```
/*
 Author: Aryan Bhatt
 Course: CSCI 136
 Instructor: Genady Maryash
 Assignment: HW 5.6
*/

Rearrange

double read_double(string prompt) {
  cout << prompt << ": ";
  double value;
  cin >> value;
  return value;
}
int main() {
  double price = read_double("First item");
  price = price + read_double("Next item");
  double rate = read_doub1e("Tax rate in percent");
  double tax = price * rate / 100;
  cout << "Amount due: " << (price + tax) << endl;
}
```

```
Code Checks 1, 2, and 3
#include <string>
#include <iostream>
using namespace std;

int find_occurrence(string str, string ch, int n);
string replace_at(string str, int position, string replacement);

/**
   Replaces all pairs of straight quotes with curly quotes.
   @param str the string to process
   @return str with adjacent pairs of straight quotes changed to
   curly quotes
*/
string smart_quotes(string str) {
   string result = str;
   string left_quote = "“";
   string right_quote = "”";
   while (find_occurrence(result, "\"", 2) != -1) {
      result = replace_at(result, find_occurrence(result, "\"", 1), left_quote);
```

```
    result = replace_at(result, find_occurrence(result, "\"", 1), right_quote);
  }

  return result;
}

/**
  Finds the nth occurrence of a given character in a string.
  @param str the string
  @param ch the character to search
  @param n the occurrence count
  @return the position of the nth occurrence of ch in str, or -1
     if ch doesn't occur n times.
*/
int find_occurrence(string str, string ch, int n) {
  int count = 0;
  for (int i = 0; i < str.length(); i++) {
    if (str.substr(i, 1) == ch) {
      count++;
      if (count == n) {
        return i;
      }
    }
  }
  return -1;
}

/**
  Replaces a character of a string at a given position.
  @param str the string where the replacement takes place
  @param position the position of the character to be replaced
  @param replacement the replacement string
  @return str with the character at the position changed to
    the replacement string, or the original string
    if position was not valid.
*/
string replace_at(string str, int position, string replacement) {
  if (0 <= position && position < str.length()) {
    return str.substr(0, position) + replacement +
      str.substr(position + 1);
  }
  else {
    return str;
```

```
  }
}
```