Difference between const pointer, pointer to const, and const pointer to const
Pointer to constant - you cannot change the value of a the object the pointer is pointings at, but can change what object reference it points towards, ex:

```
int x = { 4 };
int y = { 5 };
const int* ptr = { &x };
*ptr = 10 //will not work, but can change the value via
ptr = &y;
```

Constant pointers - The pointer cannot change what it is pointing to (has a fixed memory location), but the object referenced can change its value, ex:

```
int x = { 4 };
int y = { 5 };
int* const ptr = { &x };
*ptr = 56; //is fine since the value that it points to changes but not what it is pointing at
ptr = &y; //would give an error since it cannot point anywhere else but where it was initilized
```

Constant Pointers to constants - value it points to cannot change, and location it looks at cannot change, ex:

```
const int x = { 4 };
const int y = { 5 };
const int* const ptr = { &x };
*ptr = 10; //error, assignment of read only
ptr = &y; //error assignment of read only
```