

Modern C++ Overview

Part Three Solutions

lvalue and rvalue

- Explain briefly what is meant by the terms "lvalue" and "rvalue" in C++
 - An lvalue is a value whose address can be taken using the & operator
 - An rvalue is a value whose address cannot be taken using the & operator
- Give some examples
 - `int x;` `// x is an lvalue because &x is legal`
 - `2` `// 2 is an rvalue because &2 is not legal`
 - `myclass()` `// myclass() is an rvalue because &myclass() is not legal`

lvalue and rvalue references

- Explain briefly what is meant by the terms "lvalue reference" and "rvalue reference"
 - An lvalue reference is a traditional reference, e.g.
 - `int x;`
 - `int& y = x;` `// y is an (lvalue) reference to x`
 - An rvalue reference is a syntax feature that indicates that a variable needs to be initialized with an rvalue
 - `void func(int&& x);` `// Function taking an rvalue reference as argument`

Rvalue reference

- Write a function which has an rvalue reference as argument
- Write a program which calls this function, passing an rvalue to it
- Alter your program so that it passes an lvalue to it
- Explain your results
 - When passing an rvalue, the program runs as expected
 - When passing an lvalue, the program does not compile. This is because an lvalue cannot be automatically converted to an rvalue.

std::move

- Modify your program from the previous slide so that it works when passing an lvalue
- What implications does this have?
 - When an lvalue is passed as an rvalue to a function call, the data from the lvalue is moved into the function call argument, leaving an empty object
 - It is not safe to use the lvalue again, unless it is repopulated

Pass by value... or by move?

- Write a function which takes a vector of `std::string` by value and prints out the number of elements in the vector
- Write a program which creates a vector of `std::string` and passes it to the function. Print out the number of elements in the vector before making the call and after returning from it
- Modify your program so that the vector is passed as an rvalue

Pass by value... or by move?

- Explain your results
 - In the first case, the vector is an lvalue and is passed by value. The argument to the function will be a copy of the vector in main(). All three print statements will give the same number of elements
 - In the second case, the vector is an rvalue and is passed by move. The data in the vector in main() will be moved into the argument to the function, leaving an empty object. The final print statement will give the number of elements as zero