

Ex1:

.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

main: #jal TRACK # draw track line

addi $a0, $zero, 135 # Marsbot rotates 90\* and start running

jal ROTATE

jal GO

sleep1: addi $v0,$zero,32 # Keep running by sleeping in 1000 ms

li $a0,7000

syscall

#jal UNTRACK # keep old track

jal TRACK # and draw new track line

goDOWN: addi $a0, $zero, 150 # Marsbot rotates 180\*

jal ROTATE

sleep2: addi $v0,$zero,32 # Keep running by sleeping in 2000 ms

li $a0,5000

syscall

jal UNTRACK # keep old track

jal TRACK # and draw new track line

goLEFT: addi $a0, $zero, 270 # Marsbot rotates 270\*

jal ROTATE

sleep3: addi $v0,$zero,32 # Keep running by sleeping in 1000 ms

li $a0,5000

syscall

jal UNTRACK # keep old track

jal TRACK # and draw new track line

goASKEW:addi $a0, $zero, 30 # Marsbot rotates 120\*

jal ROTATE

sleep4: addi $v0,$zero,32 # Keep running by sleeping in 2000 ms

li $a0,5000

syscall

jal UNTRACK # keep old track

jal TRACK # and draw new track line

jal STOP

li $v0, 10

syscall

end\_main:

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

# 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

jr $ra

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

# STOP procedure, to stop running

# param[in] none

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

STOP: li $at, MOVING # change MOVING port to 0

sb $zero, 0($at) # to stop

jr $ra

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

# 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

jr $ra

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

# 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

jr $ra

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

# ROTATE procedure, to rotate the robot

# param[in] $a0, An angle between 0 and 359

# 0 : North (up)

# 90: East (right)

# 180: South (down)

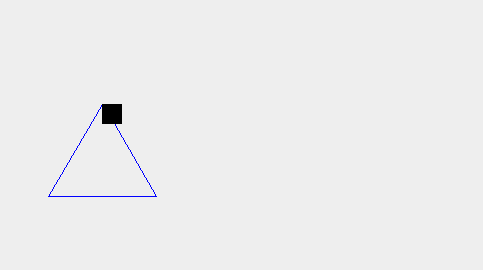
# 270: West (left)

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

ROTATE: li $at, HEADING # change HEADING port

sw $a0, 0($at) # to rotate robot

jr $ra



-------

.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

main: #jal TRACK # draw track line

addi $a0, $zero, 135 # Marsbot rotates 90\* and start running

jal ROTATE

jal GO

sleep1: addi $v0,$zero,32 # Keep running by sleeping in 1000 ms

li $a0,7000

syscall

#jal UNTRACK # keep old track

jal TRACK # and draw new track line

goDOWN: addi $a0, $zero, 90 # Marsbot rotates 180\*

jal ROTATE

sleep2: addi $v0,$zero,32 # Keep running by sleeping in 2000 ms

li $a0,5000

syscall

jal UNTRACK # keep old track

jal TRACK # and draw new track line

goLEFT: addi $a0, $zero, 180 # Marsbot rotates 270\*

jal ROTATE

sleep3: addi $v0,$zero,32 # Keep running by sleeping in 1000 ms

li $a0,5000

syscall

jal UNTRACK # keep old track

jal TRACK # and draw new track line

goASKEW:addi $a0, $zero, 270 # Marsbot rotates 120\*

jal ROTATE

sleep4: addi $v0,$zero,32 # Keep running by sleeping in 2000 ms

li $a0,5000

syscall

jal UNTRACK # keep old track

jal TRACK # and draw new track line

goUP:addi $a0, $zero, 0 # Marsbot rotates 120\*

jal ROTATE

sleep5: addi $v0,$zero,32 # Keep running by sleeping in 2000 ms

li $a0,5000

syscall

jal UNTRACK # keep old track

jal TRACK # and draw new track line

jal STOP

li $v0, 10

syscall

end\_main:

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

# 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

jr $ra

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

# STOP procedure, to stop running

# param[in] none

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

STOP: li $at, MOVING # change MOVING port to 0

sb $zero, 0($at) # to stop

jr $ra

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

# 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

jr $ra

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

# 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

jr $ra

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

# ROTATE procedure, to rotate the robot

# param[in] $a0, An angle between 0 and 359

# 0 : North (up)

# 90: East (right)

# 180: South (down)

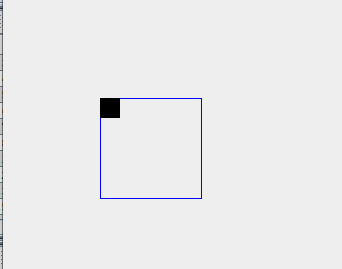
# 270: West (left)

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

ROTATE: li $at, HEADING # change HEADING port

sw $a0, 0($at) # to rotate robot

jr $ra



-----

.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

main: #jal TRACK # draw track line

addi $a0, $zero, 135 # Marsbot rotates 90\* and start running

jal ROTATE

jal GO

sleep1: addi $v0,$zero,32 # Keep running by sleeping in 1000 ms

li $a0,7000

syscall

#jal UNTRACK # keep old track

jal TRACK # and draw new track line

goDOWN: addi $a0, $zero, 162 # Marsbot rotates 180\*

jal ROTATE

sleep2: addi $v0,$zero,32 # Keep running by sleeping in 2000 ms

li $a0,5000

syscall

jal UNTRACK # keep old track

jal TRACK # and draw new track line

goLEFT: addi $a0, $zero, 306 # Marsbot rotates 270\*

jal ROTATE

sleep3: addi $v0,$zero,32 # Keep running by sleeping in 1000 ms

li $a0,5000

syscall

jal UNTRACK # keep old track

jal TRACK # and draw new track line

goASKEW:addi $a0, $zero, 90 # Marsbot rotates 120\*

jal ROTATE

sleep4: addi $v0,$zero,32 # Keep running by sleeping in 2000 ms

li $a0,5000

syscall

jal UNTRACK # keep old track

jal TRACK # and draw new track line

goUP:addi $a0, $zero, 234 # Marsbot rotates 120\*

jal ROTATE

sleep5: addi $v0,$zero,32 # Keep running by sleeping in 2000 ms

li $a0,5000

syscall

jal UNTRACK # keep old track

jal TRACK # and draw new track line

goUPs:addi $a0, $zero, 18 # Marsbot rotates 120\*

jal ROTATE

sleep6: addi $v0,$zero,32 # Keep running by sleeping in 2000 ms

li $a0,5000

syscall

jal UNTRACK # keep old track

jal TRACK # and draw new track line

jal STOP

li $v0, 10

syscall

end\_main:

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

# 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

jr $ra

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

# STOP procedure, to stop running

# param[in] none

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

STOP: li $at, MOVING # change MOVING port to 0

sb $zero, 0($at) # to stop

jr $ra

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

# 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

jr $ra

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

# 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

jr $ra

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

# ROTATE procedure, to rotate the robot

# param[in] $a0, An angle between 0 and 359

# 0 : North (up)

# 90: East (right)

# 180: South (down)

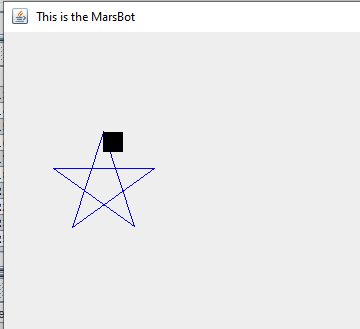
# 270: West (left)

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

ROTATE: li $at, HEADING # change HEADING port

sw $a0, 0($at) # to rotate robot

jr $ra



**Ex2:**

.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 $t3,47

li $t4,58

li $t5,64

li $t7,123

loop: nop

li $t6,91

WaitForKey: lw $t1, 0($k1) # $t1 = [$k1] = KEY\_READY

beq $t1, $zero, WaitForKey # if $t1 == 0 then Polling

ReadKey: lw $t0, 0($k0) # $t0 = [$k0] = KEY\_CODE

WaitForDis: lw $t2, 0($s1) # $t2 = [$s1] = DISPLAY\_READY

beq $t2, $zero, WaitForDis # if $t2 == 0 then Polling

slt $t8,$t3,$t0

slt $t9,$t0,$t4

add $t8,$t8,$t9

beq $t8,2,so

slt $t8,$t5,$t0

slt $t9,$t0,$t6

add $t8,$t8,$t9

beq $t8,2,hoa

li $t6,96

slt $t8,$t6,$t0

slt $t9,$t0,$t7

add $t8,$t8,$t9

beq $t8,2,thuong

j sao

#Encrypt: addi $t0, $t0, 1 # change input key

ShowKey: sw $t0, 0($s0) # show key

nop

j loop

so:

j ShowKey

hoa:

addi $t0, $t0, 32

j ShowKey

thuong:

addi $t0, $t0, -32

j ShowKey

sao:

li $t0,42

j ShowKey

