Pointers, Arrays and References

References

Variable that works as an alias for another variable. Used in function arguments to make permanent changes to a variable passed to it

Function gains direct access to the variable being passed.

```
cont type &arg
```

Arrays

Arrays are a numbered collection of elements

Range based `for`

```
int arr[] = {1, 2, 3, 4, 5};
for (int &x : arr) {
    cout << x << endl;
}
```

Range based for loop raises an error for arrays passed as arguments to functions.

Pointers

Pointers store the address of a variable

```
int i = 0;
int *p = &i;
```

The `*` or the "at" operator is used to access the address of a pointer.

```
*p = 10;
cout << i; // 10
```

Close relationship between pointers and arrays. Much more efficient to use pointer to access and modify array elements. Particularly important for arrays containing millions of element, or large multidimensional arrays.

`const` values can only be assigned to `const` pointers.

The pointer itself can also be declared as `const` e.g.

```
int * const ptr = &i;
```