

# Lvalues and Rvalues Exercises

- Explain what is meant by the terms "lvalue" and "rvalue" in traditional C++
  - If we can find the address of a variable using the & operator, it is an lvalue
  - If not, it is an rvalue

- Identify the lvalues and rvalues in the following. Explain your answers

```
int x = 2;           // x is an lvalue, 2 is an rvalue
const int& x = 2;     // x is an lvalue, 2 is an rvalue
const char *x = "test"; // x is an lvalue, "test" is an rvalue
X x = func();        // x is an lvalue, func's return value is an rvalue
```

- In all cases, we can take the address of x, so it is an lvalue
- We cannot take the address of 2, "test" or a function return value, so they are all rvalues

- Explain what is meant by the terms "prvalue", "xvalue", "lvalue" and "glvalue" in modern C++
  - A prvalue is a “pure rvalue” or a literal
  - An xvalue is an “expiring value” or temporary
  - A glvalue is a “generalized lvalue” (either an lvalue or an xvalue)

- Identify the lvalues, xvalues and prvalues in the following. Explain your answers

<code>int x = 2;</code>	<code>// x is an lvalue, 2 is a prvalue</code>
<code>const int&amp; x = 2;</code>	<code>// x is an lvalue, 2 is a prvalue</code>
<code>const char *x = "test";</code>	<code>// x is an lvalue, 2 is a prvalue</code>
<code>X x = func();</code>	<code>// x is an lvalue, func's return value is an xvalue</code>

- In all cases, we can take the address of x, so it is an lvalue
- 2 and “test” are literals, so they are prvalues
- The return value from func() is a temporary, so it is an xvalue