```
extern template class FixedArray<int, 3>;              // class template
extern template int dot(const FixedArray<int, 3>& a,   // function template
                        const FixedArray<int, 3>& b);  // for int and 3

extern template class FixedArray<double, 2>;           // for double and 2
extern template double dot(const FixedArray<double, 2>& a,
                           const FixedArray<double, 2>& b);

extern template class FixedArray<double, 3>;           // for double and 3
extern template double dot(const FixedArray<double, 3>& a,
                           const FixedArray<double, 3>& b);

}  // close namespace game

#endif  // INCLUDED_GAME_FIXEDARRAY
```

Specializations commonly used by the game project are provided by the game library. In the component header in the example above, we have used the **extern template** syntax to suppress object-code generation for instantiations of both the class template FixedArray and the function template dot for element types **int** and **double**, each for array sizes 2 and 3. To ensure that these specialized definitions are available in every program that might need them, we use the **template** syntax counterpart to *force* object-code generation within just the one .o corresponding to the game_fixedarray library component[4]:

```
// game_fixedarray.cpp:
#include <game_fixedarray.h>  // included as first substantive line of code

// Explicit-instantiation definitions for full template specializations
// commonly used by the game project are provided below.

template class game::FixedArray<int, 2>;               // class template
template int game::dot(const FixedArray<int, 2>& a,    // function template
                       const FixedArray<int, 2>& b);   // for int and 2

template class game::FixedArray<int, 3>;               // class template
template int game::dot(const FixedArray<int, 3>& a,    // function template
                       const FixedArray<int, 3>& b);   // for int and 3

template class game::FixedArray<double, 2>;            // for double and 2
template double game::dot(const FixedArray<double, 2>& a,
                          const FixedArray<double, 2>& b);
```

[4]Notice that we have chosen *not* to nest the explicit specializations — or any other definitions — of entities already declared directly within the game namespace, preferring instead to qualify each entity explicitly to be consistent with how we render free-function definitions to avoid self-declaration; see **lakos20**, section 2.5, "Component Source-Code Organization," pp. 333–342, specifically Figure 2-36b, p. 340. See also *Potential Pitfalls — Corresponding explicit-instantiation declarations and definitions* on page 371.