

**MIPS** cheat sheet (including instructions you hadn't learned at the time)

Instruction	Syntax	Example
add	add dest, src0, src1	add \$s0, \$s1, \$s2
sub	sub dest, src0, src1	sub \$s0, \$s1, \$s2
addi	addi dest, src0, immediate	addi \$s0, \$s1, 12
sll / srl	sll dest, src, immediate	sll \$t0, 4(\$s0)
lw / lb	lw dest, offset(base addr)	lw \$t0, 4(\$s0)
sw / sb	sw src, offset(base addr)	sw \$t0, 4(\$s0)
bne	bne src0, src1, branchAddr	bne \$t0, \$t1, notEq
beq	beq src0, src1, branchAddr	bne \$t0, \$t1, Eq
j	j jumpAddr	j jumpWhenDone
jr	jr reg	jr \$ra

C	MIPS
<pre>// \$s0 -&gt; a (use \$s0 for a), \$s1 -&gt; b // \$s2 -&gt; c, \$s3 -&gt; z  int a=4, b=5, c=6, z; z = a+b+c+10;</pre>	
<pre>// \$s0 -&gt; int *p = (int *)malloc //                 (3*sizeof(int)); // \$s1 -&gt; a p[0] = 0; int a = 2; p[1] = a; p[a] = a;</pre>	
<pre>// \$s0 -&gt; a, \$s1 -&gt; b int a = 5, b = 10; if (a + a == b) {     a = 0; } else {     b = a - 1; }</pre>	
<pre>/*What does this do?    (Not C, in English) */</pre>	<pre>addi \$s0, \$0, 0 addi \$s1, \$0, 1 addi \$t0, \$0, 30 loop: beq \$s0, \$t0, done       add \$s1, \$s1, \$s1       addi \$s0, \$s0, 1       j loop done: # done!</pre>

<pre>// Strcpy: // \$s1 -&gt; char s1[] = "Hello!"; // \$s2 -&gt; char *s2 = //      malloc(sizeof(char)*7); int i=0; do{     s2[i] = s1[i];     i++; } while(s1[i]!='\0')</pre>	
<pre>// Nth_Fibonacci(N): // \$s0 -&gt; N, \$s1 -&gt; fib // \$t0 -&gt; i, \$t1 -&gt; j if(N==0) return 0; else if(N==1) return 1; N-=2; int fib=1, i=1, j=1; while(N!=0){     fib = i+j;     j = i;     i = fib;     N--; } return fib;</pre>	
	<pre>#0x100-0x104 valid addresses add \$s0, \$0, \$0 addi \$s1, \$0, 10 add \$s2, \$0, \$0 addi \$s3, \$0, 0x100 loop: beq \$s0, \$s1, done       addi \$t0, \$s0, 4       lw \$t0, 0(\$s3)       lw \$t1, 4(\$s3)       sw \$t1, 0(\$s3)       add \$s2, \$s2, \$t0       add \$s2, \$s2, \$t1       sw \$s2, 4(\$s0)       addi \$s0, \$s0, 1       j add add:  addi \$s0, \$s0, 1       j loop done: # done!</pre>
<p>Fill in the blanks in the MIPS code. Also add jump labels in appropriate places</p> <pre>// 0x100 -&gt; &amp;a, 0x200 -&gt; &amp;b // \$s0 -&gt; i  int a[4], b[4]; int i; for (i = 4; i != 0; i--) {     b[i] = a[i]; }</pre>	<pre>      addi \$s0, \$0, 4       beq \$s0, \$0, <b>done</b>       add \$t0, \$0, 4       addi \$t1, \$0, \$s0       addi \$t2, \$0, 0 do_mult: beq \$t0, \$0, <b>copy</b>           add \$t2, \$t2, \$t1           sub \$t0, \$t0, 1           j <b>do_mult</b> copy: lw \$t0, 0x100(\$t1)       sw</pre>

	<pre>      subi _____       j      loop done: # done!</pre>
--	---

Editor's note: I spent half an hour trying to get rid of this extra space. Tables in word are terrible.