

# Move Function Solutions

- Why is it not possible to create a variable which is an rvalue reference?
  - A variable cannot be an rvalue (because its address can be taken)
- When an rvalue reference is passed as a function argument, what is the value category of the argument inside the function?
  - The argument is an lvalue

- What is the purpose of `std::move()`?
  - `std::move` casts its argument to an rvalue reference
- Why should it be called as `std::move()` and not just `move()`?
  - `move()` is a popular name for functions
  - Calling it as `std::move()` avoids potential conflicts

- If a variable is passed to `std::move()`, is it safe to use it again afterwards?
  - If a move operation has been performed on a variable, it should not be accessed
  - However, it is safe to assign to it and then use it

- Write a simple function `test()` which takes a string argument by reference and prints out “Lvalue reference”
- Write an overload of `test()` which takes a string argument by rvalue reference and prints out “Rvalue reference”
- What output would you expect from the main function in the following slide?
- Write a program to check your answers

```
int main() {  
    string&& r{ string {"Temp"} };  
    string l { string {"Perm"} };  
    string& lr{ l};  
    test(r);                // Lvalue reference  
    test(std::move(r));     // Rvalue reference  
    test (string{ {"Temp"} }); // Rvalue reference  
    test(l);               // Lvalue reference  
    test(lr);              // Lvalue reference  
    test(std::move(l));    // Rvalue reference  
}
```