Ejercicio 3 (Básico)

El siguiente código en alto nivel implementa el algoritmo de búsqueda binaria (bipartición) que dado una palabra ubicada en la dirección de memoria 100h y 100 palabras ordenadas en memoria a partir de la dirección 101h hasta la dirección 165h, determina si la palabra pertenece o no a la secuencia. En el caso de encontrar el elemento se escribe en la posición 166h el valor 1, en caso contrario escribe el valor 0.

```
void BBinaria () {

	int inf, sup, medio, dato;
	bool esta;

	/* Comienza a ejecutar la rutina */
	esta = false;
	inf = 0x101;
	sup = 0x165;
	dato = memoria[0x100];

	do {
		medio = (inf + sup) / 2;
		if (dato == (memoria[medio])) {
			esta = true;
		} else if (dato < memoria[medio]) {
			sup = medio - 1;
		} else {
			inf = medio + 1;
		}
	} while (!(esta || (inf > sup)));
	if (esta) {
		memoria[0x166]= 1;
	} else {
		memoria[0x166]= 0;
	}
}
```

SOLUCIÓN:

```
	SETHI	0x00 r1
	SETLO	0x00 r1		// r1 = esta = 0
	SETHI	0x01 r2
	SETLO	0x00 r2		// r2 = 0x0100
	SETHI	0x01 r3
```

```
        SETLO   0x65 r3         // r3 = 0x0165
        LOAD    r2 r4           // r4 = mem[0x100]
        SETHI   0x00 r8
        SETLO   0x01 r8         // r8 = 0x0001
        ADD     r2 r8 r2        // r2 = inf = 0x101
do:
        ADD     r2 r3 r5
        SR      r5 0x1 r5       // r5 = medio = (inf + sup)/2
        LOAD    r5 r6           // r6 = mem[medio]
        CMP     r4 r6           // dato - mem[medio]
        jz      esta_true       //
        jn      dato_menor      //
        sub     r5 r8 r3        // r3 = sup = medio - 1
        jmp     check_while
dato_mayor:
        add     r5 r8 r2        // r2 = inf = medio + 1
        jmp     check_while
esta_true:
        SETLO   0x01 r1         // r1 = esta = 1
        jmp     fin
check_while:
        CMP     r3 r2           // r3 - r2 = sup - inf
        jn      do
fin:
        SETHI   0x01 r6
        SETLO   0x66 r6         // r6 = 0x0166
        STORE   r1 r6           // mem[r6] = r1 = esta
```