Código empleado

```
.data
.word 0, 0, 0
.word 0
.word 4
.word 24
.word 25, -3, -6, 1
.word 23, -1, -4, -1

.text
lw $1, 0x14($0)
add $2, $2, $1
add $3, $3, $1
lw $1, 0x10($0)
lsl $7, $1, 2
add $3, $3, $7

loop: beq $1, $0, exit
      lw $4, 0($2)
      lw $5, 0($3)
      subabs $7, $4, $5
      add $6, $6, $7
      addi $2, $2, 4
      addi $3, $3, 4
      addi $1, $1, -1
      j loop

exit: sw $6, 0xc($1)
      halt
```

Explicación estrategia

Partiendo del procesador base realizado en las prácticas se ha elaborado el anterior código buscando ser lo más eficiente posible. Para ello, se han introducido dos nuevas instrucciones al repertorio básico. En primer lugar, la instrucción ***lsl*** (logical shift left) de tipo-I, necesaria para realizar una multiplicación por cuatro con la que podremos situar la primera dirección de memoria del vector B. En segundo lugar, la instrucción ***subabs***, de tipo R, que realiza una resta y devuelve el valor absoluto del resultado en un solo ciclo.

Más allá de las nuevas instrucciones implementadas, tan solo queda comentar el código y el rendimiento. Antes de entrar en el bucle en $1 se guarda el tamaño de los vectores y en $2 y $3 la primera dirección de los vectores A y B respectivamente.

Una vez dentro del bucle, en cada iteración se cargan los valores correspondientes de ambos vectores, se ejecuta la resta, se suma el resultado a $6 y se actualizan $1, $2, $3 para la próxima iteración.

Contando los ciclos, tenemos 6 ciclos antes de comenzar, 9 ciclos en cada iteración del bucle, y un ciclo extra al final para almacenar el resultado en memoria. Teniendo una duración de ciclo de 16 ticks, podemos calcular el tiempo de ejecución en función de la longitud de los vectores con la expresión

$$Ciclos = 6 + 9 * vector\_size + 1$$

En el programa de prueba propuesto los vectores son de longitud 4 por lo que sería un total de 43 ciclos, o 688 ticks.

El último aspecto a considerar es que se ha asumido que el estado inicial de los registros es 0; en caso de que esta asunción no fuese cierta, sería necesario añadir una cantidad fija de instrucciones para la inicialización de los mismos.