Computer Programming I — Homework Assignment #9

An identifier is a series of characters consisting of letters, digits and underscores (\_) that does not begin with a digit. Please complete programs in **Student ID-hw9-1.cpp**, **Student ID-hw9-2.cpp**, **Student ID-hw9-3.cpp** and **Student ID-hw9-4.cpp** that read in a legal C**++** program from the file **test.cpp**, and write all non-keyword identifiers into a text file **identifiers.txt**. For the sake of simplicity, we suppose that, in **test.cpp**, all comments are *single-line comments* which begin with //, no string constant contains the character ", and no character constant is the character. Moreover, we also assume that the program in **test.cpp** is syntactically correct. The following program is a sample program in **test.cpp**:

|  |
| --- |
| #include <iostream>  using namespace std;  int main()  {  system( "mode con:cols=60 lines=16" );  system( "color F0" );  char s1[ 20 ];  char s2[] = "string literal";  cout << "Enter the string \"eat here\": ";  cin >> s1; // reads "eat"  cout << "s1 is: " << s1 << "\nstring2 is: " << s2;  cout << "\nstring1 with spaces between characters is:\n";  for ( int i = 0; s1[ i ] != '\0'; i++ )  cout << s1[ i ] << ' ';  cout << '\n';  cout << '\n' << s1[ 0 ] << ' ' << s1[ 1 ] << ' ' << s1[ 2 ] << '\n';  cout << '\n' << s1[ 0 ] << '\'' << s1[ 1 ] << '\'' << s1[ 2 ] << '\n';  cout << '\n' << s1[ 0 ] << ' ' << s1[ 1 ] << '\'' << s1[ 2 ] << '\n';  cout << '\n' << s1[ 0 ] << '\'' << s1[ 1 ] << ' ' << s1[ 2 ] << '\n';  cin >> s1; // reads "here"  cout << "\nstring1 is: " << s1 << '\n';  system( "pause" );  } |

If your program reads in the above program, the contents in the output file should appear as follows:

|  |
| --- |
| iostream  std  main  system  s1  s2  cout  cin  i |

|  |
| --- |
| // Demonstrating C++ Standard Library class template vector.  #include <iostream>  #include <iomanip>  #include <vector>  using namespace std;  void outputVector( const vector< int > & ); // display the vector  void inputVector( vector< int > &, int start ); // input values into the vector  int main()  {  vector< int > integers1( 7 ); // 7-element vector< int >  vector< int > integers2( 10 ); // 10-element vector< int >    // print integers1 size and contents  cout << "Size of vector integers1 is " << integers1.size()  << "\nvector after initialization:" << endl;  outputVector( integers1 );  // print integers2 size and contents  cout << "\nSize of vector integers2 is " << integers2.size()  << "\nvector after initialization:" << endl;  outputVector( integers2 );  // input and print integers1 and integers2  cout << "\nEnter 17 integers:" << endl;  inputVector( integers1, 1 );  inputVector( integers2, integers1.size() + 1 );  cout << "\nAfter input, the vectors contain:\n"  << "integers1:" << endl;  outputVector( integers1 );  cout << "integers2:" << endl;  outputVector( integers2 );  // use inequality (!=) operator with vector objects  cout << "\nEvaluating: integers1 != integers2" << endl;  if ( integers1 != integers2 )  cout << "integers1 and integers2 are not equal" << endl;  // create vector integers3 using integers1 as an  // initializer; print size and contents  vector< int > integers3( integers1 ); // copy constructor  cout << "\nSize of vector integers3 is " << integers3.size()  << "\nvector after initialization:" << endl;  outputVector( integers3 );  // use assignment (=) operator with vector objects  cout << "\nAssigning integers2 to integers1:" << endl;  integers1 = integers2; // assign integers2 to integers1  cout << "integers1:" << endl;  outputVector( integers1 );  cout << "integers2:" << endl;  outputVector( integers2 );  // use equality (==) operator with vector objects  cout << "\nEvaluating: integers1 == integers2" << endl;  if ( integers1 == integers2 )  cout << "integers1 and integers2 are equal" << endl;  // use square brackets to create rvalue  cout << "\nintegers1[5] is " << integers1[ 5 ];  // use square brackets to create lvalue  cout << "\n\nAssigning 1000 to integers1[5]" << endl;  integers1[ 5 ] = 1000;  cout << "integers1:" << endl;  outputVector( integers1 );  // attempt to use out-of-range subscript  cout << "\nAttempt to assign 1000 to integers1.at( 15 )" << endl;  integers1.at( 15 ) = 1000; // ERROR: out of range  }  void outputVector( const vector< int > &array )  {  size\_t i;    for ( i = 0; i < array.size(); i++ )  {  cout << setw( 12 ) << array[ i ];  if ( ( i + 1 ) % 4 == 0 )  cout << endl;  }  if ( i % 4 != 0 )  cout << endl;  }  void inputVector( vector< int > &array, int start )  {  for ( size\_t i = 0; i < array.size(); i++ )  cin >> array[ i ];  } |

If your program reads in the above program, the contents in the output file should appear as follows:

|  |
| --- |
| iostream  iomanip  vector  std  outputVector  inputVector  start  main  integers1  integers2  cout  size  endl  integers3  at  array  size\_t  i  setw  cin |