Written Questions

A) virtual & override

virtual allows a member function in a base class to be overwritten by a derived class. In the Trade class, the payoff function is defined using virtual and by including override in the payoff functions in the derived classes, the function can be used to calculate each contracts respective payoff.

Using **virtual** can cause the code to become slower at runtime as the virtual function in the base class must lookup the functions in the derived classes. However unless speed is extremely important in the code it is a very small issue.

Declaring payoff as a pure function requires the derived classes to have their own implementation of payoff. It cannot be called from the base class.

Override makes the function in a derived class override the virtual function in the base class. The binary code shouldnt be different as omitting an override statement will cause an error at compile time and the code wont run.

B) default & delete

	Default Ctor	**Destr-uctor**	**Copy Ctor**	**Copy As-sign-ment**	**Move Ctor**	**Move As-sign-ment**
Overloaded Ctor	No	Yes	Yes	Yes	Yes	Yes
Destructor	Yes	No	Yes	Yes	Yes	Yes
Copy Assignment	Yes	Yes	No	No	Yes	Yes
Move Ctor	Yes	Yes	Yes	Yes	No	No

Defaulting the copy and move assignment operators means the compiler will opt for the most efficient assignment of members.

C) polymorphism

Static polymorphism occurs at compile time and **dynamic polymorphism** occurs at runtime. In the example, dymanic polymorphism occurs at runtime when the payoff function is called. Trade has no specific payoff function, so the payoff function of whichever of the derived classes is called, is then called instead.

D) singleton

A **singleton** is a class that only has one instance throughout the program. To achieve this, the constructor should be made private to prevent it being called. A public **static** method creates the instance. The static keyword ensures that the instance is shared across all future calls.

E) virtual inheritance

Virtual Inheritance needs to be used when a class is derived from 2 classes that are derived from the same class. When this happens, the 2 classes that the derived class inherit from will have 2 of each method from the base class. In this case, we can use virtual inheritance to avoid duplicate methods.