###############################################################################

# Title: Assign02P3 Author: Samuel Wait

# Class: CS 2318-253 Spring 2023 Submitted: 4/22/2023

###############################################################################

# Program: MIPS translation of a given C++ program

###############################################################################

# Pseudocode description: supplied a2p2\_SampSoln.cpp

###############################################################################

.data

a1: .space 48

a2: .space 48

a3: .space 48

einStr: .asciiz "Enter integer #"

moStr: .asciiz "Max of "

ieStr: .asciiz " ints entered..."

emiStr: .asciiz "Enter more ints? (n or N = no, others = yes)"

begA1Str: .asciiz "beginning a1: "

am1dA1Str: .asciiz "a1 (dups<=1): "

procA1Str: .asciiz "processed a1: "

procA2Str: .asciiz " a2: "

procA3Str: .asciiz " a3: "

dacStr: .asciiz "Do another case? (n or N = no, others = yes) "

d1Str: .asciiz "================================"

byeStr: .asciiz "bye..."

.text

.globl main

main:

################################################

# Register usage:

#################

# $a0: short-lived holder (to locally comment)

# $a1: endPtr1

# $a2: endPtr2

# $a3: endPtr3

# $t0: endPtr11

# $t1: used1

# $t2: used2

# $t3: used3

# $t4: hopPtr1

# $t5: hopPtr2

# $t6: hopPtr11

# $t7: hopPtr3

# $t8: reply or sum (non-overlappingly)

# $t9: found or truncAvg (non-overlappingly)

# $v0: short-lived holder (to locally comment)

# $v1: short-lived holder (to locally comment)

################################################

#reply = 'y';

li $t8, 'y'

#while (reply != 'n' && reply != 'N')

j WTest1

begW1:# {

#used1 = 0;

li $t1, 0

#hopPtr1 = a1;

la $t4, a1

#while (reply != 'n' && reply != 'N')

j WTest2

begW2:# {

#cout << endl;

li $v0, 11

li $a0, '\n'

syscall

#cout << einStr;

li $v0, 4

la $a0, einStr

syscall

#cout << (used1 + 1);

li $v0, 1

addi $a0, $t1, 1

syscall

#cout << ':' << ' ';

li $v0, 11

li $a0, ':'

syscall

li $v0, 11

li $a0, ' '

syscall

#cin >> \*hopPtr1;

li $v0, 5

syscall

sw $v0, 0($t4)

#++used1;

addi $t1, $t1, 1

#++hopPtr1;

addi $t4, $t4, 4

#cout << endl;

li $v0, 11

li $a0, '\n'

syscall

#//if (used1 < 12)

#if ((used1 >= 12)) goto else1;

li $v1, 12

bge $t1, $v1, else1

begI1:# {

#cout << emiStr;

li $v0, 4

la $a0, emiStr

syscall

#cin >> reply;

li $v0, 12

syscall

move $t8, $v0

j endI1

#// }

else1:#//

#else

#// {

#cout << moStr << 12 << ieStr << endl;

li $v0, 4

la $a0, moStr

syscall

li $v0, 1

li $a0, 12

syscall

li $v0, 4

la $a0, ieStr

syscall

li $v0, 11

li $a0, '\n'

syscall

#reply = 'n';

li $t8, 'n'

endI1:#// }

#//WTest2: if (reply != 'n' && reply != 'N') goto begW2;

WTest2: #if (reply == 'n') goto xitW2;

li $v1, 'n'

beq $t8, $v1, xitW2

#if (reply != 'N') goto begW2;

li $v1, 'N'

bne $t8, $v1, begW2

endW2:#// }

xitW2:

#cout << begA1Str;

li $v0, 4

la $a0, begA1Str

syscall

#//if (used1 > 0)

#if (used1 <= 0) goto endI2;

li $v1, 0

ble $t1, $v1, endI2

begI2:#// {

#hopPtr1 = a1;

la $t4, a1

#endPtr1 = a1 + used1;

sll $v1, $t1, 2

add $a1, $v1, $t4

#//do

begDW1:#// {

#cout << \*hopPtr1 << ' ' << ' ';

li $v0, 1

lw $a0, 0($t4)

syscall

li $v0, 11

li $a0, ' '

syscall

syscall

#++hopPtr1;

addi $t4, $t4, 4

endDW1:#// }

#//while (hopPtr1 < endPtr1);

DWTest1: #if (hopPtr1 < endPtr1) goto begDW1;

blt $t4, $a1, begDW1

endI2:#// }

#cout << endl;

li $v0, 11

li $a0, '\n'

syscall

#//if (used1 > 1)

#if (used1 <= 1) goto else3;

li $v1, 1

ble $t1, $v1, else3

begI3:#// {

#hopPtr1 = a1;

la $t4, a1

#endPtr1 = a1 + used1 - 1;

sll $a1, $t1, 2

add $a1, $a1, $t4

li $v1, -4

add $a1, $a1, $v1

#//while (hopPtr1 < endPtr1)

j WTest3

begW3:#// {

#found = 0;

li $t9, 0

#endPtr2 = a1 + used1;

la $v0, a1

sll $v1, $t1, 2

add $a2, $v1, $v0

#//for (hopPtr2 = hopPtr1 + 1; hopPtr2 < endPtr2; ++hopPtr2)

#hopPtr2 = hopPtr1 + 1;

li $v1, 4

add $t5, $t4, $v1

j FTest1

begF1:#// {

#//if (\*hopPtr2 == \*hopPtr1)

#if (\*hopPtr2 != \*hopPtr1) goto endI4;

lw $v0, 0($t5)

lw $v1, 0($t4)

bne $v0, $v1, endI4

begI4:#// {

#//if (found == 1)

#if (found != 1) goto else5;

li $v1, 1

bne $t9, $v1, else5

begI5:#// {

#endPtr3 = a1 + used1;

la $v0, a1

sll $a3, $t1, 2

add $a3, $a3, $v0

#//for (hopPtr3 = hopPtr2 + 1; hopPtr3 < endPtr3; ++hopPtr3)

#hopPtr3 = hopPtr2 + 1;

li $v1, 4

add $t7, $t5, $v1

j FTest2

begF2:#// {

#\*(hopPtr3 - 1) = \*hopPtr3;

lw $v1, 0($t7)

sw $v1, -4($t7)

#++hopPtr3;

li $v0, 4

add $t7, $t7, $v0

FTest2: #if (hopPtr3 < endPtr3) goto begF2;

blt $t7, $a3, begF2

endF2:#// }

#--used1;

addi $t1, $t1, -1

#--endPtr1;

addi $a1, $a1, -4

#--endPtr2;

addi $a2, $a2, -4

#--endPtr3;

addi $a3, $a3, -4

#--hopPtr2;

addi $t5, $t5, -4

j endI5

#// }

else5:#// else

#// {

#++found;

addi $t9, $t9, 1

endI5:#// }

endI4:#// }

#++hopPtr2;

addi $t5, $t5, 4

FTest1: #if (hopPtr2 < endPtr2) goto begF1;

blt $t5, $a2, begF1

endF1:#// }

#++hopPtr1;

addi $t4, $t4, 4

WTest3: #if (hopPtr1 < endPtr1) goto begW3;

blt $t4, $a1, begW3

endW3:#// }

#cout << am1dA1Str;

li $v0, 4

la $a0, am1dA1Str

syscall

#//if (used1 > 0)

#if (used1 <= 0) goto endI6;

blez, $t1, endI6

begI6:#// {

#hopPtr1 = a1;

la $t4, a1

#endPtr1 = a1 + used1;

sll $a1, $t1, 2

add $a1, $a1, $t4

#// do

begDW2:#// {

#cout << \*hopPtr1 << ' ' << ' ';

li $v0, 1

lw $a0, 0($t4)

syscall

li $v0, 11

li $a0, ' '

syscall

syscall

#++hopPtr1;

addi $t4, $t4, 4

endDW2:#// }

#//while (hopPtr1 < endPtr1);

DWTest2: #if (hopPtr1 < endPtr1) goto begDW2;

blt $t4, $a1, begDW2

endI6:#// }

#cout << endl;

li $v0, 11

li $a0, '\n'

syscall

#//if (used1 > 0)

#if (used1 <= 0) goto endI7;

blez $t1, endI7

begI7:#// {

#sum = 0;

li $t8, 0

#hopPtr1 = a1 + used1 - 1;

la $t4, a1

sll $v1,$t1,2

add $t4, $t4, $v1

li $v0, -4

add $t4, $t4, $v0

#endPtr1 = a1;

la $a1, a1

#// do

begDW3:#// {

#sum += \*hopPtr1;

lw $v1, 0($t4)

add $t8, $t8, $v1

#--hopPtr1;

li $v0, -4

add $t4, $t4, $v0

endDW3:#// }

#//while (hopPtr1 >= endPtr1);

DWTest3: #if (hopPtr1 >= endPtr1) goto begDW3;

bge $t4, $a1, begDW3

#truncAvg = sum / used1

div $t8, $t1

mflo $t9

#used2 = 0;

li $t2, 0

#used3 = 0;

li $t3, 0

#hopPtr2 = a2;

la $t5, a2

#hopPtr3 = a3;

la $t7, a3

#endPtr1 = a1 + used1;

sll $a1, $t1, 2

la $v1, a1

add $a1, $a1, $v1

#//for (hopPtr1 = a1; hopPtr1 < endPtr1; ++hopPtr1)

#hopPtr1 = a1;

la $t4, a1

j FTest3

begF3:#// {

#//if (\*hopPtr1 != truncAvg)

#if (\*hopPtr1 == truncAvg) goto endI8;

lw $v0, 0($t4)

beq $v0, $t9, endI8

begI8:#// {

#//if (\*hopPtr1 < truncAvg)

#if (\*hopPtr1 >= truncAvg) goto else9;

lw $v0, 0($t4)

bge $v0, $t9, else9

begI9:#// {

#\*hopPtr2 = \*hopPtr1;

lw $v0, 0($t4)

sw $v0, 0($t5)

#++used2;

addi $t2, $t2, 1

#++hopPtr2;

addi $t5, $t5, 4

j endI9

#// }

else9:#// else

#// {

#\*hopPtr3 = \*hopPtr1;

lw $v0, 0($t4)

sw $v0, 0($t7)

#++used3;

addi $t3, $t3, 1

#++hopPtr3;

addi $t7, $t7, 4

endI9:#// }

#endPtr11 = a1 + used1;

sll $t0, $t1, 2

la $v1, a1

add $t0, $t0, $v1

#//for (hopPtr11 = hopPtr1 + 1; hopPtr11 < endPtr11; ++hopPtr11)

#hopPtr11 = hopPtr1 + 1;

li $v0, 4

add $t6, $t4, $v0

j FTest4

begF4:#// {

#\*(hopPtr11 - 1) = \*hopPtr11;

lw $v0, 0($t6)

sw $v0, -4($t6)

#++hopPtr11;

addi $t6, $t6, 4

FTest4: #if (hopPtr11 < endPtr11) goto begF4;

blt $t6, $t0, begF4

endF4:#// }

#--used1;

addi $t1, $t1, -1

#--endPtr1;

addi $a1, $a1, -4

#--hopPtr1;

addi $t4, $t4, -4

endI8:#// }

#++hopPtr1;

addi $t4, $t4, 4

FTest3: #if (hopPtr1 < endPtr1) goto begF3;

blt $t4, $a1, begF3

endF3:#// }

#//if (used1 == 0)

#if (used1 != 0) goto endI10;

bnez $t1, endI10

begI10:#// {

#\*(a1+ 0) = truncAvg;

la $v1, a1

sw $t9, 0($v1)

#++used1;

addi $t1, $t1, 1

endI10:#// }

endI7:#// }

j endI3

#// }

else3:#// else

#// {

#hopPtr1 = a1;

la $t4, a1

#cout << am1dA1Str;

li $v0, 4

la $a0, am1dA1Str

syscall

#cout << \*hopPtr1;

li $v0, 1

lw $a0, 0($t4)

syscall

#cout << endl;

li $v0, 11

li $a0, '\n'

syscall

#used2 = 0;

li $t2, 0

#used3 = 0;

li $t3, 0

endI3:#// }

#cout << procA1Str;

li $v0, 4

la $a0, procA1Str

syscall

#//if (used1 > 0)

#if (used1 <= 0) goto endI11;

blez $t1, endI11

begI11:#// {

#hopPtr1 = a1;

la $t4, a1

#endPtr1 = a1 + used1;

sll $a1, $t1, 2

add $a1, $a1, $t4

#// do

begDW4:#// {

#cout << \*hopPtr1 << ' ' << ' ';

li $v0, 1

lw $a0, 0($t4)

syscall

li $v0, 11

li $a0, ' '

syscall

syscall

#++hopPtr1;

addi $t4, $t4, 4

endDW4:#// }

#//while (hopPtr1 < endPtr1);

DWTest4: #if (hopPtr1 < endPtr1) goto begDW4;

blt $t4, $a1, begDW4

endI11:#// }

#cout << endl;

li $v0, 11

li $a0, '\n'

syscall

#cout << procA2Str;

li $v0, 4

la $a0, procA2Str

syscall

#//if (used2 > 0)

#if (used2 <= 0) goto endI12;

blez $t2, endI12

begI12:#// {

#hopPtr2 = a2;

la $t5, a2

#endPtr2 = a2 + used2;

sll $a2, $t2, 2

add $a2, $a2, $t5

#// do

begDW5:#// {

#cout << \*hopPtr2 << ' ' << ' ';

li $v0, 1

lw $a0, 0($t5)

syscall

li $v0, 11

li $a0, ' '

syscall

syscall

#++hopPtr2;

addi $t5, $t5, 4

endDW5:#// }

#//while (hopPtr2 < endPtr2);

DWTest5: #if (hopPtr2 < endPtr2) goto begDW5;

blt $t5, $a2, begDW5

endI12:#// }

#cout << endl;

li $v0, 11

li $a0, '\n'

syscall

#cout << procA3Str;

li $v0, 4

la $a0, procA3Str

syscall

#//if (used3 > 0)

#if (used3 <= 0) goto endI13;

blez, $t3, endI13

begI13:#// {

#hopPtr3 = a3;

la $t7, a3

#endPtr3 = a3 + used3;

sll $a3, $t3, 2

add $a3, $a3, $t7

#do

begDW6:# {

#cout << \*hopPtr3 << ' ' << ' ';

li $v0, 1

lw $a0, 0($7)

syscall

li $v0, 11

li $a0, ' '

syscall

syscall

#++hopPtr3;

addi $t7, $t7, 4

endDW6:# }

#//while (hopPtr3 < endPtr3);

DWTest6: #if (hopPtr3 < endPtr3) goto begDW6;

blt $t7, $a3, begDW6

endI13:# }

#cout << endl;

li $v0, 11

li $a0, '\n'

syscall

#cout << dacStr;

li $v0, 4

la $a0, dacStr

syscall

#cin >> reply;

li $v0, 8

syscall

move $t8, $v0

#//WTest1: if (reply != 'n' && reply != 'N') goto begW1;

WTest1:

#if (reply == 'n') goto xitW1;

li $v1, 'n'

beq $t8, $v1, xitW1

#if (reply != 'N') goto begW1;

li $v1, 'N'

bne $t8, $v1, begW1

endW1: #}

xitW1:

#cout << d1Str << '\n';

li $v0, 4

la $a0, d1Str

syscall

li $v0, 11

li $a0, '\n'

syscall

#cout << byeStr << '\n';

li $v0, 4

la $a0, byeStr

syscall

li $v0, 11

li $a0, '\n'

syscall

#cout << d1Str << '\n';

li $v0, 4

la $a0, d1Str

syscall

li $v0, 11

li $a0, '\n'

syscall

#Exit

li $v0, 10

syscall