```
/*##########################################################################
# Noah del Angel, CS 2318-002, Assignment 2 Part 2
##########################################################################
# This is an exercise for you to re-write a C++ program (that involves
# processing
# 1-D arrays using selection and repetition constructs) in a form that will
# facilitate its translation into MIPS assembly language, namely:
#  All high-level selection constructs (if, if-else), repetition constructs (for
#  , while, do-while) and jump constructs (break, continue) are replaced with
#  conditional and unconditional goto's.
# You are to modify a2p2.cpp of this supplied file. The supplied file also has
# Excel spreadsheet (Assign02P2_labelingDFIW_minSelectTestCases.xlsx) with 2
# worksheets:
#  a2p2_labelingDFIW that has all the DFIW (Do-while_For_If_While) in the
#  supplied program labeled for you.
#     For uniformity (and to simplify debugging/grading/...) you MUST name your
#     MUST name your labels (begDW1, FTest1, else1, endW1, xitDW1, brk1, ...)
#     accordingly.
#  minSelectTestCases that has the cases you should at least test (because they
#  they are the cases I will use when grading).
#     minSelectTestCases was obtained using the supplied program, so a
#     correctly-modified-as-required program should produce the same result.
#  Extra time-saving, tedium-reducing helper if you work in Linux command
#  (LCLE):
#  ( Link from CS3358 you may find useful:  C++/C Programming on CS Linux:
#  Minimal Survival Guide )
#     Extract the files (a2p2test.in, a2p2testBase.out and Makefile) of this
#     LCLE extra helper file and upload them into the same folder that has the
#     file
#     (a2p2.cpp) you are working on.
#     (Note that the helper file does not include a2p2.cpp that's part of this
#     supplied file above.)
#     Do make to (re-)compile a2p2.cpp you are working on; if successful,
#     executable file a2p2 will be created (replacing any earlier version that
#     may exist).
#     Do make test to run the test cases specified in a2p2test.in (encompassing
#     the minSelectTestCases referred to earlier) using the last successfully
#     executable file a2p2, with the test run outcome captured in a2p2test.out.
#     Do make diff (or manually do diff a2p2test.out a2p2testBase.out) to see if
#     there are any differences between the test run outcome from your last
#     successfully created executable file a2p2 and the baseline test run
#     outcome.
*/
#include <iostream>
using namespace std;

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...";

int main()
{
               char reply;
               int used1,
                   used2,
                   used3,
                   target,
                   intHolder,
                   count,
                   iter,
```

```
    *hopPtr1,
    *hopPtr11,
    *hopPtr2,
    *hopPtr3,
    *endPtr1,
    *endPtr2,
    *endPtr3;

reply = 'y';

goto WTest1;
begW1:

   used1 = 0;
   hopPtr1 = a1;

   goto WTest2;
   begW2:

      cout << einStr;
      cout << (used1 + 1);
      cout << ':' << ' ';
      cin >> *hopPtr1;
      ++used1;
      ++hopPtr1;

      if (used1 >= 12 ) goto else1;

         cout << emiStr;
         cin >> reply;

         goto endif1;

      else1:

         cout << moStr << 12 << ieStr << endl;
         reply = 'n';

      endif1:

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

   cout << begA1Str;

   if (used1 <= 0) goto endif2;

      hopPtr1 = a1;
      endPtr1 = a1 + used1;

      begDW1:

         cout << *hopPtr1 << ' ' << ' ';
         ++hopPtr1;

      DWTest1: if (hopPtr1 < endPtr1) goto begDW1;

   endif2:

   cout << endl;

   if (used1 <= 0) goto endif3;

      hopPtr1 = a1,
      endPtr1 = a1 + used1;

      goto Ftest1;

         begF1:
         target = *hopPtr1;
         if ( target < 0 ) goto bodyIf4;
         if ( target <= 9 ) goto endIf4;
```

```
   bodyIf4:

      hopPtr11 = hopPtr1 + 1;
      begF2:

         *(hopPtr11 - 1) = *hopPtr11;
         ++hopPtr11;

      Ftest2: if ( hopPtr11 < endPtr1 ) goto begF2;

      --used1;
      --endPtr1;
      --hopPtr1;

   endIf4:

++hopPtr1;

Ftest1: if ( hopPtr1 < endPtr1 ) goto begF1;

cout << nn09A1Str;

if (used1 <= 0) goto endIf5;

   hopPtr1 = a1;
   endPtr1 = a1 + used1;

   begDW2:
      cout << *hopPtr1 << ' ' << ' ';
      ++hopPtr1;

   DWTest2: if (hopPtr1 < endPtr1) goto begDW2;

endIf5:

cout << endl;

used2 = 0;
used3 = 0;
hopPtr1 = a1;
hopPtr2 = a2;
hopPtr3 = a3;
endPtr1 = a1 + used1;

goto WTest3;
begW3:

   intHolder = *hopPtr1;
   *hopPtr2 = intHolder;
   ++used2;
   ++hopPtr2;
   *hopPtr3 = intHolder;
   ++used3;
   ++hopPtr3;
   ++hopPtr1;

WTest3: if (hopPtr1 < endPtr1) goto begW3;

iter = 0;

begDW3:

   ++iter;
   count = 0;

   if (iter != 1) goto else6;

      hopPtr1 = a1,
      endPtr1 = a1 + used1;

      goto Ftest3;
      begF3:
```

```
            target = *hopPtr1;

            if (target == 5) goto else7;

               ++count;
               goto endif7;

            else7:

               if (count == 0) goto endif8;

                  *(hopPtr1 - count) = *hopPtr1;

               endif8:

            endif7:

            ++hopPtr1;

         Ftest3: if ( hopPtr1 < endPtr1 ) goto begF3;

         used1 -= count;

         if (used1 != 0) goto endif9;

            hopPtr1 = a1;
            *hopPtr1 = -99;
            ++used1;

         endif9:

      goto endif6;
      else6:

         if (iter != 2) goto else10;

            hopPtr2 = a2,
            endPtr2 = a2 + used2;

            goto Ftest4;
            begF4:

               target = *hopPtr2;

               if (target <= 4) goto else11;

                  ++count;

                  goto endif11;

               else11:

                  if (count == 0) goto endif12;

                     *(hopPtr2 - count) = *hopPtr2;

                  endif12:

               endif11:

            ++hopPtr2;

            Ftest4: if ( hopPtr2 < endPtr2 ) goto begF4;

            used2 -= count;

            if (used2 != 0) goto endif13;

               hopPtr2 = a2;
               *hopPtr2 = -99;
               ++used2;
```

```
            endif13:
         goto endif10;

         else10:

            hopPtr3 = a3,
            endPtr3 = a3 + used3;

            goto Ftest5;
            begF5:

               target = *hopPtr3;

               if (target >= 6) goto else14;

                  ++count;
                  goto endif14;

               else14:

                  if (count == 0) goto endif15;

                     *(hopPtr3 - count) = *hopPtr3;

                  endif15:

               endif14:

            ++hopPtr3;

            Ftest5: if ( hopPtr3 < endPtr3 ) goto begF5;

            used3 -= count;

            if (used3 != 0) goto endif16;

               hopPtr3 = a3;
               *hopPtr3 = -99;
               ++used3;

            endif16:

         endif10:

      endif6:

   DWTest4: while (iter < 3) goto begDW3;

endif3:

cout << procA1Str;

if (used1 <= 0) goto endif17;

   hopPtr1 = a1;
   endPtr1 = a1 + used1;

   begDW4:

      cout << *hopPtr1 << ' ' << ' ';
      ++hopPtr1;

   testDw4: if (hopPtr1 < endPtr1) goto begDW4;

endif17:

cout << endl;

cout << procA2Str;

if (used2 <= 0) goto endif18;
```

```
            hopPtr2 = a2;
            endPtr2 = a2 + used2;

            begDW5:
               cout << *hopPtr2 << ' ' << ' ';
               ++hopPtr2;

            DWTest5: if (hopPtr2 < endPtr2) goto begDW5;

         endif18:

         cout << endl;

         cout << procA3Str;

         if (used3 <= 0) goto endif19;

            hopPtr3 = a3;
            endPtr3 = a3 + used3;

            begDW6:

               cout << *hopPtr3 << ' ' << ' ';
               ++hopPtr3;

            DWTest6: if (hopPtr3 < endPtr3) goto begDW6;

         endif19:

         cout << endl;

         cout << dacStr;
         cin >> reply;

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


      cout << dlStr << '\n';
      cout << byeStr << '\n';
      cout << dlStr << '\n';

      return 0;
}
```