46      j    rotate
47  LEFT: bne $t0, 'a', RIGHT
48      addi $a0, $zero, 270
49      j    rotate
50  RIGHT: bne $t0, 'd', continue
51      addi $a0, $zero, 90
52      j    rotate
53  rotate:
54      jal ROTATE
55
56      j    continue
57
58  continue: jal TRACK      # and draw new track line
59      j loop
60  end_main:
61      addi $v0, $0, 10
62      syscall
63
64  GO:   li $at, MOVING # change MOVING port
65      addi $a0, $zero, 1 # to logic 1,
66      sb $a0, 0($at) # to start running
67      jr $ra
68  STOP: li $at, MOVING # change MOVING port to 0
69      sb $zero, 0($at) # to stop
70      jr $ra
71  TRACK: li $at, LEAVETRACK # change LEAVETRACK port
72      addi $a0, $zero, 1 # to logic 1,
73      sb $a0, 0($at) # to start tracking
74      jr $ra
75  UNTRACK: li $at, LEAVETRACK # change LEAVETRACK port to 0
76
77      sb $zero, 0($at) # to stop drawing tail
78      jr $ra
79  ROTATE: li $at, HEADING # change HEADING port
80      sw $a0, 0($at) # to rotate robot
81      jr $ra

```

The screenshot shows a Linux desktop environment. On the left, there is a terminal window with a blue border and a black cursor. The main window is a MIPS assembly editor titled "File Edit Run Settings Tools Help". It contains a table of assembly instructions:

Bkpt	Address	Code	Basic	Source
	0x04000000	0x3c0fffff	lui \$1, -1	
	0x04000004	0x343a0004	ori \$26, \$1, 4	15: li \$k0, 0xffff0004
	0x04000008	0x3c0fffff	lui \$1, -1	16: li \$k1, 0xffff0000
	0x0400000c	0x343a0000	ori \$27, \$1, 0	
	0x04000010	0x3c0fffff	lui \$1, -1	17: li \$s0, 0xffff000c
	0x04000014	0x34300000	ori \$16, \$1, 12	
	0x04000018			
	0x0400001c			
	0x04000020			
	0x04000024			
	0x04000028			
	0x0400002c			
	0x04000030			
	0x04000034			
	0x04000038			
	0x0400003c			
	0x04000040			
	0x04000044			
	0x04000048			
	0x0400004c			
	0x04000050			
	0x04000054			
	0x04000058			
	0x0400005c			
	0x04000060			
	0x04000064			
	0x04000068			
	0x0400006c			
	0x04000070			
	0x04000074			
	0x04000078			
	0x0400007c			
	0x04000080			
	0x04000084			
	0x04000088			
	0x0400008c			
	0x04000090			
	0x04000094			
	0x04000098			
	0x0400009c			
	0x040000a0			
	0x040000a4			
	0x040000a8			
	0x040000ac			
	0x040000b0			
	0x040000b4			
	0x040000b8			
	0x040000bc			
	0x040000c0			
	0x040000c4			
	0x040000c8			
	0x040000cc			
	0x040000d0			
	0x040000d4			
	0x040000d8			
	0x040000dc			
	0x040000e0			
	0x040000e4			
	0x040000e8			
	0x040000ec			
	0x040000f0			
	0x040000f4			
	0x040000f8			
	0x040000fc			
	0x04000100			
	0x04000104			
	0x04000108			
	0x0400010c			
	0x04000110			
	0x04000114			
	0x04000118			
	0x0400011c			
	0x04000120			
	0x04000124			
	0x04000128			
	0x0400012c			
	0x04000130			
	0x04000134			
	0x04000138			
	0x0400013c			
	0x04000140			
	0x04000144			
	0x04000148			
	0x0400014c			
	0x04000150			
	0x04000154			
	0x04000158			
	0x0400015c			
	0x04000160			