# Báo cáo thực thành tuần 10 P2

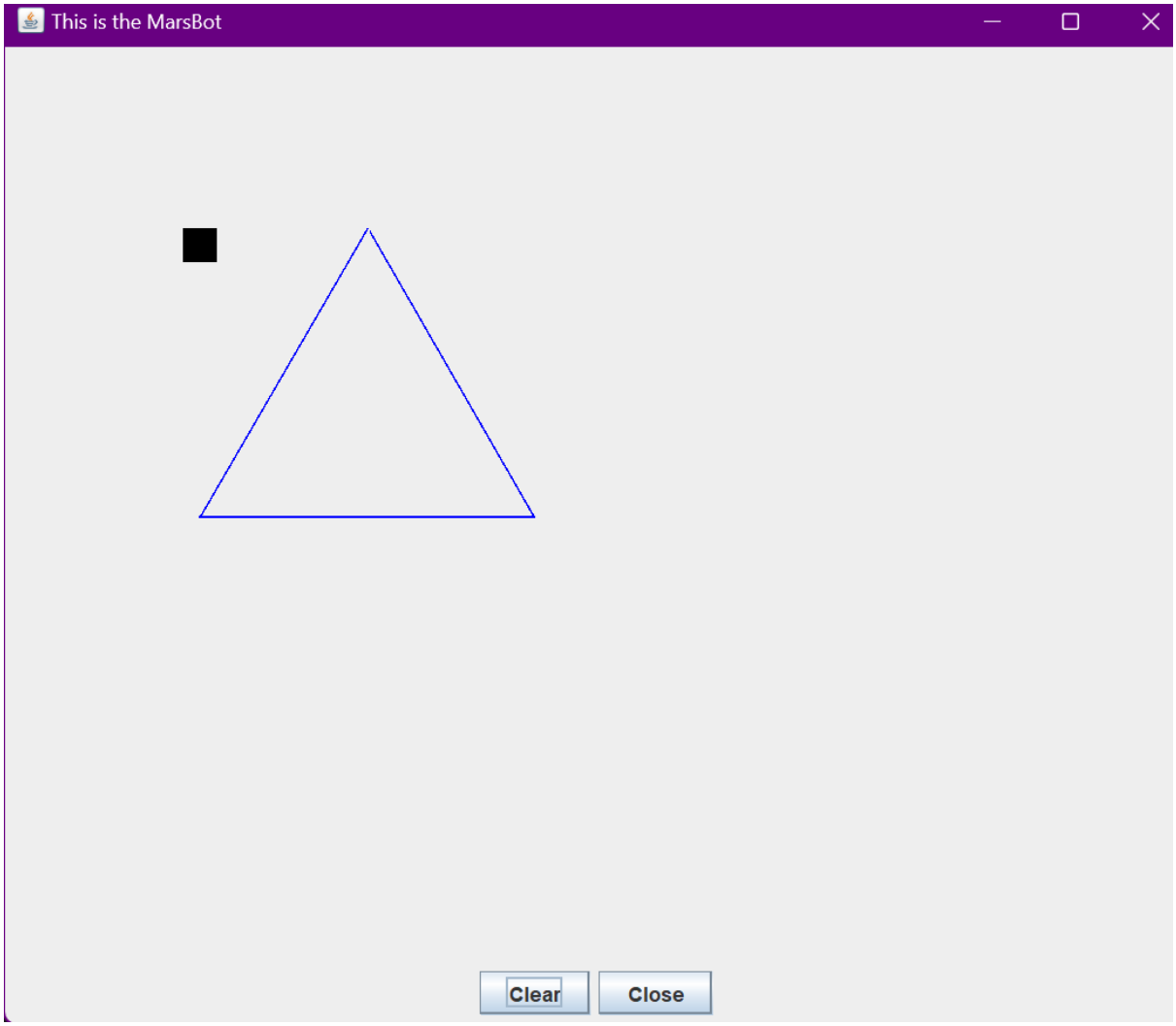
# Phạm Thành Lập 20215076

## *Assignment1:*

- **Tam giác đều :**

```
.eqv HEADING 0xffff8010 # Integer: An angle between 0 and 359
.eqv MOVING 0xffff8050 # Boolean: whether or not to move
.eqv LEAVETRACK 0xffff8020 # Boolean (0 or non-0):
.text
main:
        addi $a0, $zero, 90
running:
        jal ROTATE
        jal GO
sleep:
        addi $v0,$zero,32
        li $a0,10000
        syscall
goDown:
        addi $a0, $zero, 180
        jal ROTATE
sleep1:
        addi $v0,$zero,32
        li $a0,5000
        syscall
        jal TRACK
goASEW_1:
        addi $a0, $zero, 150
        jal ROTATE
sleep2:
        addi $v0,$zero,32
        li $a0,9000
        syscall
        jal UNTRACK
        jal TRACK
goASEW_2:
        addi $a0, $zero, 270
        jal ROTATE
sleep3:
        addi $v0,$zero,32
        li $a0,9000
        syscall
        jal UNTRACK
        jal TRACK
goASKEW_3:
```

```
        addi $a0, $zero, 30
        jal ROTATE
sleep4:
        addi $v0,$zero,32
        li $a0,9000
        syscall
        jal UNTRACK
ESC:
        addi $a0, $zero, 270
        jal ROTATE
sleep5:
        addi $v0,$zero,32
        li $a0,5000
        syscall
end_main:
        jal STOP
        li $v0, 10
        syscall
GO:
        li $at, MOVING
        addi $k0, $zero,1
        sb $k0, 0($at)
        jr $ra
STOP:
        li $at, MOVING # change MOVING port to 0
        sb $zero, 0($at) # to stop
        jr $ra
TRACK:
        li $at, LEAVETRACK # change LEAVETRACK port
        addi $k0, $zero,1 # to logic 1,
        sb $k0, 0($at) # to start tracking
        jr $ra
UNTRACK:
        li $at, LEAVETRACK # change LEAVETRACK port to 0
        sb $zero, 0($at) # to stop drawing tail
        jr $ra
ROTATE:
        li $at, HEADING # change HEADING port
        sw $a0, 0($at) # to rotate robot
        jr $ra
```

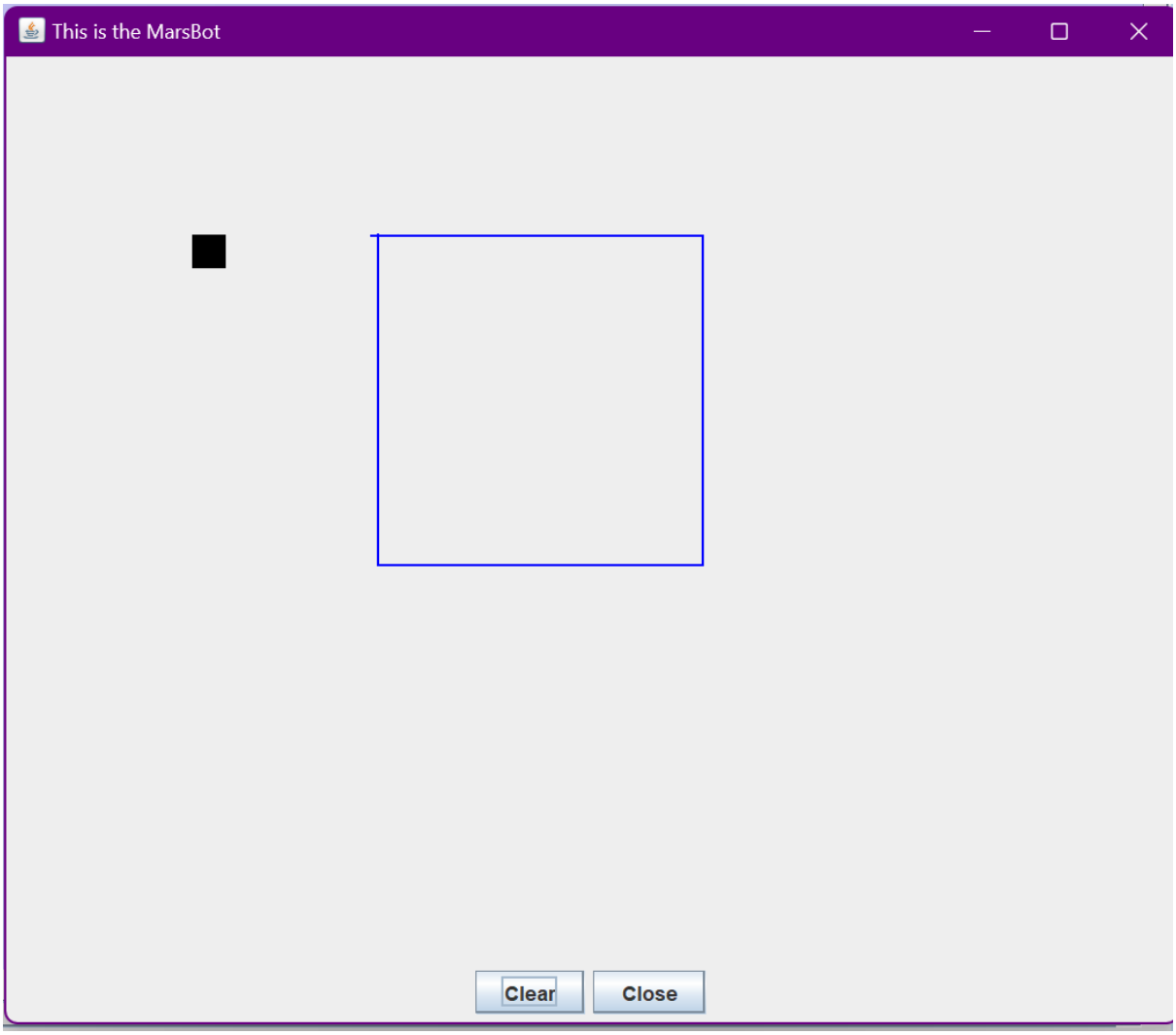



- **Hình vuông :**

```
.eqv HEADING 0xffff8010 # Integer: An angle between 0 and 359
.eqv MOVING 0xffff8050 # Boolean: whether or not to move
.eqv LEAVETRACK 0xffff8020 # Boolean (0 or non-0):
.text
main:
        addi $a0, $zero, 90
running:
        jal ROTATE
        jal GO
sleep:
        addi $v0,$zero,32
        li $a0,10000
        syscall
goDown:
        addi $a0, $zero, 180
        jal ROTATE
sleep1:
```

```
        addi $v0,$zero,32
        li $a0,5000
        syscall
        jal TRACK
goRIGHT:
        addi $a0, $zero, 90
        jal ROTATE
sleep2:
        addi $v0,$zero,32
        li $a0,9000
        syscall
        jal UNTRACK
        jal TRACK
goDOWN:
        addi $a0, $zero, 180
        jal ROTATE
sleep3:
        addi $v0,$zero,32
        li $a0,9000
        syscall
        jal UNTRACK
        jal TRACK
goLEFT:
        addi $a0, $zero, 270
        jal ROTATE
sleep4:
        addi $v0,$zero,32
        li $a0,9000
        syscall
        jal UNTRACK
        jal TRACK
goTOP:
        addi $a0, $zero, 0
        jal ROTATE
sleep5:
        addi $v0,$zero,32
        li $a0,9000
        syscall
        jal UNTRACK
ESC:
        addi $a0, $zero, 270
        jal ROTATE
sleep6:
        addi $v0,$zero,32
        li $a0,5000
        syscall
```

```
end_main:
        jal STOP
        li $v0, 10
        syscall
GO:
        li $at, MOVING
        addi $k0, $zero,1
        sb $k0, 0($at)
        jr $ra
STOP:
        li $at, MOVING # change MOVING port to 0
        sb $zero, 0($at) # to stop
        jr $ra
TRACK:
        li $at, LEAVETRACK # change LEAVETRACK port
        addi $k0, $zero,1 # to logic 1,
        sb $k0, 0($at) # to start tracking
        jr $ra
UNTRACK:
        li $at, LEAVETRACK # change LEAVETRACK port to 0
        sb $zero, 0($at) # to stop drawing tail
        jr $ra
ROTATE:
        li $at, HEADING # change HEADING port
        sw $a0, 0($at) # to rotate robot
        jr $ra
```

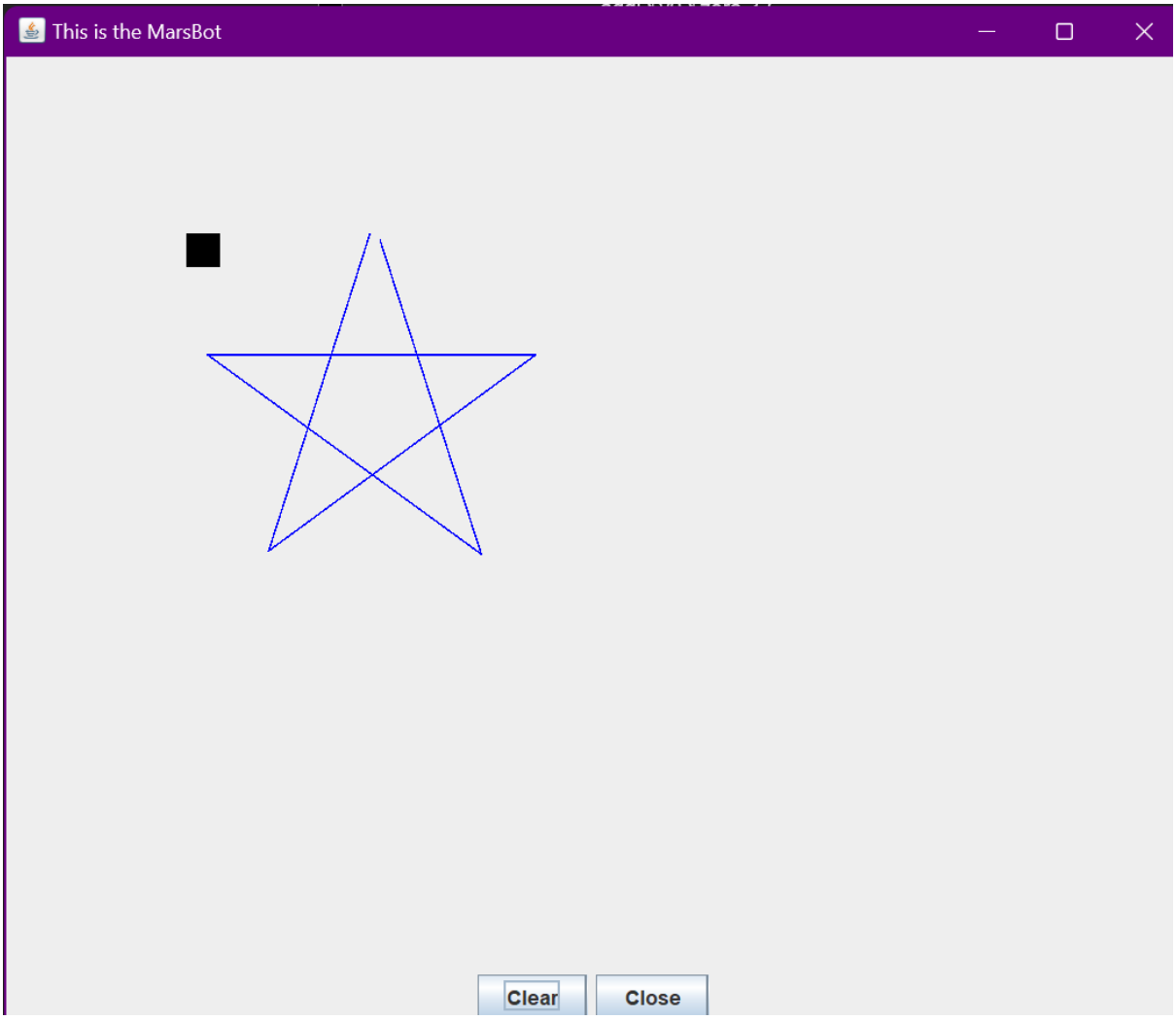



- **Hình sao :**

```
.eqv HEADING 0xffff8010 # Integer: An angle between 0 and 359
.eqv MOVING 0xffff8050 # Boolean: whether or not to move
.eqv LEAVETRACK 0xffff8020 # Boolean (0 or non-0):
.text
main:
        addi $a0, $zero, 90
running:
        jal ROTATE
        jal GO
sleep:
        addi $v0,$zero,32
        li $a0,10000
        syscall
goDown:
        addi $a0, $zero, 180
        jal ROTATE
sleep1:
        addi $v0,$zero,32
```

```
        li $a0,5000
        syscall
        jal TRACK
goASEW_1:
        addi $a0, $zero, 162
        jal ROTATE
sleep2:
        addi $v0,$zero,32
        li $a0,9000
        syscall
        jal UNTRACK
        jal TRACK
goASEW_2:
        addi $a0, $zero, 306
        jal ROTATE
sleep3:
        addi $v0,$zero,32
        li $a0,9000
        syscall
        jal UNTRACK
        jal TRACK
goRIGHT:
        addi $a0, $zero, 90
        jal ROTATE
sleep4:
        addi $v0,$zero,32
        li $a0,9000
        syscall
        jal UNTRACK
        jal TRACK
goASEW_3:
        addi $a0, $zero, 234
        jal ROTATE
sleep5:
        addi $v0,$zero,32
        li $a0,9000
        syscall
        jal UNTRACK
        jal TRACK
goASEW_4:
        addi $a0, $zero, 18
        jal ROTATE
sleep6:
        addi $v0,$zero,32
        li $a0,9000
        syscall
```

```
        jal UNTRACK
ESC:
        addi $a0, $zero, 270
        jal ROTATE
sleep7:
        addi $v0,$zero,32
        li $a0,5000
        syscall
end_main:
        jal STOP
        li $v0, 10
        syscall
GO:
        li $at, MOVING
        addi $k0, $zero,1
        sb $k0, 0($at)
        jr $ra
STOP:
        li $at, MOVING # change MOVING port to 0
        sb $zero, 0($at) # to stop
        jr $ra
TRACK:
        li $at, LEAVETRACK # change LEAVETRACK port
        addi $k0, $zero,1 # to logic 1,
        sb $k0, 0($at) # to start tracking
        jr $ra
UNTRACK:
        li $at, LEAVETRACK # change LEAVETRACK port to 0
        sb $zero, 0($at) # to stop drawing tail
        jr $ra
ROTATE:
        li $at, HEADING # change HEADING port
        sw $a0, 0($at) # to rotate robot
        jr $ra
```

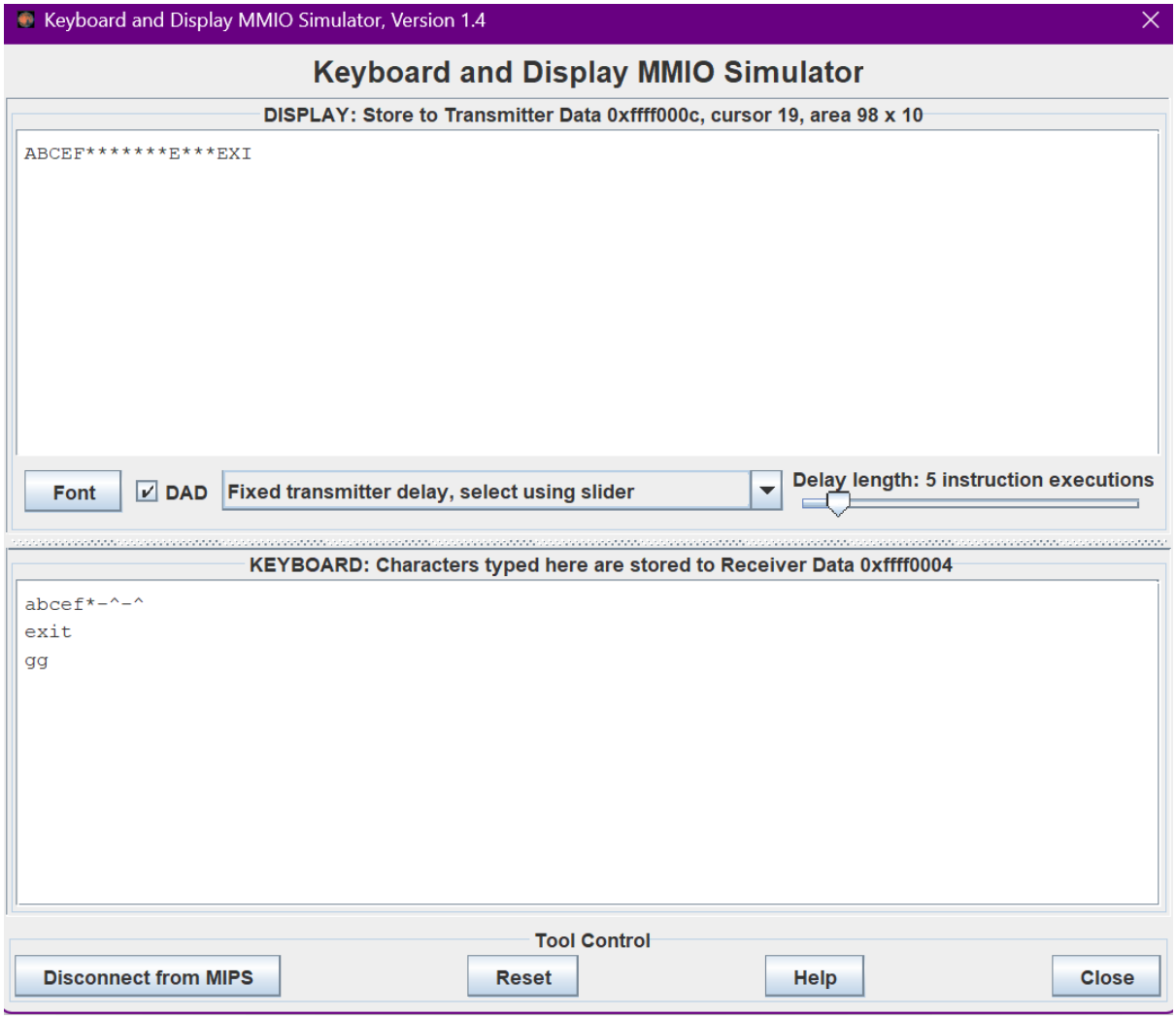



## *Assignment2:*

```
.eqv KEY_CODE 0xFFFF0004
.eqv KEY_READY 0xFFFF0000
.eqv DISPLAY_CODE 0xFFFF000C
.eqv DISPLAY_READY 0xFFFF0008
.text
        li $k0, KEY_CODE
        li $k1, KEY_READY
        li $s0, DISPLAY_CODE
        li $s1, DISPLAY_READY
loop:   nop
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
Check1:
        bge $t0, 48, Number
        j ELSE
Check2:
        bge $t0, 65, LowCase
        j ELSE
Check3:
        bge $t0, 97, UpCase
ELSE:
        addi $t0, $zero, 42
ShowKey:
        beq $t0, 69, CheckExit1
        beq $t0, 88, CheckExit2
        beq $t0, 73, CheckExit3
        beq $t0, 84, CheckExit4
Reset:
        addi $v0, $zero, 0
Show:
        sw $t0, 0($s0) # show key
        nop
        j loop
Number:
        ble $t0, 57, ShowKey
        j Check2
```

```
LowCase:
        bgt $t0, 90, Check3
        addi $t0, $t0, 32
        j ShowKey
UpCase:
        bgt $t0, 122, ELSE
        addi $t0, $t0, -32
        j ShowKey
CheckExit1:
        bnez $v0, Reset
        addi $v0, $v0, 1
        j Show
CheckExit2:
        bne $v0, 1, Reset
        addi $v0, $v0, 1
        j Show
CheckExit3:
        bne $v0, 2, Reset
        addi $v0, $v0, 1
        j Show
CheckExit4:
        bne $v0, 3, Reset
End:
        li $v0, 10
        syscall
```




## *Assignment2:*

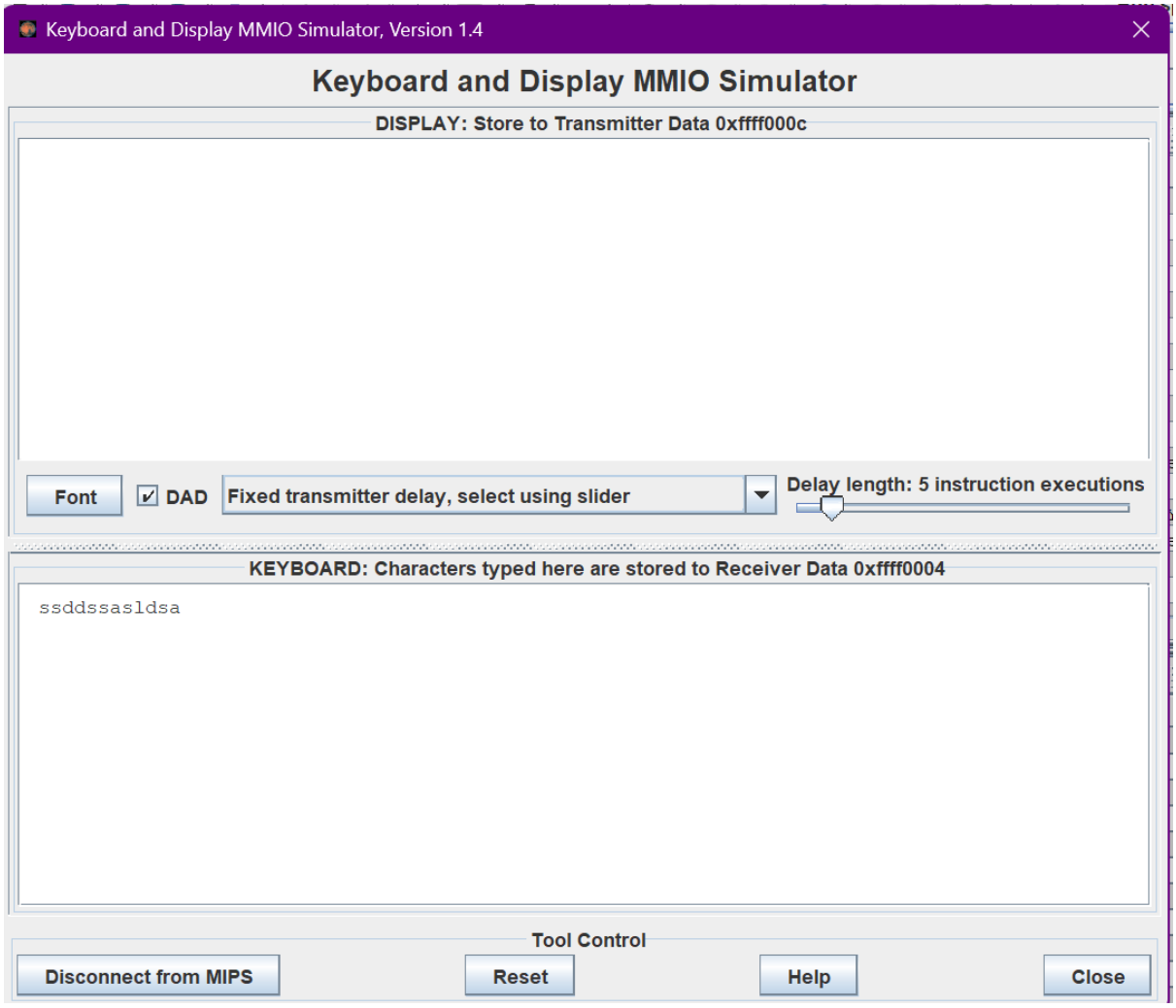
```
.eqv HEADING 0xffff8010 # Integer: An angle between 0 and 359
.eqv MOVING 0xffff8050 # Boolean: whether or not to move
.eqv LEAVETRACK 0xffff8020 # Boolean (0 or non-0):
.eqv KEY_CODE 0xFFFF0004
.eqv KEY_READY 0xFFFF0000
.eqv DISPLAY_CODE 0xFFFF000C
.eqv DISPLAY_READY 0xFFFF0008
.text
main:

        li $k0, KEY_CODE
```

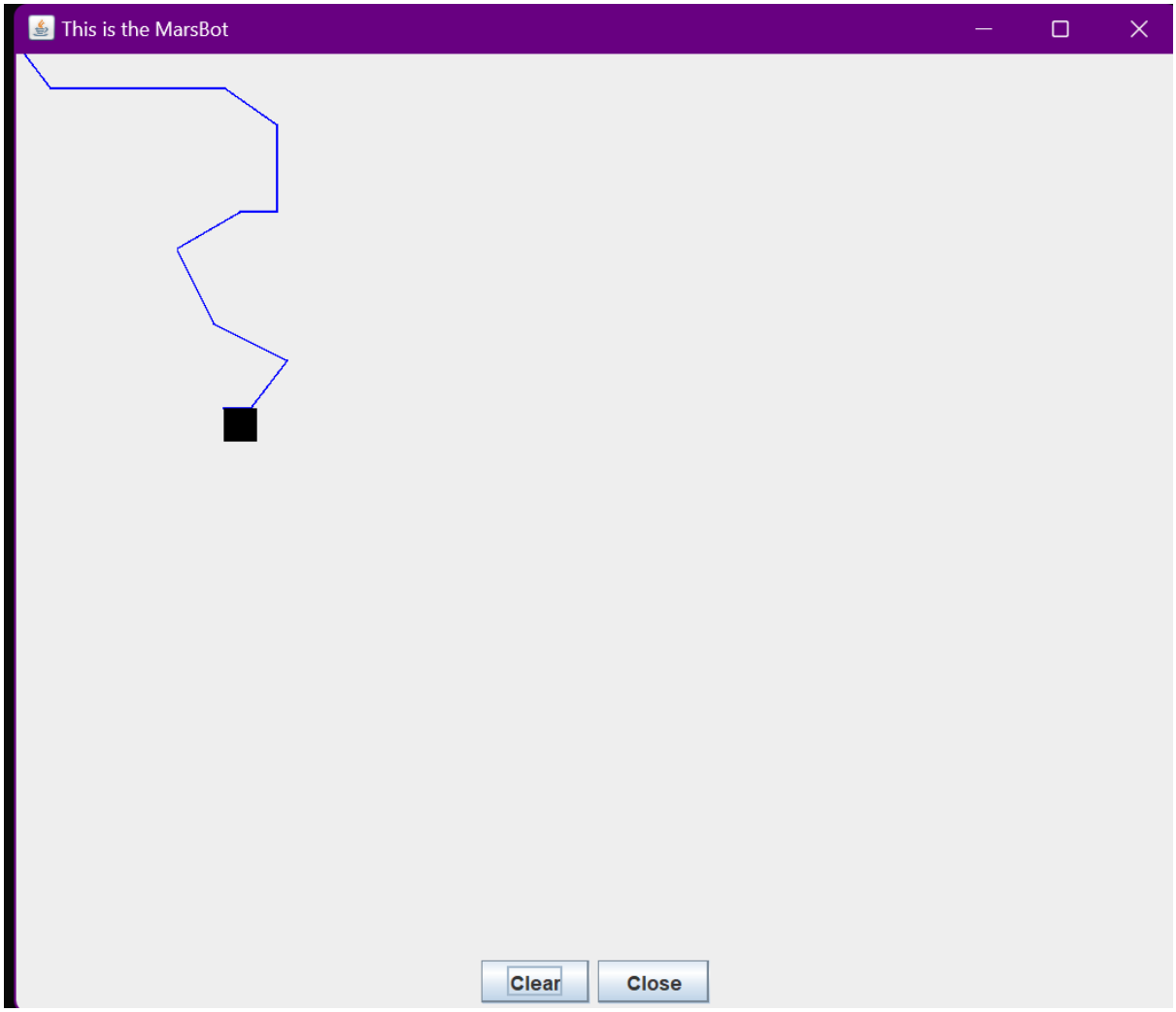
```
        li $k1, KEY_READY
        li $s0, DISPLAY_CODE
        li $s1, DISPLAY_READY
loop:   nop
WaitForKey:
        lw $t1, 0($k1) # $t1 = [$k1] = KEY_READY
        beq $t1, $zero, WaitForKey # if $t1 == 0 then Polling
ReadKey:
        lw $t0, 0($k0) # $t0 = [$k0] = KEY_CODE
WaitForDis:
        lw $t2, 0($s1) # $t2 = [$s1] = DISPLAY_READY
        beq $t2, $zero, WaitForDis # if $t2 == 0 then Polling
Check:
        beq $t0, 32, Start # show key #W87 w119 A65 a97 S83 s115 D67 d100
space32
        beq $t0, 87, goTop
        beq $t0, 119, goTop
        beq $t0, 83, goDown
        beq $t0, 115, goDown
        beq $t0, 67,goRight
        beq $t0, 100, goRight
        beq $t0, 65, goLeft
        beq $t0, 97, goLeft
        j loop
Start:
        bnez $t3, Exit
        addi $t3, $t3, 1
        jal GO
        jal TRACK
        j loop
goTop:
```

```
        addi $a0, $zero, 0
        jal ROTATE
        addi $v0,$zero,32
        li $a0,1000
        syscall
        jal UNTRACK
        jal TRACK
        j loop
goDown:
        addi $a0, $zero, 180
        jal ROTATE
        addi $v0,$zero,32
        li $a0,1000
        syscall
        jal UNTRACK
        jal TRACK
        j loop
goRight:
        addi $a0, $zero, 90
        jal ROTATE
        addi $v0,$zero,32
        li $a0,1000
        syscall
        jal UNTRACK
        jal TRACK
        j loop
goLeft:
        addi $a0, $zero, 270
        jal ROTATE
        addi $v0,$zero,32
```

```
        li $a0,1000
        syscall
        jal UNTRACK
        jal TRACK
        j loop
Exit:
        jal STOP
        li $v0, 10
        syscall
GO:
        li $at, MOVING
        addi $v1, $zero,1
        sb $v1, 0($at)
        jr $ra
STOP:
        li $at, MOVING # change MOVING port to 0
        sb $zero, 0($at) # to stop
        jr $ra
TRACK:
        li $at, LEAVETRACK # change LEAVETRACK port
        addi $v1, $zero,1 # to logic 1,
        sb $v1, 0($at) # to start tracking
        jr $ra
UNTRACK:
        li $at, LEAVETRACK # change LEAVETRACK port to 0
        sb $zero, 0($at) # to stop drawing tail
        jr $ra
ROTATE:
        li $at, HEADING # change HEADING port
        sw $a0, 0($at) # to rotate robot
```

jr $ra


Keyboard and Display MMIO Simulator, Version 1.4
Keyboard and Display MMIO Simulator
DISPLAY: Store to Transmitter Data 0xffff000c
Font
DAD
Fixed transmitter delay, select using slider
Delay length: 5 instruction executions
KEYBOARD: Characters typed here are stored to Receiver Data 0xffff0004
ssddssasldsa
Tool Control
Disconnect from MIPS
Reset
Help
Close


This is the MarsBot
Clear
Close