```
/*
  File: vector2D.cpp
  Created by: Tan Qi Hao
  Creation Date: 3/6/2019
  Synopsis: This program read 2 2D vectors and then apply it to the
addition
            ,subtraction, scalar multiplication, and perpendicularity
between
               the 2 2D vectors.
*/

#include <iostream>
#include <iomanip>
#include <cmath>

using namespace std;

const double EPSILON(1e-12);

// function prototypes
// FUNCTION PROTOTYPE FOR read_vector HERE.
void read_vector(const string & prompt1, double & x, double & y);

// FUNCTION PROTOTYPE FOR vector_length HERE.
double vector_length(double x, double y);

// FUNCTION PROTOTYPE FOR write_vector HERE.
void write_vector(const string & prompt2, double x, double y);

// FUNCTION PROTOTYPE FOR vector_add HERE.
void vector_add(double x1, double y1, double x2, double y2, double & x3,
double & y3);

// FUNCTION PROTOTYPE FOR vector_subtract HERE.
void vector_subtract(double x1, double y1, double x2, double y2, double &
x3, double & y3);

// FUNCTION PROTOTYPE FOR scalar_mult HERE.
void scalar_mult(double x1, double y1, double s, double & x2, double &
y2);

// FUNCTION PROTOTYPE FOR normalize HERE.
void normalize(double & x, double & y);

// FUNCTION PROTOTYPE FOR perpendicular HERE.
void perpendicular(double x1, double y1, double x2, double y2);

// *** DO NOT CHANGE ANY CODE IN THE MAIN FUNCTION.
int main()
{
  double u1, v1; // coordinates of first vector
  double u2, v2; // coordinates of second vector
  double u3, v3;
  double scalar;
```

```
  read_vector("Enter first vector (2 floats): ", u1, v1);
  read_vector("Enter second vector (2 floats): ", u2, v2);

  cout << "Enter scalar multiplier: ";
  cin >> scalar;
  cout << endl;

  write_vector("First vector: ", u1, v1);
  write_vector("Second vector: ", u2, v2);

  cout << endl;

  vector_add(u1, v1, u2, v2, u3, v3);
  write_vector("Vector add: ", u3, v3);

  vector_subtract(u1, v1, u2, v2, u3, v3);
  write_vector("Vector subtract: ", u3, v3);

  scalar_mult(u1, v1, scalar, u3, v3);
  write_vector("Scalar multiplier: ", u3, v3);

  cout << endl;

  write_vector("First vector: ", u1, v1);
  write_vector("Second vector: ", u2, v2);
  perpendicular(u1, v1, u2, v2);

  return(0);
}

// DEFINE FUNCTION read_vector HERE.
void read_vector(const string & prompt1, double & x, double & y)
{
  cout << prompt1;
  cin >> x >> y;
 }

// DEFINE FUNCTION vector_length HERE.
double vector_length(double x, double y)
{
  double length; //Find the vector length.

 length = sqrt(pow(x , 2.0) + pow(y , 2.0));

 return length;
}


// DEFINE FUNCTION write_vector HERE.
void write_vector(const string & prompt2, double x, double y)
{

  cout << prompt2 << "(" << x << ", " << y << ") has length "
```

```
       << vector_length(x, y);
  cout << endl;

}


// DEFINE FUNCTION vector_add HERE.
void vector_add( double x1, double y1, double x2, double y2, double & x3,
double & y3){
  x3 = x1 + x2;
  y3 = y1 + y2;

}

// DEFINE FUNCTION vector_subtract HERE.
void vector_subtract( double x1, double y1, double x2, double y2, double &
x3, double & y3){
  x3 = x1 - x2;
  y3 = y1 - y2;
}

// DEFINE FUNCTION scalar_mult HERE.
void scalar_mult(double x1, double y1, double scalar, double & x2, double
& y2)
{
  x2 = scalar * x1;
  y2 = scalar * y1;
  cout << endl;
}

// DEFINE FUNCTION normalize HERE.
void normalize(double & x, double & y)
{
  double length = vector_length(x, y); //Find the vector length

  if (abs(length - 0) < EPSILON){

    x = x / length;
    y = y / length;
 }

  else {

   double x = 0;
   double y = 0;

  }

}

// DEFINE FUNCTION perpendicular HERE.

void perpendicular(double  x1, double  y1, double  x2, double  y2){
   normalize(x1, y1);
```

```
   normalize(x2, y2);

   double px1 = -y1; // The x-coordinate of the first perpendicular
vector.
   double py1 = x1;  // The y-coordinate of the first perpendicular
vector.
   double px2 = -px1; // The x-coordinate of the second perpendicular
vector.
   double py2 = -py1; // The y-coordinate of the second perpendicular
vector.

   // vector v1 =(x1, y1), v2 = (x2, y2)
   // new vector: p1 = (px1, py1), p2 = (px2, py2)

   /* Determine if v2 is the same as either p1 or p2 */
   if ((abs(x2 - px1) < EPSILON && abs(y2 - py1) < EPSILON) || (abs(x2 -
px2) < EPSILON && abs(y2 - py2) < EPSILON))
     {
       cout << "Vectors are PERPENDICULAR." << endl;
     }
   else
     {
       cout << "Vectors are NOT PERPENDICULAR." << endl;
     }
}
```