- Examples

In this lesson, we'll look at a few examples of using templates with friends.

WE'LL COVER THE FOLLOWING Example 1: Class Template General Friendship Explanation Example 2: Class Template Special Friendship Explanation Example 3: Class Template Type Friendship Explanation

Example 1: Class Template General Friendship

```
// templateClassTemplateGeneralFriendship.cpp
#include <iostream>
template <typename T> void myFriendFunction(T);
template <typename U> class MyFriend;

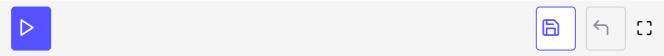
class GrantingFriendshipAsClass{
   template <typename U> friend void myFriendFunction(U);
   template <typename U> friend class MyFriend;

private:
   std::string secret("My secret from GrantingFriendshipAsClass.");
};

template <typename T>
class GrantingFriendshipAsClassTemplate{
   template <typename U> friend void myFriendFunction(U);
   template <typename U> friend class MyFriend;

private:
   std::string secret("My secret from GrantingFriendshipAsClassTemplate.");
```

```
};
template <typename T>
void myFriendFunction(T){
  GrantingFriendshipAsClass myFriend;
  std::cout << myFriend.secret << std::endl;</pre>
  GrantingFriendshipAsClassTemplate<double> myFriend1;
  std::cout << myFriend1.secret << std::endl;</pre>
}
template <typename T>
class MyFriend{
public:
  MyFriend(){
    GrantingFriendshipAsClass myFriend;
    std::cout << myFriend.secret << std::endl;</pre>
    GrantingFriendshipAsClassTemplate<T> myFriend1;
    std::cout << myFriend1.secret << std::endl;</pre>
};
int main(){
  std::cout << std::endl;</pre>
  int a{2011};
  myFriendFunction(a);
  MyFriend<double> myFriend;
  std::cout << std::endl;</pre>
```



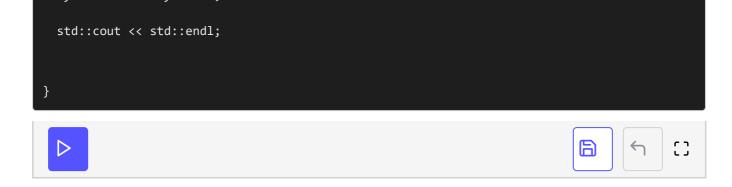
Explanation

In the example above, we have created a function myFriendFunction and a class MyFriend. We have defined two classes: GrantingFriendshipAsClass and GrantingFriendshipAsClassTemplate. As the names imply, we are using one class with a template and one without a template. The class MyFriend and the function myFriendFunction have access to the private members of the other classes by using a friend keyword. We have defined a private variable secret which is of type string and can be called from the object of myFriendFunction and MyFriend.

Example 2: Class Template Special Friendship

```
// templateClassTemplateSpecialFriendship.cpp
#include <iostream>
template <typename T> void myFriendFunction(T);
template <typename U> class MyFriend;
class GrantingFriendshipAsClass{
  friend void myFriendFunction<>(int);
  friend class MyFriend<int>;
private:
  std::string secret{"My secret from GrantingFriendshipAsClass."};
};
template <typename T>
class GrantingFriendshipAsClassTemplate{
  friend void myFriendFunction<>(int);
  friend class MyFriend<int>;
  friend class MyFriend<T>;
private:
  std::string secret{"My secret from GrantingFriendshipAsClassTemplate."};
};
template <typename T>
void myFriendFunction(T){
  GrantingFriendshipAsClass myFriend;
  std::cout << myFriend.secret << std::endl;</pre>
  GrantingFriendshipAsClassTemplate<T> myFriend1;
  std::cout << myFriend1.secret << std::endl;</pre>
}
template <typename T>
class MyFriend{
public:
  MyFriend(){
    GrantingFriendshipAsClass myFriend;
    std::cout << myFriend.secret << std::endl;</pre>
    GrantingFriendshipAsClassTemplate<int> myFriendInt;
    std::cout << myFriendInt.secret << std::endl;</pre>
    GrantingFriendshipAsClassTemplate<T> myFriendT;
    std::cout << myFriendT.secret << std::endl;</pre>
  }
};
int main(){
  std::cout << std::endl;</pre>
  int a{2011};
  myFriendFunction(a);
  MvFriend<int> mvFriend:
```

6



Explanation

This example is similar to example 1 with a small change: we have explicitly set the type of class template to int. Now, the class template is called for int and also for any other type mentioned in the typename portion.

Example 3: Class Template Type Friendship

```
// templateClassTemplateTypeFriendship.cpp
                                                                                               6
#include <iostream>
template <typename T>
class Bank{
  std::string secret{"Import secret from the bank."};
  friend T;
};
class Account{
public:
  Account(){
    Bank<Account> bank;
    std::cout << bank.secret << std::endl;</pre>
};
int main(){
  std::cout << std::endl;</pre>
  Account acc;
  std::cout << std::endl;</pre>
```

Explanation

In the code above, we have created an Account class that contains a Bank class object. We can access the Bank class member secret with the help of friend

Now, the value stored in the secret is accessible in the Account class.

We'll solve an exercise in the next lesson.