POST LAB - 09

NAME - RAJENDRA SINGH
ROLL NO. - 111601017

Problem:

The temperature inside a laboratory should be maintained at 20 0C. Design a temperature controller system that operates as below:

The system should poll for the outside temperature every 60 time units. The temperature thus obtained must be then adjusted to get the required room temperature (20 0C).

After each poll, the difference between the temperature outside and the required room temperature must be stored in an array named 'error_array'.
This process should continue until the outside temperature becomes equal to the required room temperature. It should be ensured that the temperature outside has become Stable.

Once the temperature outside is stable, display the error_array.
Use loops, procedures, nested procedures for iterative and reusable operations respectively. Provide comments and display statements wherever required.

CODE :

```
# lab-9
# temperature_control.asm

.data
    optimal_temperature : .word 20      # 20 degree C
    wait_time           : .word 60      # 60 minutes
    error_array         : .word 0:1000  # array declaration of maxsize: 1000
    array_capacity      : .word 1000    # array capacity
    prompt_poll         : .asciiz "Enter the current temperature: "
    prompt_optimal      : .asciiz "Optimal Temperature: "
    prompt_content_size : .asciiz "Total entries in `error_array` : "
    newline             : .asciiz "\n"        # new-line character
    temp_eq             : .asciiz "\nTemperature is equal to optimal temperature.\n"
    stable              : .asciiz "\nTemperature is stable.\n"
.text
.globl main
.ent main
    # load the base address
    la      $t0 , error_array
    move    $s0 , $t0       # safe copy
    # load the array capacity
    lw      $s1 , array_capacity
```

```
# set the initial content size to 0
    move    $t1 , $zero
    # load the optimal temperature
    lw      $s2 , optimal_temperature

    # print the optimal temperature on the screen
    li      $v0 , 4
        la      $a0 , prompt_optimal      # prompt
        syscall
        li      $v0 , 1
        lw      $a0 , optimal_temperature # integer to print
        syscall
    li $v0 , 4
        la $a0 , newline
        syscall


# break out this loop if anyone of the give
# two cases are true:
# (i) if the final temperature is equal to optimal_temperature
# (ii) or the current temperature is equal to the previous temperature
```

```
# input the first poll
    # ask the user to input the temperature
    li      $v0 , 4
        la      $a0 , prompt_poll       # prompt
        syscall
        li      $v0 , 5
        syscall
        move    $t9 , $v0
    # store this number into the array
    sub     $t9, $t9, $s2           # error: outside_temp - optimal_temp
    sw      $t9 , ($t0)
    # increment the $t1 : the current content size of the array
        addi    $t1 , $t1 , 1
condition1:
    # if condition (i) is satisfied
    beq             $t9, $zero, out_of_loop_one
    # incrementing the base address
    addi    $t0 , $t0 , 4

temperature_poll:
    # now it's sure here that size is greater than equal to 1 at any time.
    # ask the user to input the temperature
    li      $v0 , 4
        la      $a0 , prompt_poll       # prompt
```

```
syscall
        li      $v0 , 5
        syscall
        move    $t9 , $v0
    # store this number into the array
    sub     $t9, $t9, $s2           # error: outside_temp - optimal_temp
    sw      $t9 , ($t0)
    # increment the $t1 : the current content size of the array
        addi    $t1 , $t1 , 1
condition_one:
    # if condition (i) is satisfied
    beq             $t9, $zero, out_of_loop_one
condition_two:
    # if condition (ii) is satisfied
    sub     $t8, $t0 , 4
    lw      $t8 , ($t8)     # retrieve the previous value
    beq             $t8, $t9, out_of_loop_two

    # incrementing the base address
    addi    $t0 , $t0 , 4
    # if the array is not full: then branching
    blt     $t1 , $s1 , temperature_poll
```

```
out_of_loop_one:
    li      $v0 , 4
            la      $a0 , temp_eq       # prompt
            syscall
    j print_the_array
out_of_loop_two:
    li      $v0 , 4
            la      $a0 , stable        # prompt
            syscall

print_the_array:
    li $v0 , 4
            la $a0 , newline
            syscall
    # print the size of the array
    li      $v0 , 4
            la      $a0 , prompt_content_size       # prompt
            syscall
            li      $v0 , 1
            move    $a0 , $t1 # integer to print
            syscall
    li $v0 , 4
            la $a0 , newline
            syscall
```

```
# printing the error_array
    move    $t0 , $s0 # set the base address
    li $t3 , 0          # loop index - (0 to size-1)
print_loop:
        li $v0 , 1
        lw $a0 , ($t0)
        syscall
        li $v0 , 4
        la $a0 , newline
        syscall
        addi $t0 , $t0 , 4
        addi $t3 , $t3 , 1
        blt $t3 , $t1 , print_loop
        j exit

exit:
        li $v0 , 10
        syscall
.end main
```

CASE 1:

WHEN TEMPERATURE BECOME STABLE

FP Regs | nt Regs [10] | Text | Data

Int Regs [10]

```
PC       = 4194656
EPC      = 0
Cause    = 0
BadVAddr = 0
Status   = 805371664

HI       = 0
LO       = 0

R0  [r0] = 0
R1  [at] = 0
R2  [v0] = 10
R3  [v1] = 0
R4  [a0] = 268505092
R5  [a1] = 2147482032
R6  [a2] = 2147482040
R7  [a3] = 0
R8  [t0] = 268501024
R9  [t1] = 6
R10 [t2] = 0
R11 [t3] = 6
R12 [t4] = 0
R13 [t5] = 0
R14 [t6] = 0
R15 [t7] = 0
R16 [s0] = 268501000
R17 [s1] = 1000
R18 [s2] = 20
R19 [s3] = 0
R20 [s4] = 0
R21 [s5] = 0
R22 [s6] = 0
R23 [s7] = 0
R24 [t8] = 36
R25 [t9] = 36
R26 [k0] = 0
R27 [k1] = 0
```

Text

```
[004000cc] 34241006 ori $4, $1, 4102 [temp_eq]
[004000d0] 0000000c syscall                    ; 96: syscall
[004000d4] 0810003a j 0x004000e8 [print_the_array]; 97: j print_the_array
[004000d8] 34020004 ori $2, $0, 4              ; 99: li $v0 , 4
[004000dc] 3c011001 lui $1, 4097 [stable]      ; 100: la $a0 , stable # prompt
[004000e0] 34241035 ori $4, $1, 4149 [stable]
[004000e4] 0000000c syscall                    ; 101: syscall
[004000e8] 34020004 ori $2, $0, 4              ; 104: li $v0 , 4
[004000ec] 3c011001 lui $1, 4097 [newline]     ; 105: la $a0 , newline
[004000f0] 34241004 ori $4, $1, 4100 [newline]
[004000f4] 0000000c syscall                    ; 106: syscall
[004000f8] 34020004 ori $2, $0, 4              ; 108: li $v0 , 4
[004000fc] 3c011001 lui $1, 4097 [prompt_content_size]
[00400100] 34240fe2 ori $4, $1, 4066 [prompt_content_size]
[00400104] 0000000c syscall                    ; 110: syscall
[00400108] 34020001 ori $2, $0, 1              ; 111: li $v0 , 1
[0040010c] 00092021 addu $4, $0, $9            ; 112: move $a0 , $t1 # integer to print
[00400110] 0000000c syscall                    ; 113: syscall
[00400114] 34020004 ori $2, $0, 4              ; 114: li $v0 , 4
[00400118] 3c011001 lui $1, 4097 [newline]     ; 115: la $a0 , newline
[0040011c] 34241004 ori $4, $1, 4100 [newline]
[00400120] 0000000c syscall                    ; 116: syscall
[00400124] 00104021 addu $8, $0, $16           ; 118: move $t0 , $s0 # set the base address
[00400128] 340b0000 ori $11, $0, 0             ; 119: li $t3 , 0 # loop index - (0 to size-1)
[0040012c] 34020001 ori $2, $0, 1              ; 121: li $v0 , 1
[00400130] 8d040000 lw $4, 0($8)               ; 122: lw $a0 , ($t0)
[00400134] 0000000c syscall                    ; 123: syscall
[00400138] 34020004 ori $2, $0, 4              ; 124: li $v0 , 4
[0040013c] 3c011001 lui $1, 4097 [newline]     ; 125: la $a0 , newline
[00400140] 34241004 ori $4, $1, 4100 [newline]
[00400144] 0000000c syscall                    ; 126: syscall
[00400148] 21080004 addi $8, $8, 4             ; 127: addi $t0 , $t0 , 4
[0040014c] 216b0001 addi $11, $11, 1           ; 128: addi $t3 , $t3 , 1
[00400150] 0169082a slt $1, $11, $9            ; 129: blt $t3 , $t1 , print_loop
[00400154] 1420fff6 bne $1, $0, -40 [print_loop-0x00400154]
[00400158] 08100057 j 0x0040015c [exit]        ; 130: j exit
[0040015c] 3402000a ori $2, $0, 10             ; 133: li $v0 , 10
[00400160] 0000000c syscall                    ; 134: syscall
```

CASE 1 CONSOLE :

```
Optimal Temperature: 20
Enter the current temperature: 10
Enter the current temperature: -10
Enter the current temperature: 303
Enter the current temperature: 40
Enter the current temperature: 56
Enter the current temperature: 56

Temperature is stable.

Total entries in `error_array` : 6
-10
-30
283
20
36
36
```

CASE 2:

WHEN TEMPERATURE REACH OPTIMAL TEMPERATURE

FP Regs | nt Regs [10] | Text | Data

Int Regs [10]

```
PC       = 4194656
EPC      = 0
Cause    = 0
BadVAddr = 0
Status   = 805371664

HI       = 0
LO       = 0

R0  [r0] = 0
R1  [at] = 0
R2  [v0] = 10
R3  [v1] = 0
R4  [a0] = 268505092
R5  [a1] = 2147482032
R6  [a2] = 2147482040
R7  [a3] = 0
R8  [t0] = 268501028
R9  [t1] = 7
R10 [t2] = 0
R11 [t3] = 7
R12 [t4] = 0
R13 [t5] = 0
R14 [t6] = 0
R15 [t7] = 0
R16 [s0] = 268501000
R17 [s1] = 1000
R18 [s2] = 20
R19 [s3] = 0
R20 [s4] = 0
R21 [s5] = 0
R22 [s6] = 0
R23 [s7] = 0
R24 [t8] = 3
R25 [t9] = 0
R26 [k0] = 0
R27 [k1] = 0
```

Text

```
                    User Text Segment [00400000]..[00440000]
[00400000] 3c011001  lui $1, 4097 [error_array]; 19: la $t0 , error_array
[00400004] 34280008  ori $8, $1, 8 [error_array]
[00400008] 00088021  addu $16, $0, $8          ; 20: move $s0 , $t0 # safe copy
[0040000c] 3c011001  lui $1, 4097              ; 22: lw $s1 , array_capacity
[00400010] 8c310fa8  lw $17, 4008($1)
[00400014] 00004821  addu $9, $0, $0           ; 24: move $t1 , $zero
[00400018] 3c011001  lui $1, 4097              ; 26: lw $s2 , optimal_temperature
[0040001c] 8c320000  lw $18, 0($1)
[00400020] 34020004  ori $2, $0, 4             ; 29: li $v0 , 4
[00400024] 3c011001  lui $1, 4097 [prompt_optimal]; 30: la $a0 , prompt_optimal # prompt
[00400028] 34240fcc  ori $4, $1, 4044 [prompt_optimal]
[0040002c] 0000000c  syscall                   ; 31: syscall
[00400030] 34020001  ori $2, $0, 1             ; 32: li $v0 , 1
[00400034] 3c011001  lui $1, 4097              ; 33: lw $a0 , optimal_temperature # integer to print
[00400038] 8c240000  lw $4, 0($1)
[0040003c] 0000000c  syscall                   ; 34: syscall
[00400040] 34020004  ori $2, $0, 4             ; 35: li $v0 , 4
[00400044] 3c011001  lui $1, 4097 [newline]    ; 36: la $a0 , newline
[00400048] 34241004  ori $4, $1, 4100 [newline]
[0040004c] 0000000c  syscall                   ; 37: syscall
[00400050] 34020004  ori $2, $0, 4             ; 48: li $v0 , 4
[00400054] 3c011001  lui $1, 4097 [prompt_poll]; 49: la $a0 , prompt_poll # prompt
[00400058] 34240fac  ori $4, $1, 4012 [prompt_poll]
[0040005c] 0000000c  syscall                   ; 50: syscall
[00400060] 34020005  ori $2, $0, 5             ; 51: li $v0 , 5
[00400064] 0000000c  syscall                   ; 52: syscall
[00400068] 0002c821  addu $25, $0, $2          ; 53: move $t9 , $v0
[0040006c] 0332c822  sub $25, $25, $18         ; 55: sub $t9, $t9, $s2 # error: outside_temp -
optimal_temp
[00400070] ad190000  sw $25, 0($8)             ; 56: sw $t9 , ($t0)
[00400074] 21290001  addi $9, $9, 1            ; 58: addi $t1 , $t1 , 1
[00400078] 13200013  beq $25, $0, 76 [out_of_loop_one-0x00400078]
[0040007c] 21080004  addi $8, $8, 4            ; 63: addi $t0 , $t0 , 4
[00400080] 34020004  ori $2, $0, 4             ; 68: li $v0 , 4
[00400084] 3c011001  lui $1, 4097 [prompt_poll]; 69: la $a0 , prompt_poll # prompt
[00400088] 34240fac  ori $4, $1, 4012 [prompt_poll]
```

CASE 2 CONSOLE :

Console

```
Optimal Temperature: 20
Enter the current temperature: 30
Enter the current temperature: 40
Enter the current temperature: 50
Enter the current temperature: 71
Enter the current temperature: 23
Enter the current temperature: 21
Enter the current temperature: 20

Temperature is equal to optimal temperature.

Total entries in `error_array` : 7
10
20
30
51
3
1
0
```