Comparing References to Pointers


int
var
0x12ab
33

References	Pointers
• Don't use dereferencing for reading and writing • Can't be changed to reference something else • Must be initialized at declaration	• Must go through dereference operator to read/write through pointed to value • Can be changed to point somewhere else • Can be declared un-initialized (will contain garbage addresses)

Declaration and reading

```
//Declare pointer and reference

double double_value {12.34};

double& ref_double_value {double_value}; // Reference to double_value

double* p_double_value {&double_value}; //Pointer to double_value


//Reading
std::cout << "double_value : " << double_value << std::endl;
std::cout << "ref_double_value : " << ref_double_value << std::endl;
std::cout << "p_double_value : " << p_double_value << std::endl;
std::cout << "*p_double_value : " << *p_double_value << std::endl;
```

Writing

```
//Writting through pointer
std::cout << std::endl;
std::cout << "Writting through pointer : " << std::endl;

*p_double_value = 15.44;

std::cout << "double_value : " << double_value << std::endl;
std::cout << "ref_double_value : " << ref_double_value << std::endl;
std::cout << "p_double_value : " << p_double_value << std::endl;
std::cout << "*p_double_value : " << *p_double_value << std::endl;

//Writting through reference
std::cout << std::endl;
std::cout << "Writting through reference : " << std::endl;

ref_double_value = 18.44;

std::cout << "double_value : " << double_value << std::endl;
std::cout << "ref_double_value : " << ref_double_value << std::endl;
std::cout << "p_double_value : " << p_double_value << std::endl;
std::cout << "*p_double_value : " << *p_double_value << std::endl;
```

Can't make a reference refer to something else

```
double double_value {12.34};

double& ref_double_value {double_value}; // Reference to double_value

double other_double_value{100.23};

//This works, but it doesn't make ref_double_value reference other_double_value
//it merely changes the value referenced by ref_double_value to 100.23
//Visualize this in slides.
ref_double_value = other_double_value;


//If you change ref_double_value now, other_double_value stays the same
//proving that ref_double_value is not referencing other_double_value.

ref_double_value = 333.33;
```

A pointer can point somewhere else

```
//A pointer can point somewhere else
std::cout << std::endl;
std::cout << "A pointer can point somewhere else : " << std::endl;

p_double_value = & other_double_value;

std::cout << "double_value : " << double_value << std::endl;
std::cout << "ref_double_value : " << ref_double_value << std::endl;
std::cout << "p_double_value : " << p_double_value << std::endl;
std::cout << "*p_double_value : " << *p_double_value << std::endl;
std::cout << "other_double_value : " << other_double_value << std::endl;

std::cout << std::endl;
std::cout << "Changing the now pointed to value : " << std::endl;

*p_double_value = 555.66;

std::cout << "double_value : " << double_value << std::endl;
std::cout << "ref_double_value : " << ref_double_value << std::endl;
std::cout << "p_double_value : " << p_double_value << std::endl;
std::cout << "*p_double_value : " << *p_double_value << std::endl;
std::cout << "other_double_value : " << other_double_value << std::endl;
```

References are somewhat like const pointers

```
//References behave like constant pointers, but they have
//a much friendlier syntax as they don't require dereferencing
//to read and write through referenced data.

double *const const_p_double_value {&double_value};

const_p_double_value = &other_double_value;// Error
```