CSE 333 Section 7 - Casting

Welcome back to section! We're glad that you're here :)

Casting in C++

While in C++, we want to use casts that are more explicit in their behaviour. This gives us a better understanding of what happens when we read our code, because C-style casts can do many (sometimes unwanted) things. There are four types of casts we will use in C++:

`static_cast<to_type>(expression);`

- ★ Converts between pointers of related types.
  - Compiler error if not related.
- ★ Performs not pointer conversion (e.g. float to int conversion).

`dynamic_cast<to_type>(expression);`

- ★ Converts between pointers of related types.
  - Compiler error if not related.
  - Also checks at runtime to make sure it is a 'safe' conversion (returns `nullptr` if not).

`const_cast<to_type>(expression);`

- ★ Used to add or remove const-ness.

`reinterpret_cast<to_type>(expression);`

- ★ Casts between incompatible types *without changing the data*.
  - The types you are casting to and from must be the same size.
  - Will not let you convert between integer and floating point types.

Exercise 1

For each of the following snippets of code, fill in the blank with the most appropriate C++ style cast. Assume that we have the following classes defined:

```
class Base {
 public:
  int x;
};
```

```
class Derived : public Base {
 public:
  int y;
};
```

```
int64_t x = 0x7ffffffffe870;
char* str = reinterpret_cast<char *>(x);
```

```
void foo(Base *b) {
  Derived *d = dynamic_cast<Derived *>(b);
  // additional code omitted
}
```

```
Derived *d = new Derived;
Base *b = static_cast<Base *>(d);
```

```
double x = 64.382;
int64_t y = static_cast<int64_t>(x);
```