EXERCISE

1. **Apply Loop Unrolling** for processing 4 elements at once.

SOURCE CODE

```
#CODE CREATED BY TAMIA NAEEM CTAI-004
.data
list: .word 1,3,5,7,9,11,13,15
size: .word 8
msg: .asciiz "Array elements sum is:"
.text
.globl main
main:
la $t0,list
li $t1,0
lw $t2,size
li $t3,0
loop:
bge $t1,$t2,end
mul $t4,$t1,4
add $t4,$t0,$t4
lw $t5,0($t4)
lw $t6,4($t4)
lw $t7,8($t4)
lw $t8,12($t4)
add $t3,$t3,$t5
add $t3,$t3,$t6
add $t3,$t3,$t7
add $t3,$t3,$t8
addi $t1,$t1,4
j loop
end:
```

```
li $v0,4
la $a0,msg
syscall
li $v0,1
move $a0,$t3
syscall
li $v0,10
syscall
```

OUTPUT:

```
Array elements sum is:64
-- program is finished running
```

2. **Modify the loop** to process 2 elements and handle an odd-sized array (e.g., 7 elements).
SOURCE CODE

```
#CODE CREATED BY TAMIA NAEEM CTAI-004
.data
list: .word 1,3,5,7,9,11,13
size: .word 7
msg: .asciiz "Array elements sum is:"
.text
.globl main
main:
    la $t0, list
    li $t1, 0
    lw $t2, size
    li $t3, 0

    rem $t7, $t2, 2
    beq $t7, 0, even_case

    subi $t2, $t2, 1
    mul $t4, $t2, 4
    add $t4, $t0, $t4
    lw $t5, 0($t4)
    add $t3, $t3, $t5

even_case:
    li $t1, 0
loop:
    bge $t1, $t2, end
```

```
    mul $t4, $t1, 4
    add $t4, $t0, $t4
    lw $t5, 0($t4)
    lw $t6, 4($t4)
    add $t3, $t3, $t5
    add $t3, $t3, $t6

    addi $t1, $t1, 2
    j loop

end:
    li $v0, 4
    la $a0, msg
    syscall

    li $v0, 1
    move $a0, $t3
    syscall

    li $v0, 10
    syscall
```

OUTPUT:

```
Array elements sum is:49
-- program is finished running --
```

3. **Compare execution time** of normal loop vs unrolled loop.

   A regular loop processes just one item in each iteration, but an unrolled loop can handle several items (e.g., 4) at once.

   This makes unrolled loops typically faster because they:

- Cut down the number of instructions used to control the loop (like jumps and checks)
- Decrease the performance hit from frequent branching

  The downside is that unrolling slightly increases the length of the code.

4. **Implement loop unrolling** for vector subtraction.
   SOURCE CODE

```
#CODE CREATED BY TAMIA NAEEM CTAI-004
.data
array1: .word 12,20,30,40
array2: .word 1,32,43,4
array3: .space 16
msg: .asciiz "Result: "

.text
.globl main

main:
la $t0,array1
la $t1,array2
la $t2,array3
li $t3,0
li $t4,4

loop:
bge $t3,$t4,print_result
lw $t5,0($t0)
lw $t6,0($t1)
sub $t7,$t5,$t6
sw $t7,0($t2)
addi $t0,$t0,4
addi $t1,$t1,4
addi $t2,$t2,4
addi $t3,$t3,1
```

```
j loop

print_result:
li $v0,4
la $a0,msg
syscall

la $t2,array3
li $t3,0
li $t4,4

print_loop:
bge $t3,$t4,exit
lw $a0,0($t2)
li $v0,1
syscall
li $v0,11
la $a0,32
syscall
addi $t2,$t2,4
addi $t3,$t3,1
j print_loop

exit:
li $v0,10
syscall
    syscall
```

OUTPUT:

```
Result: 11 -12 -13 36
-- program is finished running --
```

5. **Write a program** to add two arrays of 8 elements using loop unrolling (4 elements per iteration).

   SOURCE CODE

```
#CODE CREATED BY TAMIA NAEEM CTAI-004
.data
array1: .word 12,32,43,4,5,32,73,3
array2: .word 1,2,3,4,5,6,7,8
array3: .space 32
msg: .asciiz "Result:\n"

.text
.globl main

main:
la $t0, array1
la $t1, array2
la $t2, array3
li $t3, 0
li $t4, 2

loop:
bge $t3, $t4, print_result
lw $t5, 0($t0)
lw $t6, 0($t1)
add $t7, $t5, $t6
sw $t7, 0($t2)

lw $t5, 4($t0)
lw $t6, 4($t1)
```

```
add $t7, $t5, $t6
sw $t7, 4($t2)

lw $t5, 8($t0)
lw $t6, 8($t1)
add $t7, $t5, $t6
sw $t7, 8($t2)

lw $t5, 12($t0)
lw $t6, 12($t1)
add $t7, $t5, $t6
sw $t7, 12($t2)

addi $t0, $t0, 16
addi $t1, $t1, 16
addi $t2, $t2, 16
addi $t3, $t3, 1
j loop

print_result:
li $v0, 4
la $a0, msg
syscall
```

```
la $t2, array3
li $t3, 0
li $t4, 8

print_loop:
bge $t3, $t4, exit
lw $a0, 0($t2)
li $v0, 1
syscall
li $v0, 11
la $a0, 32
syscall
addi $t2, $t2, 4
addi $t3, $t3, 1
j print_loop

exit:
li $v0, 10
syscall
```

OUTPUT:

```
Result:
13 34 46 8 10 38 80 11
-- program is finished running
```