Instrucciones

Tipo R

```
add $0, $1, $2
```

> Almacena en el registro `$0` la **suma** de `$1` y `$2`

Tipo I

```
lw $1, d($2) // d = 0 / 4 / 8 ...
```

> **Carga en el registro** `$1` la palabra almacenada en la dirección de memoria que contiene el registro `$2` más el **desplazamiento**. La dirección calculada debe ser múltiplo de 4.

```
sw $1, d($2)
```

> **Almacena en memoria** la palabra del registro `$1` en la dirección de memoria que contiene el registro `$2` más el **desplazamiento** . La dirección debe ser múltiplo de 4.

```
addi $1, $2, 3
```

> Almacena en el registro `$1` la **suma** del registro `$2` y la constante

```
beq $1, $2, tag
```

> Si el valor de `$1` y `$2` es igual se **modifica el valor del PC** para pasar a ejecutar el trozo de código apuntado por la etiqueta

```
bne $1, $2, tag
```

> Si el valor de `$1` y `$2` NO es igual se **modifica el valor del PC** para pasar a ejecutar el trozo de código apuntado por la etiqueta

Tipo J

```
j tag
```

> **Modifica el valor del PC** para ejecutar la instrucción siguiente a la etiqueta

Pseudoinstrucción `la`

```
.data
cadena: .asciiz "Hola mundo"

la $a0, cadena
```

> Guarda en `$a0` la dirección a la etiqueta (cadena). Es una pseudoinstrucción, se divide en:

```
lui $a0, 0x1001
ori $a0, $a0, 0x0000
```

Syscall

```
syscall
```

> Toma el tipo de llamada al sistema del registro `$v0` y los argumentos de los registros `$a0` y `$a1`. Por ejemplo

```
addi $v0, $0, 10 // Añadimos al registro $v0 el código de la instrucción
syscall
```

Código	Función
10	Finalizar la ejecución
1	Imprime como entero lo que se encuentre en `$a0`
4	Imprime como string lo que se encuentre en la dirección de memoriao indicada por `$a0`

Secciones

Las primeras líneas de la sección .text son obligatorias ya que indican el punto de inicio del programa:

```
.text
.globl main

main:  // A partir de esta llamada se encontrarían nuestras instrucciones
```

La sección `.data` contiene las variables del programa y es opcional. Estas se declaran de la siguiente forma:

```
.data
[ETIQUETA] : [TIPO] [INICIALIZACION]
  cadena:   .asciiz   "Hola mundo"
```

Tipo	Definición	Estructura de la definición
.ascii	Almacena en memoria el string como una lista de caracteres	variable: .ascii “string a almacenar”
.asciiz	Almacena en memoria el string como una lista de caracteres y lo termina con 0	variable: .asciiz “string a almacenar”
.word	Almacena la lista de palabras en posiciones secuenciales de memoria	variable: .word palabra1, palabra2, . . .
.space	Reserva **n** bytes de espacio en la memoria	variable: .space n
.float	Almacena la lista de números en punto flotante de simple precisión en posiciones sucesivas de memoria	variable: .float f1, f2, . . .
.double	Almacena la lista de números en punto flotante de doble precisión en posiciones sucesivas de memoria	variable: .double d1, d2, . . .

Simulador