Chapter 1 Review Questions:

A pointer is an address of memory in which you have stored some data. It is usually a pointer to a data type

and what you pass is a-

What is the address of operator? - is the & and gives you the address of a variable

What is the de-reference operator? - it is the asterisk \*. It gives you value to which you are pointing to.

What is a null pointer? a Null pointer is a pointer to any data type, but the value is 0. You would want to define one

to help find bugs and is a good way to program to test.

What might cause segmentation fault? - Usually if you have a pointer pointing to something in memory that isn't mapped then it could cause it.

What are the possible uses of cost when dealing with pointers? -skipped

What is a function signature? - skipped

Why is it an error to have two functions with the same signature but different return types in one scope? - Because it is not allowed in C++ (C++11).

Chapter 2 Review Questions:

2.16 Review Questions.

1) Struct and class is equivalent but in Struct, all members are public. But in class, you have public, private, and protected members which is not possible in a Struct.

3) Friend functions are functions that are able to access private and public member of the class.

4) A static data member differs from non-static because -

6) to declare a member function to be cost means that function does not change any data member of our class.

Chapter 4 Review Questions:

4.5 Review Questions

4) the iterator pattern is the first pattern we discussed. With it, we are able to, in a standard way, walk through a data structure.

Is very powerful as with an iterator, you don't need to know which data structure your class implements.

6) the difference between composition and aggregation. Composite relation will manage the data over lifecycle of container. In Aggregate, container does not manage the lifecycle of the data.

Chapter 5 Review Questions:

5.13 review questions

1) what is difference between function declaration and function definition? We are declaring a type when we are writing it. We are defining it instead

when we are putting the prototype and body.

2)for every member function, it can have default functions. Meaning you can provide standard values to one of the or multiple arguments. So if

user doesn’t set one of these arguments, then it will have default value. Default values part of the definition.

6) By default in C, they are pass by value. Every copy is initialized on stack before calling the function. But in C++, we can also pass by reference.

Chapter 6 Review Questions:

6.11

1) what is diff between a function and a method? due to if a function in class was a virtual, it would thus be method.

2) what does it mean for base class to be hidden

3) constructor, copy constructor, copy assignment I

Chapter 7 Review Questions:

Abstract cl

What is a design patter? What’d most design patterns that in common.

Chapter 8 Review Questions:

8.9 Review Question

m

1) what does it mean when QObject A is aren’t of QObject B. Means that specifies at run time who is pointing to

\\\\\\\\'

event loop allows take representing'

l

Chapter 10 Review Questions

QMainWindow and Dialog both inherit from QWidget. QMainWindow is divided in multiple llllllllllllllllllllllllllllllllllllllllllllllllllllllllllllllllllllllllllllllllllllllllllllllllllllllllllll

Chapter 11 Review Questions

2) when we define template. We define template parameters. Can be multiple and can be type but also constant. instead the PARAM of function can only be types.

3) this is because to use same template, then you get an error.

Chapter 12 review

Very important is MainobjectFramework. Divided into two parts. One part allows you to know everything about QObjectType and the other about QValue Type. Like this because in QT, our types are divided into two categories. One has value types so you can clone and the other being dynamic from QObject. Everything inherited from QObject is not closable. Only one instance.

QObjectType and Value Type. Different types. Value Types can be copied.

QMetaObject