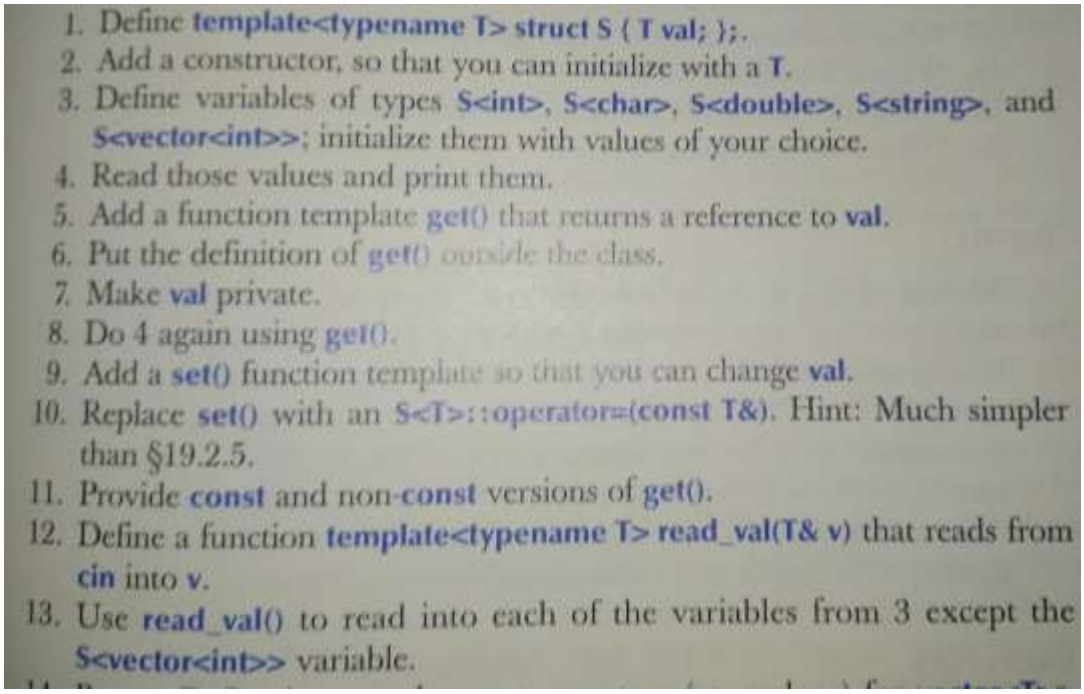
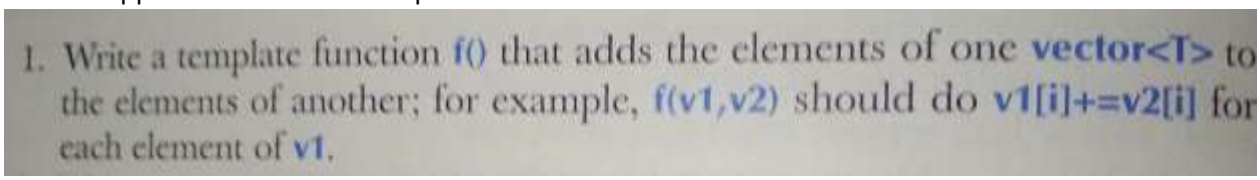
drill.cpp – the drill exercise from Chapter 19



1. Define template<typename T> struct S { T val; };.
2. Add a constructor, so that you can initialize with a T.
3. Define variables of types S<int>, S<char>, S<double>, S<string>, and S<vector<int>>; initialize them with values of your choice.
4. Read those values and print them.
5. Add a function template get() that returns a reference to val.
6. Put the definition of get() outside the class.
7. Make val private.
8. Do 4 again using get().
9. Add a set() function template so that you can change val.
10. Replace set() with an S<T>::operator=(const T&). Hint: Much simpler than §19.2.5.
11. Provide const and non-const versions of get().
12. Define a function template<typename T> read_val(T& v) that reads from cin into v.
13. Use read_val() to read into each of the variables from 3 except the S<vector<int>> variable.

exercise.cpp – exercise 1 from Chapter 19



1. Write a template function f() that adds the elements of one vector<T> to the elements of another; for example, f(v1,v2) should do v1[i]+=v2[i] for each element of v1.