// Austin Keith Faulkner: a\_f408

// September 29, 2019

//

// FILE: sequenceTest.cpp

// An interactive test program for the sequence class

#include <cctype> // provides toupper

#include <iostream> // provides cout and cin

#include <cstdlib> // provides EXIT\_SUCCESS

#include "sequence.h"

using namespace CS3358\_FA2019\_A04;

using namespace std;

// PROTOTYPES for functions used by this test program:

void print\_menu();

// Pre: (none)

// Post: A menu of choices for this program is written to cout.

char get\_user\_command();

// Pre: (none)

// Post: The user is prompted to enter a one character command.

// The next character is read (skipping blanks and newline

// characters), and this character is returned.

template <class Item>

void show\_list(Item src);

// Pre: (none)

// Post: The items of src are printed to cout (one per line).

int get\_object\_num();

// Pre: (none)

// Post: The user is prompted to enter either 1 or 2. The

// prompt is repeated until a valid integer can be read

// and the integer's value is either 1 or 2. The valid

// integer read is returned. The input buffer is cleared

// of any extra input until and including the first

// newline character.

double get\_number();

// Pre: (none)

// Post: The user is prompted to enter a real number. The prompt

// is repeated until a valid real number can be read. The

// valid real number read is returned. The input buffer is

// cleared of any extra input until and including the

// first newline character.

char get\_character();

// Pre: (none)

// Post: The user is prompted to enter a non-whitespace character.

// The prompt is repeated until a non-whitespace character

// can be read. The non-whitespace character read is returned.

// The input buffer is cleared of any extra input until and

// including the first newline character.

int main(int argc, char \*argv[])

{

sequence<double> s1; // A sequence of double for testing

sequence<char> s2; // A sequence of char for testing

int objectNum; // A number to indicate selection of s1 or s2

double numHold; // Holder for a real number

char charHold; // Holder for a character

char choice; // A command character entered by the user

cout << "An empty sequence of real numbers (s1) and\n"

<< "an empty sequence of characters (s2) have been created."

<< endl;

do

{

if (argc == 1)

print\_menu();

choice = toupper( get\_user\_command() );

switch (choice)

{

case '!':

objectNum = get\_object\_num();

if (objectNum == 1)

{

s1.start();

cout << "s1 started" << endl;

}

else

{

s2.start();

cout << "s2 started" << endl;

}

break;

case '&':

objectNum = get\_object\_num();

if (objectNum == 1)

{

s1.end();

cout << "s1 ended" << endl;

}

else

{

s2.end();

cout << "s2 ended" << endl;

}

break;

case '+':

objectNum = get\_object\_num();

if (objectNum == 1)

{

if ( ! s1.is\_item() )

cout << "Can't advance s1." << endl;

else

{

s1.advance();

cout << "Advanced s1 one item."<< endl;

}

}

else

{

if ( ! s2.is\_item() )

cout << "Can't advance s2." << endl;

else

{

s2.advance();

cout << "Advanced s2 one item."<< endl;

}

}

break;

case '-':

objectNum = get\_object\_num();

if (objectNum == 1)

{

if ( ! s1.is\_item() )

cout << "Can't move back s1." << endl;

else

{

s1.move\_back();

cout << "Moved s1 back one item."<< endl;

}

}

else

{

if ( ! s2.is\_item() )

cout << "Can't move back s2." << endl;

else

{

s2.move\_back();

cout << "Moved s2 back one item."<< endl;

}

}

break;

case '?':

objectNum = get\_object\_num();

if (objectNum == 1)

{

if ( s1.is\_item() )

cout << "s1 has a current item." << endl;

else

cout << "s1 has no current item." << endl;

}

else

{

if ( s2.is\_item() )

cout << "s2 has a current item." << endl;

else

cout << "s2 has no current item." << endl;

}

break;

case 'C':

objectNum = get\_object\_num();

if (objectNum == 1)

{

if ( s1.is\_item() )

cout << "Current item in s1 is: "

<< s1.current() << endl;

else

cout << "s1 has no current item." << endl;

}

else

{

if ( s2.is\_item() )

cout << "Current item in s2 is: "

<< s2.current() << endl;

else

cout << "s2 has no current item." << endl;

}

break;

case 'P':

objectNum = get\_object\_num();

if (objectNum == 1)

{

if (s1.size() > 0)

{

cout << "s1: ";

show\_list(s1);

cout << endl;

}

else

cout << "s1 is empty." << endl;

}

else

{

if (s2.size() > 0)

{

cout << "s2: ";

show\_list(s2);

cout << endl;

}

else

cout << "s2 is empty." << endl;

}

break;

case 'S':

objectNum = get\_object\_num();

if (objectNum == 1)

cout << "Size of s1 is: " << s1.size() << endl;

else

cout << "Size of s2 is: " << s2.size() << endl;

break;

case 'A':

objectNum = get\_object\_num();

if (objectNum == 1)

{

numHold = get\_number();

s1.add(numHold);

cout << numHold << " added to s1." << endl;

}

else

{

charHold = get\_character();

s2.add(charHold);

cout << charHold << " added to s2." << endl;

}

break;

case 'R':

objectNum = get\_object\_num();

if (objectNum == 1)

{

if ( s1.is\_item() )

{

numHold = s1.current();

s1.remove\_current();

cout << numHold << " removed from s1." << endl;

}

else

cout << "s1 has no current item." << endl;

}

else

{

if ( s2.is\_item() )

{

charHold = s2.current();

s2.remove\_current();

cout << charHold << " removed from s2." << endl;

}

else

cout << "s2 has no current item." << endl;

}

break;

case 'Q':

cout << "Quit option selected...bye" << endl;

break;

default:

cout << choice << " is invalid...try again" << endl;

}

}

while (choice != 'Q');

cin.ignore(999, '\n');

cout << "Press Enter or Return when ready...";

cin.get();

return EXIT\_SUCCESS;

}

void print\_menu()

{

cout << endl;

cout << "The following choices are available:\n";

cout << " ! Activate the start() function\n";

cout << " & Activate the end() function\n";

cout << " + Activate the advance() function\n";

cout << " - Activate the move\_back() function\n";

cout << " ? Print the result from the is\_item() function\n";

cout << " C Print the result from the current() function\n";

cout << " P Print a copy of the entire sequence\n";

cout << " S Print the result from the size() function\n";

cout << " A Add a new item with the add(...) function\n";

cout << " R Activate the remove\_current() function\n";

cout << " Q Quit this test program" << endl;

}

char get\_user\_command()

{

char command;

cout << "Enter choice: ";

cin >> command;

cout << "You entered ";

cout << command << endl;

return command;

}

template <class Item>

void show\_list(Item src)

{

for ( src.start(); src.is\_item(); src.advance() )

cout << src.current() << " ";

}

int get\_object\_num()

{

int result;

cout << "Enter object # (1 = s1, 2 = s2) ";

cin >> result;

while ( ! cin.good() )

{

cerr << "Invalid integer input..." << endl;

cin.clear();

cin.ignore(999, '\n');

cout << "Re-enter object # (1 = s1, 2 = s2) ";

cin >> result;

}

// cin.ignore(999, '\n');

while (result != 1 && result != 2)

{

cin.ignore(999, '\n');

cerr << "Invalid object # (must be 1 or 2)..." << endl;

cout << "Re-enter object # (1 = s1, 2 = s2) ";

cin >> result;

while ( ! cin.good() )

{

cerr << "Invalid integer input..." << endl;

cin.clear();

cin.ignore(999, '\n');

cout << "Re-enter object # (1 = s1, 2 = s2) ";

cin >> result;

}

// cin.ignore(999, '\n');

}

cout << "You entered ";

cout << result << endl;

return result;

}

double get\_number()

{

double result;

cout << "Enter a real number: ";

cin >> result;

while ( ! cin.good() )

{

cerr << "Invalid real number input..." << endl;

cin.clear();

cin.ignore(999, '\n');

cout << "Re-enter a real number ";

cin >> result;

}

// cin.ignore(999, '\n');

cout << "You entered ";

cout << result << endl;

return result;

}

char get\_character()

{

char result;

cout << "Enter a non-whitespace character: ";

cin >> result;

while ( ! cin )

{

cerr << "Invalid non-whitespace character input..." << endl;

cin.ignore(999, '\n');

cout << "Re-enter a non-whitespace character: ";

cin >> result;

}

// cin.ignore(999, '\n');

cout << "You entered ";

cout << result << endl;

return result;

}