```
# This is not a good example of a makefile.
# It's written this way to be clear what the ste
ps
# are to build a Cebollita application.

CC = java comp.parser
ASM = java asm.parser
LINK = java asm.Linker

expression: prologue-os.o
        ${CC} expression.c
        ${ASM} expression.s
        ${LINK} --output expression.exe prologue
-os.o expression.o

array: prologue-os.o
        ${CC} array.c
        ${ASM} array.s
        ${LINK} --output array.exe prologue-os.o
 array.o

arrayWithPrint: prologue-os.o iolib.o
        ${CC} arrayWithPrint.c
        ${ASM} arrayWithPrint.s
        ${LINK} --output arrayWithPrint.exe prol
ogue-os.o arrayWithPrint.o iolib.o

clean:
        rm -f a.exe *.o *.html *.exe *.sym *~
        rm -f array.s arrayWithPrint.s expressio
n.s iolib.s

iolib.o: iolib.s
        ${ASM} iolib.s

iolib.s: iolib.c
        ${CC} iolib.c

prologue-os.o: prologue-os.s
        ${ASM} prologue-os.s
```



```
# The Cebollita ISA simulator's loader has set t
hings up so that on entry:
#       $gp = size of text segment
#       $sp = text size + data size + heap size
+ stack size - 4
#       $pc = entry point (presumably __start he
re!)
.text
        .global __start
__start: jal    main
        ori     $v0, $0, 10             # exit
        syscall
```

```
int x = 10;
int y;
int z;

void main() {
  x = x + 2;
  y = 3*x - 6;
  z = y*y - (20*y + 3*x + 8);
}
```



```
.data
x:      .word 10
y:      .space 4
z:      .space 4
.text
#       ----------------------------------------
--------------
#       FuncDef for main([])
        .global main
main:   addi $sp, $sp, -8
        sw $ra, 0($sp)
        sw $fp, 4($sp)
        add $fp, $0, $sp
#       x = (+ x 2)
        ori $t0, $0, x
        add $t0, $t0, $gp
#       (+ x 2)
        ori $t1, $0, x
        add $t1, $t1, $gp
        lw $t1, 0($t1)
        addi $t2, $0, 2
        add $t3, $t1, $t2
        sw $t3, 0($t0)
#       y = (- (* 3 x) 6)
        ori $t0, $0, y
        add $t0, $t0, $gp
#       (- (* 3 x) 6)
#       (* 3 x)
        addi $t1, $0, 3
        ori $t2, $0, x
        add $t2, $t2, $gp
        lw $t2, 0($t2)
        mult $t3, $t1, $t2
        addi $t1, $0, 6
        sub $t2, $t3, $t1
        sw $t2, 0($t0)
#       z = (- (* y y) (+ (+ (* 20 y) (* 3 x)) 8
))
        ori $t0, $0, z
        add $t0, $t0, $gp
#       (- (* y y) (+ (+ (* 20 y) (* 3 x)) 8))
#       (* y y)
        ori $t1, $0, y
        add $t1, $t1, $gp
        lw $t1, 0($t1)
        ori $t2, $0, y
        add $t2, $t2, $gp
        lw $t2, 0($t2)
        mult $t3, $t1, $t2
#       (+ (+ (* 20 y) (* 3 x)) 8)
#       (+ (* 20 y) (* 3 x))
#       (* 20 y)
        addi $t1, $0, 20
        ori $t2, $0, y
        add $t2, $t2, $gp
        lw $t2, 0($t2)
        mult $t4, $t1, $t2
#       (* 3 x)
        addi $t1, $0, 3
        ori $t2, $0, x
        add $t2, $t2, $gp
        lw $t2, 0($t2)
        mult $t5, $t1, $t2
        add $t1, $t4, $t5
        addi $t2, $0, 8
        add $t4, $t1, $t2
        sub $t1, $t3, $t4
        sw $t1, 0($t0)
__L1:
        lw $ra, 0($fp)
        lw $fp, 4($fp)
        addi $sp, $sp, 8
        jr $ra
```

```
expression.o
  SymTabSize: 24
   {main:= [0x0], x:= [0x0], y:= [0x4], z:= [0x8
]}
  RefTabSize: 40
   {x:= [0x10, 0x18, 0x3c, 0x94], y:= [0x30, 0x6
0, 0x6c, 0x80], z:= [0x58]}
  DataSize: 12
  TextSize: 200
  00000000 0x23bdfff8: addi    $sp, $sp, -8
  00000004 0xafbf0000: sw      $ra, 0($sp)
  00000008 0xafbe0004: sw      $fp, 4($sp)
  0000000c 0x001df020: add     $fp, $0, $sp
  00000010 0x34080000: ori     $t0, $0, 0
  00000014 0x011c4020: add     $t0, $t0, $gp
  00000018 0x34090000: ori     $t1, $0, 0
  0000001c 0x013c4820: add     $t1, $t1, $gp
  00000020 0x8d290000: lw      $t1, 0($t1)
  00000024 0x200a0002: addi    $t2, $0, 2
  00000028 0x012a5820: add     $t3, $t1, $t2
  0000002c 0xad0b0000: sw      $t3, 0($t0)
  00000030 0x34080000: ori     $t0, $0, 0
  00000034 0x011c4020: add     $t0, $t0, $gp
  00000038 0x20090003: addi    $t1, $0, 3
  0000003c 0x340a0000: ori     $t2, $0, 0
  00000040 0x015c5020: add     $t2, $t2, $gp
  00000044 0x8d4a0000: lw      $t2, 0($t2)
  00000048 0x012a5818: mult    $t3, $t1, $t2
  0000004c 0x20090006: addi    $t1, $0, 6
  00000050 0x01695022: sub     $t2, $t3, $t1
  00000054 0xad0a0000: sw      $t2, 0($t0)
  00000058 0x34080000: ori     $t0, $0, 0
  0000005c 0x011c4020: add     $t0, $t0, $gp
  00000060 0x34090000: ori     $t1, $0, 0
  00000064 0x013c4820: add     $t1, $t1, $gp
  00000068 0x8d290000: lw      $t1, 0($t1)
  0000006c 0x340a0000: ori     $t2, $0, 0
  00000070 0x015c5020: add     $t2, $t2, $gp
  00000074 0x8d4a0000: lw      $t2, 0($t2)
  00000078 0x012a5818: mult    $t3, $t1, $t2
  0000007c 0x20090014: addi    $t1, $0, 20
  00000080 0x340a0000: ori     $t2, $0, 0
  00000084 0x015c5020: add     $t2, $t2, $gp
  00000088 0x8d4a0000: lw      $t2, 0($t2)
  0000008c 0x012a6018: mult    $t4, $t1, $t2
  00000090 0x20090003: addi    $t1, $0, 3
  00000094 0x340a0000: ori     $t2, $0, 0
  00000098 0x015c5020: add     $t2, $t2, $gp
  0000009c 0x8d4a0000: lw      $t2, 0($t2)
  000000a0 0x012a6818: mult    $t5, $t1, $t2
  000000a4 0x018d4820: add     $t1, $t4, $t5
  000000a8 0x200a0008: addi    $t2, $0, 8
  000000ac 0x012a6020: add     $t4, $t1, $t2
  000000b0 0x016c4822: sub     $t1, $t3, $t4
  000000b4 0xad090000: sw      $t1, 0($t0)
  000000b8 0x8fdf0000: lw      $ra, 0($fp)
  000000bc 0x8fde0004: lw      $fp, 4($fp)
  000000c0 0x23bd0008: addi    $sp, $sp, 8
  000000c4 0x03e00008: jr      $0, $ra, $0
```



```
Text size:   0x000000d4
Data size:   0x0000000c
Stack size:  0x00000800
Heap size:   0x00000200
Entry point: 0x00000000
00000000        0x0c000003: jal     0x3
00000004        0x3402000a: ori     $v0, $0, 10
00000008        0x0000000c: syscall $0, $0, $0
0000000c        0x23bdfff8: addi    $sp, $sp, -8
00000010        0xafbf0000: sw      $ra, 0($sp)
00000014        0xafbe0004: sw      $fp, 4($sp)
00000018        0x001df020: add     $fp, $0, $sp
0000001c        0x34080000: ori     $t0, $0, 0
00000020        0x011c4020: add     $t0, $t0, $g
p
00000024        0x34090000: ori     $t1, $0, 0
00000028        0x013c4820: add     $t1, $t1, $g
p
0000002c        0x8d290000: lw      $t1, 0($t1)
00000030        0x200a0002: addi    $t2, $0, 2
00000034        0x012a5820: add     $t3, $t1, $t
2
00000038        0xad0b0000: sw      $t3, 0($t0)
0000003c        0x34080004: ori     $t0, $0, 4
00000040        0x011c4020: add     $t0, $t0, $g
p
00000044        0x20090003: addi    $t1, $0, 3
00000048        0x340a0000: ori     $t2, $0, 0
0000004c        0x015c5020: add     $t2, $t2, $g
p
00000050        0x8d4a0000: lw      $t2, 0($t2)
00000054        0x012a5818: mult    $t3, $t1, $t
2
00000058        0x20090006: addi    $t1, $0, 6
0000005c        0x01695022: sub     $t2, $t3, $t
1
00000060        0xad0a0000: sw      $t2, 0($t0)
00000064        0x34080008: ori     $t0, $0, 8
00000068        0x011c4020: add     $t0, $t0, $g
p
0000006c        0x34090004: ori     $t1, $0, 4
00000070        0x013c4820: add     $t1, $t1, $g
p
00000074        0x8d290000: lw      $t1, 0($t1)
00000078        0x340a0004: ori     $t2, $0, 4
0000007c        0x015c5020: add     $t2, $t2, $g
p
00000080        0x8d4a0000: lw      $t2, 0($t2)
00000084        0x012a5818: mult    $t3, $t1, $t
2
00000088        0x20090014: addi    $t1, $0, 20
0000008c        0x340a0004: ori     $t2, $0, 4
00000090        0x015c5020: add     $t2, $t2, $g
p
00000094        0x8d4a0000: lw      $t2, 0($t2)
00000098        0x012a6018: mult    $t4, $t1, $t
2
0000009c        0x20090003: addi    $t1, $0, 3
000000a0        0x340a0000: ori     $t2, $0, 0
000000a4        0x015c5020: add     $t2, $t2, $g
p
000000a8        0x8d4a0000: lw      $t2, 0($t2)
000000ac        0x012a6818: mult    $t5, $t1, $t
2
000000b0        0x018d4820: add     $t1, $t4, $t
5
000000b4        0x200a0008: addi    $t2, $0, 8
000000b8        0x012a6020: add     $t4, $t1, $t
2
000000bc        0x016c4822: sub     $t1, $t3, $t
4
000000c0        0xad090000: sw      $t1, 0($t0)
000000c4        0x8fdf0000: lw      $ra, 0($fp)
000000c8        0x8fde0004: lw      $fp, 4($fp)
000000cc        0x23bd0008: addi    $sp, $sp, 8
000000d0        0x03e00008: jr      $0, $ra, $0
```

```
int vec[8];

int i;
int sum;

void main() {

  i = 0;
  while ( i<8 ) {
    vec[i] = i;
    i = i+1;
  }

  sum=0;
  i = 0;
  while ( i<8 ) {
    sum = sum + vec[i];
    i = i+1;
  }
}
```



```
.data
vec:    .space 36
i:      .space 4
sum:    .space 4
.text
#       ----------------------------------------
--------------
#       FuncDef for main([])
        .global main
main:   addi $sp, $sp, -8
        sw $ra, 0($sp)
        sw $fp, 4($sp)
        add $fp, $0, $sp
#       i = 0
        ori $t0, $0, i
        add $t0, $t0, $gp
        addi $t1, $0, 0
        sw $t1, 0($t0)
#       while ((< i 8))
#       (< i 8)
        ori $t0, $0, i
        add $t0, $t0, $gp
        lw $t0, 0($t0)
        addi $t1, $0, 8
        slt $t2, $t0, $t1
        beq $0, $t2, __L3
__L2:
#       vec[i] = i
        ori $t0, $0, i
        add $t0, $t0, $gp
        lw $t0, 0($t0)
        addi $at, $0, 2
        sllv $t0, $t0, $at
        ori $t1, $0, vec
        add $t1, $t1, $gp
        add $t1, $t1, $t0
        ori $t0, $0, i
        add $t0, $t0, $gp
        lw $t0, 0($t0)
        sw $t0, 0($t1)
#       i = (+ i 1)
        ori $t0, $0, i
        add $t0, $t0, $gp
#       (+ i 1)
        ori $t1, $0, i
        add $t1, $t1, $gp
        lw $t1, 0($t1)
        addi $t3, $0, 1
        add $t4, $t1, $t3
        sw $t4, 0($t0)
#       (< i 8)
        ori $t0, $0, i
        add $t0, $t0, $gp
        lw $t0, 0($t0)
        addi $t1, $0, 8
        slt $t3, $t0, $t1
        bne $0, $t3, __L2
__L3:
#       sum = 0
        ori $t0, $0, sum
        add $t0, $t0, $gp
        addi $t1, $0, 0
        sw $t1, 0($t0)
#       i = 0
        ori $t0, $0, i
        add $t0, $t0, $gp
        addi $t1, $0, 0
        sw $t1, 0($t0)
#       while ((< i 8))
#       (< i 8)
        ori $t0, $0, i
        add $t0, $t0, $gp
        lw $t0, 0($t0)
        addi $t1, $0, 8
        slt $t2, $t0, $t1
        beq $0, $t2, __L5
__L4:
#       sum = (+ sum vec[i])
        ori $t0, $0, sum
        add $t0, $t0, $gp
#       (+ sum vec[i])
        ori $t1, $0, sum
        add $t1, $t1, $gp
        lw $t1, 0($t1)
```

```
        ori $t3, $0, i
        add $t3, $t3, $gp
        lw $t3, 0($t3)
        addi $at, $0, 2
        sllv $t3, $t3, $at
        ori $t4, $0, vec
        add $t4, $t4, $gp
        add $t4, $t4, $t3
        lw $t4, 0($t4)
        add $t3, $t1, $t4
        sw $t3, 0($t0)
#       i = (+ i 1)
        ori $t0, $0, i
        add $t0, $t0, $gp
#       (+ i 1)
        ori $t1, $0, i
        add $t1, $t1, $gp
        lw $t1, 0($t1)
        addi $t3, $0, 1
        add $t4, $t1, $t3
        sw $t4, 0($t0)
#       (< i 8)
        ori $t0, $0, i
        add $t0, $t0, $gp
        lw $t0, 0($t0)
        addi $t1, $0, 8
        slt $t3, $t0, $t1
        bne $0, $t3, __L4
__L5:
__L1:
        lw $ra, 0($fp)
        lw $fp, 4($fp)
        addi $sp, $sp, 8
        jr $ra
```



```
int vec[8];

int i;
int sum;

void main() {

  i = 0;
  while ( i<8 ) {
    vec[i] = i;
    i = i+1;
  }

  sum=0;
  i = 0;
  while ( i<8 ) {
    sum = sum + vec[i];
    i = i+1;
  }

  printString("Sum is ");
  printInt( sum );
}
```

```
.data
__L1:    .asciiz "Sum is "
vec:    .space 36
i:      .space 4
sum:    .space 4
.text
#       ----------------------------------------
--------------
#       FuncDef for main([])
        .global main
main:   addi $sp, $sp, -8
        sw $ra, 0($sp)
        sw $fp, 4($sp)
        add $fp, $0, $sp
#       i = 0
        ori $t0, $0, i
        add $t0, $t0, $gp
        addi $t1, $0, 0
        sw $t1, 0($t0)
#       while ((< i 8))
#       (< i 8)
        ori $t0, $0, i
        add $t0, $t0, $gp
        lw $t0, 0($t0)
        addi $t1, $0, 8
        slt $t2, $t0, $t1
        beq $0, $t2, __L4
__L3:
#       vec[i] = i
        ori $t0, $0, i
        add $t0, $t0, $gp
        lw $t0, 0($t0)
        addi $at, $0, 2
        sllv $t0, $t0, $at
        ori $t1, $0, vec
        add $t1, $t1, $gp
        add $t1, $t1, $t0
        ori $t0, $0, i
        add $t0, $t0, $gp
        lw $t0, 0($t0)
        sw $t0, 0($t1)
#       i = (+ i 1)
        ori $t0, $0, i
        add $t0, $t0, $gp
#       (+ i 1)
        ori $t1, $0, i
        add $t1, $t1, $gp
        lw $t1, 0($t1)
        addi $t3, $0, 1
        add $t4, $t1, $t3
        sw $t4, 0($t0)
#       (< i 8)
        ori $t0, $0, i
        add $t0, $t0, $gp
        lw $t0, 0($t0)
        addi $t1, $0, 8
        slt $t3, $t0, $t1
        bne $0, $t3, __L3
__L4:
#       sum = 0
        ori $t0, $0, sum
        add $t0, $t0, $gp
        addi $t1, $0, 0
        sw $t1, 0($t0)
#       i = 0
        ori $t0, $0, i
        add $t0, $t0, $gp
        addi $t1, $0, 0
        sw $t1, 0($t0)
#       while ((< i 8))
#       (< i 8)
        ori $t0, $0, i
        add $t0, $t0, $gp
        lw $t0, 0($t0)
        addi $t1, $0, 8
        slt $t2, $t0, $t1
        beq $0, $t2, __L6
__L5:
#       sum = (+ sum vec[i])
        ori $t0, $0, sum
        add $t0, $t0, $gp
#       (+ sum vec[i])
        ori $t1, $0, sum
        add $t1, $t1, $gp
```



```
        lw $t1, 0($t1)
        ori $t3, $0, i
        add $t3, $t3, $gp
        lw $t3, 0($t3)
        addi $at, $0, 2
        sllv $t3, $t3, $at
        ori $t4, $0, vec
        add $t4, $t4, $gp
        add $t4, $t4, $t3
        lw $t4, 0($t4)
        add $t3, $t1, $t4
        sw $t3, 0($t0)
#       i = (+ i 1)
        ori $t0, $0, i
        add $t0, $t0, $gp
#       (+ i 1)
        ori $t1, $0, i
        add $t1, $t1, $gp
        lw $t1, 0($t1)
        addi $t3, $0, 1
        add $t4, $t1, $t3
        sw $t4, 0($t0)
#       (< i 8)
        ori $t0, $0, i
        add $t0, $t0, $gp
        lw $t0, 0($t0)
        addi $t1, $0, 8
        slt $t3, $t0, $t1
        bne $0, $t3, __L5
__L6:
#       printString(["Sum is "])
        addi $sp, $sp, -4
#       ""Sum is ""
        ori $t0, $0, __L1
        add $t0, $t0, $gp
        sw $t0, 0($sp)
        jal printString
        addi $sp, $sp, 4
        add $t0, $0, $v0
#       printInt([sum])
        addi $sp, $sp, -4
        ori $t0, $0, sum
        add $t0, $t0, $gp
        lw $t0, 0($t0)
        sw $t0, 0($sp)
        jal printInt
        addi $sp, $sp, 4
        add $t0, $0, $v0
__L2:
        lw $ra, 0($fp)
        lw $fp, 4($fp)
        addi $sp, $sp, 8
        jr $ra
```