COMP.2030R Lab 3 9/27/22

1. Suppose you have an integer array A with N elements (A is global). The following function select\_sort sorts A using the selection sort algorithm. Convert the following to Pseudo-C code with (un)conditional gotos in place of the for and if statements.

void select\_sort (int N)

{

int i, j, min, max, tmp;

for (i = 0; i <= N / 2; i++) {

min = i;

max = i;

for (j = i+1; j < N-i; j++) {

if (A[j] > A[max]) {

max = j;

continue;

}

if (A[j] < A[min]) {

min = j;

}

}

tmp = A[i];

A[i] = A[min];

A[min] = tmp;

tmp = A[N–1 - i];

A[N–1 - i] = A[max];

A[max] = tmp;

}

}

void select\_sort (int N)

{

int i, j, min, max, tmp;

i = 0;

loop:

if(i > N / 2) goto exit

min = i;

max = i;

j = i+1;

loop2:

if (j >= N – i) goto next

if (A[j] <= A[max]) goto chkmin

max = j;

continue;

chkmin:

if (A[j] >= A[min]) goto chkmax

min = j;

chkmax:

j++;

goto loop2

next:

tmp = A[i];

A[i] = A[min];

A[min] = tmp;

tmp = A[N-1 - i];

A[N-1 - i] = A[max];

A[max] = tmp;

i++;

goto loop

exit:

}

1. Convert the Pseudo-C code from part 1 into MIPS.

select\_sort: # N in $a0

# $t0, $t1, $s0, $s1, $t9 for int i, j, min, max, tmp;

li $t0, 0 # i = 0;

loop:

sra $a1, $a0, 1 # N/2 in $s1

bgt $t0, $s1, exit # if( i > N / 2) goto exit

move $s0, $t0 # min = i;

move $s1, $t0 # max = i;

addi $t1, $t0, 1 # j = i+1;

loop2:

sub $t3, $a0, $t0 #N – i in $t3

bge $t2, $t3, next # if (j >= N – i) goto next

sll $t1, $t1, 2 # from C index to byte offset

sll $s1, $s1, 2

lw $v0, A($t1) # A[j] in $v0

lw $v1, A($s1) # A[max] in $v1

sra $t1, $t1, 2 # from byte offset to C index

sra $s1, $s1, 2

ble $v0, $v1, chkmin #x if (A[j] <= A[max]) goto chkmin

move $s1, $t1 # max = j;

# continue;

chkmin:

sll $s0, $s0, 2 # from C index to byte offset

lw $v1, A($s1) # A[min] in $v1

sra $s0, $s0, 2

bge $v0, $v1, chkmax # if (A[j] >= A[min]) goto chkmax

move $s0, $t1 # min = j;

chkmax:

addi $t1, $t1, 1 # j++;

b loop2 # goto loop2

next:

sll $t0,$t0,2 # C index to byte offset

addi $t0,$t0,-4 # index i-1

lw $v0, A($t0) # tmp = A[i - 1];

sll $s0, $s0, 2 # from C index to byte offset

lw $v1, A($s1) # A[min] in $v1

sw $v1, A($t0) # A[i - 1] = A[min];

sw $v0, A(s1) # A[min] = tmp;

sra $t0, $t0, 2

sra $s0, $s0, 2

sub $t3, $a0, $t0 # N-i in $t3

subi $t3, $t3, 1 # N-1-I in $t3

sll $t3, $t3, 2 # to byte offset

lw $v0, A($t3) # tmp = A[N-1 - i];

sll $s1, $s1, 2 # byte offset

lw $v1, A($s1) # A[max] in $v1

sw $v1, A($t3) # A[N-1 - i] = A[max];

sw $v0, A($s1) # A[max] = tmp;

sra $s1, $s1, 2

addi $t0, $t0, 1 # i++;

b loop # goto loop

exit:

jr $ra