```
################################################################################
# Title: Assign02P3                     Author: Ronaldo A Amaya
# Class: CS 2318-003, Fall 2020         Submitted: 11/5/20
################################################################################
# Program: MIPS tranlation of a given C++ program
################################################################################
# Pseudocode description: supplied a2p2_SampSoln.cpp
################################################################################

#int a1[12],
#    a2[12],
#    a3[12];
#char einStr[]    = "Enter integer #";
#char moStr[]     = "Max of ";
#char ieStr[]     = " ints entered...";
#char emiStr[]    = "Enter more ints? (n or N = no, others = yes) ";
#char begA1Str[]  = "beginning a1: ";
#char nn09A1Str[] = "a1 (noneg09): ";
#char procA1Str[] = "processed a1: ";
#char procA2Str[] = "          a2: ";
#char procA3Str[] = "          a3: ";
#char dacStr[]    = "Do another case? (n or N = no, others = yes) ";
#char dlStr[]     = "================================";
#char byeStr[]    = "bye...";



                  .data
a1:               .space 48
a2:               .space 48
a3:               .space 48
einStr:           .asciiz "\nEnter integer #"
moStr:            .asciiz "Max of "
ieStr:            .asciiz " ints entered..."
emiStr:           .asciiz "Enter more ints? (n or N = no, others = yes) "
begA1Str:         .asciiz "beginning a1: "
nn09A1Str:        .asciiz "a1 (noneg09): "
procA1Str:        .asciiz "processed a1: "
procA2Str:        .asciiz "          a2: "
procA3Str:        .asciiz "          a3: "
dacStr:           .asciiz "\nDo another case? (n or N = no, others = yes) "
dlStr:            .asciiz "\n================================"
byeStr:           .asciiz "bye..."
# int main()
#{
                  .text
                  .globl main
main:

#####################################################
# Register usage:
#################
# $a0: extra short-lived holder (as locally commented)
# $a1: endPtr1
# $a2: endPtr2
# $a3: endPtr3
# $t0: target
# $t1: used1
# $t2: used2
# $t3: used3
# $t4: hopPtr1
# $t5: hopPtr2
# $t6: hopPtr11
# $t7: hopPtr3
# $t8: reply or intHolder or iter (non-overlappingly)
# $t9: count
# $v0: extra short-lived holder (as locally commented)
# $v1: short-lived holder (as locally commented)
#######################################################



           #reply = 'y';
                   li $t8, 'y'
#            //while (reply != 'n' && reply != 'N')
#            goto WTest1;
                   j WTest1
begW1:#//    {
#               used1 = 0;
                   li $t1, 0        # used1 = 0
#               hopPtr1 = a1;
                   la $t4, a1
#               //while (reply != 'n' && reply != 'N')
#               goto WTest2;
                   j WTest2
begW2:#//       {
#                  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 $a0, ' '
                   syscall
#                  cin >> *hopPtr1;
                   li $v0, 5
                   syscall
                   sw $v0, 0($t4)
#                  ++used1;
                   addi $t1, $t1, 1
#                  ++hopPtr1;
                   addi $t4, $t4, 4
#                 ///if (used1 < 12)
#                 if (used1 >= 12) goto else1;
                   li $v1, 12
                   bge $t1 , $v1,  e# short-lived holder & bge = branch greater than equal
begI1:#//         {
#                    cout << emiStr;
                   li $v0,  4
                   la $a0, emiStr
                   syscall
#                    cin >> reply;
                   li $v0, 12
                   syscall
                   move $t8, $v0
#                 goto endI1;
                   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:#//         }
#                 ///;
endW2:#//      }
WTest2:
#              ///if (reply != 'n' && reply != 'N') goto begW2;
#            if (reply == 'n') goto xitW2;
                   li $v1, 'n'
                   beq $v1, $t8, xitW2
#             if (reply != 'N') goto begW2;
                   li $v1, 'N'
                   bne $t8, $v1, begW2
xitW2:

#              cout << begA1Str;
                   li $v0, 4
                   la $a0, begA1Str
                   syscall

#              ///if (used1 > 0)
#              if (used1 <= 0) goto endI2;
```

```
                   ble $t1, $0, endI2
begI2:#//      {
#                 hopPtr1 = a1;
                 la $t4, a1
#                 endPtr1 = a1 + used1;
                  sll $v0, $t1, 2
                  add $a1, $t4, $v0
#                 //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 > 0)
#              if (used1 <= 0) goto endI3;
                  ble $t1, $0, endI3

begI3:#//      {
#                 //for (hopPtr1 = a1, endPtr1 = a1 + used1;  // multi-init
#                 //                     hopPtr1 < endPtr1;  // test
#                 //                              ++hopPtr1) // update
#                 hopPtr1 = a1;
                 la $t4, a1
#                 endPtr1 = a1 + used1;
                  sll $v0, $t1, 2
                  add $a1, $t4, $v0
#                goto FTest1;
                  j FTest1
begF1:#//         {
#                   target = *hopPtr1;
                lw $t0, 0($t4)
 #                  //if (target < 0 || target > 9)
#                   ///if (target >= 0 && target <= 9) goto endI4;
#                   if (target < 0) goto begI4;
```

```
                   blt $t0, $0, begI4
#                    if (target <= 9) goto endI4;
                   li $v1, 9
                   ble $t0, $v1,  endI4
begI4:#//            {
#                       //for (hopPtr11 = hopPtr1 + 1;  // init
#                       //         hopPtr11 < endPtr1;  // test
#                       //                   ++hopPtr11) // update
#                       hopPtr11 = hopPtr1 + 1;
                   addi $t6, $t4, 4
#                       goto FTest2;
                   j FTest2
begF2:#//               {
#                          *(hopPtr11 - 1) = *hopPtr11;
                lw $v0, 0($t6)
                   sw $v0, -4($t6)
#                       ++hopPtr11;
                addi $t6, $t6, 4
endF2:#//               }
FTest2:
#                       if (hopPtr11 < endPtr1) goto begF2;
                blt $t6, $a1, begF2

 #                      --used1;
                addi $t1, $t1, -1
 #                      --endPtr1;
                addi $a1, $a1, -4
 #                      --hopPtr1;
                   addi $t4, $t4, -4
endI4:#//            }
 #                ++hopPtr1;
                   addi $t4, $t4, 4
endF1:#//         }
FTest1:
#                 if (hopPtr1 < endPtr1) goto begF1;
                   blt $t4, $a1, begF1

#                 cout << nn09A1Str;
                   li $v0, 4
                   la $a0, nn09A1Str
                   syscall

#                 //if (used1 > 0)
#                if (used1 <= 0) goto endI5;
#                  li $a0, 0
                   ble $t1, $0, endI5
begI5:#//         {
 #                   hopPtr1 = a1;
                la $t4, a1

#                    endPtr1 = a1 + used1;
#                  la $a0, a1
                   sll $v0, $t1, 2
                   add $a1, $t4, $v0
```

```
#                     //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

endI5:#//           }
#                   cout << endl;

                   li $v0, 11
                   li $a0, '\n'
                   syscall

#                   used2 = 0;
                   li $t2, 0
#                   used3 = 0;
                   li $t3, 0
#                   hopPtr1 = a1;
                   la $t4, a1
#                   hopPtr2 = a2;
                la $t5, a2
#                   hopPtr3 = a3;
                la $t7, a3
 #                  endPtr1 = a1 + used1;
#                  la $a0, a1
                   sll $v0, $t1, 2
                   add $a1, $t4, $v0

#                   //while (hopPtr1 < endPtr1)
#                   goto WTest3;
                   j WTest3
begW3:#//           {
#                     intHolder = *hopPtr1;
                lw $t8, 0($t4)
#                     *hopPtr2 = intHolder;
                sw $t8, 0($t5)
#                     ++used2;
                addi $t2, $t2, 1
#                     ++hopPtr2;
                   addi $t5, $t5, 4
#                     *hopPtr3 = intHolder;
```

```
                    sw $t8, 0($t7)
#                     ++used3;
                    addi $t3, $t3, 1
#                     ++hopPtr3;
                    addi $t7, $t7, 4
#                     ++hopPtr1;
                    addi $t4, $t4, 4
endW3:#//          }
WTest3:
#                  if (hopPtr1 < endPtr1) goto begW3;
                blt $t4, $a1,  begW3

#                  iter = 0;
                    li $t8, 0
#                  //do
begDW3:#//         {
#                     ++iter;
                    addi $t8, $t8, 1
#                     count = 0;
                    li $t9, 0
#                     //if (iter == 1)
#                     if (iter != 1) goto else6;
                    li $v1, 1
                    bne $t8, $v1, else6


begI6:#//             {
#                        //for (hopPtr1 = a1, endPtr1 = a1 + used1;  // multi-init
#                        //                     hopPtr1 < endPtr1;  // test
#                        //                              ++hopPtr1) // update
#                        hopPtr1 = a1;
                    la $t4, a1

#                        endPtr1 = a1 + used1;
                    sll $v0, $t1, 2
                    add $a1, $t4, $v0
#                        goto FTest3;
                    j FTest3
begF3:#//                {
#                           target = *hopPtr1;
                    lw $t0, 0($t4)

#                           //if (target != 5)
#                           if (target == 5) goto elseI7;
                    li $v1, 5
                    beq $t0, $v1, elseI7
begI7:#//                   {
#                              ++count;
                    addi $t9, $t9, 1
#                           goto endI7;
                    j endI7
#//                         }
elseI7:#//                  else
#//                         {
```

```
#                             //if (count != 0)
#                             if (count == 0) goto endI8;
                   beq $t9, $0, endI8
begI8:#//                     {
#                                *(hopPtr1 - count) = *hopPtr1;
                   sll $a0, $t9, 2
                   sub $a0, $t4, $a0
                   lw $v0, 0($t4)
                   sw $v0, 0($a0)

endI8:#//                     }
endI7:#//                  }
#                       ++hopPtr1;
                addi $t4, $t4, 4
endF3:#//               }
FTest3:
#                       if (hopPtr1 < endPtr1) goto begF3;
                   blt $t4, $a1, begF3

#                       used1 -= count;
                   sub $t1, $t1, $t9
#                       //if (used1 == 0)
#                       if (used1 != 0) goto endI9;
                   bne $t1, $0, endI9
begI9:#//               {
#                          hopPtr1 = a1;
                la $t4, a1
#                          *hopPtr1 = -99;
                   li $a0, -99
                   sw $a0, 0($t4)
#                          ++used1;
                   addi $t1, $t1, 1
endI9:#//               }
#                    goto endI6;
                   j endI6
#//                  }
else6:#//            else
#//                  {
#                       //if (iter == 2)
#                       if (iter != 2) goto elseI10;

                   li $v1, 2
                   bne $t8, $v1, elseI10

begI10:#//              {
#                          //for (hopPtr2 = a2, endPtr2 = a2 + used2;  // multi-init
#                          //                       hopPtr2 < endPtr2;  // test
#                          //                                ++hopPtr2) // update
#                          hopPtr2 = a2;
                   la $t5, a2
#                         endPtr2 = a2 + used2;
                   sll $v0, $t2, 2
                   add $a2, $t5, $v0
#                          goto FTest4;
```

```
                    j FTest4
begF4:#//                   {
#                              target = *hopPtr2;
                    lw $t0, 0($t5)
#                              //if (target > 4)
#                              if (target <= 4) goto elseI11;
                    li $v1, 4
                    ble $t0, $v1, elseI11
begI11:#//                     {
#                                 ++count;
                    addi $t9, $t9, 1
#                              goto endI11;
                    j endI11
#//                            }
elseI11:#//                    else
#//                            {
#                                 //if (count != 0)
#                                 if (count == 0) goto endI12;
#                   li $v1, 0
                    beq $t9, $0, endI12
begI12:#//                        {
#                                  *(hopPtr2 - count) = *hopPtr2;
                    sll $a0, $t9, 2
                    sub $a0, $t5, $a0
                    lw $v0, 0($t5)
                    sw $v0, 0($a0)
endI12:#//                        }
endI11:#//                     }
#                           ++hopPtr2;
                    addi $t5, $t5, 4
endF4:#//                   }
FTest4:
#                           if (hopPtr2 < endPtr2) goto begF4;
                    blt $t5, $a2, begF4

#                           used2 -= count;
                    sub $t2, $t2, $t9
#                           //if (used2 == 0)
#                           if (used2 != 0) goto endI13;
#                   li $a0, 0
                    bne $t2, $0, endI13
begI13:#//                  {
#                              hopPtr2 = a2;
                    la $t5, a2
#                              *hopPtr2 = -99;
                    li $v1, -99
                    sw $v1, 0($t5)
#                              ++used2;
                    addi $t2 ,$t2 , 1
endI13:#//                  }
#                        goto endI10;
                    j endI10
#//                      }
elseI10:#//              else
```

```
#//                     {
#                          //for (hopPtr3 = a3, endPtr3 = a3 + used3;  // multi-init
#                          //                      hopPtr3 < endPtr3;  // test
#                          //                               ++hopPtr3) // update
#                          hopPtr3 = a3;
                   la $t7, a3
#                          endPtr3 = a3 + used3;
                   sll $a0, $t3, 2
                   add $a3, $t7, $a0
#                          goto FTest5;
                   j FTest5
begF5:#//                  {
#                             target = *hopPtr3;
                   lw $t0, 0($t7)
#                             //if (target < 6)
#                             if (target >= 6) goto elseI14;
                   li $v1, 6
                   bge $t0, $v1, elseI14
begI14:#//                    {
#                                ++count;
                   addi $t9, $t9, 1
#                             goto endI14;
                   j endI14
#//                           }
elseI14:#//                   else
#//                           {
#                                //if (count != 0)
#                                if (count == 0) goto endI15;

                   beq $t9, $0, endI15
begI15:#//                       {
#                                   *(hopPtr3 - count) = *hopPtr3;
                   sll $a0 $t9, 2
                   sub $a0, $t7, $a0
                lw $v0, 0($t7)
                   sw $v0, 0($a0)
endI15:#//                       }
endI14:#//                    }
#                          ++hopPtr3;
                   addi $t7, $t7, 4
endF5:#//                  }
FTest5:
#                          if (hopPtr3 < endPtr3) goto begF5;
                   blt $t7, $a3, begF5

#                          used3 -= count;
                   sub $t3, $t3, $t9
#                          //if (used3 == 0)
#                          if (used3 != 0) goto endI16;

                   bne $t3, $0, endI16
begI16:#//                 {
#                             hopPtr3 = a3;
                   la $t7, a3
#                              *hopPtr3 = -99;
                   li $v0, -99
                   sw $v0, 0($t7)
#                              ++used3;
                   addi $t3, $t3, 1
endI16:#//                   }
endI10:#//                }
endI6:#//              }
endDW3:#//          }
#                 //while (iter < 3);
DWTest3:
#                 if (iter < 3) goto begDW3;
                   li $v1, 3
                   blt $t8, $v1, begDW3

endI3:#//      }

#              cout << procA1Str;
                   li $v0, 4
                   la $a0, procA1Str
                   syscall
#              //if (used1 > 0)
#              if (used1 <= 0) goto endI17;
                   ble $t1, $0, endI17
begI17:#//     {
#                 hopPtr1 = a1;
               la $t4, a1
#                 endPtr1 = a1 + used1;
                   sll $a0, $t1, 2
                   add $a1, $t4, $a0
#                 //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

endI17:#//     }
#              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 endI18;
                    ble $t2, $0, endI18
begI18:#//      {
#                  hopPtr2 = a2;
                 la $t5, a2
#                  endPtr2 = a2 + used2;
                    sll $a0, $t2, 2
                    add $a2, $t5, $a0
#                  //do
begDW5:#//         {
#                     cout << *hopPtr2 << ' ' << ' ';
                 li $v0, 1
                    lw $a0, 0($t5)
                    syscall
                    li $v0, 11
                    li $a0, ' '
                    syscall
                    li $v0, 11
                    li $a0, ' '
                    syscall
#                     ++hopPtr2;
                    addi $t5, $t5, 4
endDW5:#//         }
#                  //while (hopPtr2 < endPtr2);
DWTest5:
#                  if (hopPtr2 < endPtr2) goto begDW5;
                    blt $t5, $a2, begDW5

endI18:#//      }
#               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 endI19;
#                   li $a0, 0
                    ble $t3, $0, endI19
begI19:#//      {
#                  hopPtr3 = a3;
                la $t7, a3
#                  endPtr3 = a3 + used3;
                    sll $a0, $t3, 2
                    add $a3, $t7, $a0
#                  //do
begDW6:#//         {
```

```
#                    cout << *hopPtr3 << ' ' << ' ';
                   li $v0, 1
                   lw $a0, 0($t7)
                   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

endI19:#//     }
#              cout << endl;
                   li $v0, 11
                   li $a0, '\n'
                   syscall

#              cout << dacStr;
                   li $v0, 4
                   la $a0, dacStr
                   syscall
#              cin >> reply;
                   li $v0, 12
                   syscall
                   move $t8, $v0
endW1:#//   }
WTest1:
#           ///if (reply != 'n' && reply != 'N') goto begW1;
#           if (reply == 'n') goto xitW1;
                   li $v1, 'n'
                   beq $t8, $v1, xi#change to $v0 instead of #v1
#           if (reply != 'N') goto begW1;
                   li $v1, 'N'
                   bne $t8, $v1, begW1
xitW1:

#           cout << dlStr << '\n';
                   li $v0, 4
                   la $a0, dlStr
                   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 << dlStr << '\n';
                      li $v0, 4
                      la $a0, dlStr
                      syscall
                      li $v0, 11
                      li $a0, '\n'
                      syscall

#           return 0;
                      li $v0, 10
                      syscall

#}
```