Task 6 Keywords & Polymorphism

class keywords

You should learn the keywords `this,static,const,friend,delete` in class before doint these tasks. You can read the last PPT for more information.

- Find a situation where the `this` keyword must be used.
- Can member functions of a class access static variables, and can static member functions of a class access ordinary variables? Why?

  ```
  class A {
      int x;
  public:
      int &get( ) { return x; }
      int get() const { return x; }
  };
  ```

  Why can't you set the first member function `get` const?

  What are the advantages of giving these two different `get` functions?

- What is `friend` keyword? Does the friendship defined by the `friend` keyword satisfy symmetry, transitivity, and is it inheritable?

Polymorphism

- In the next exercises, we'll explore a set of classes that let you build and modify functions at runtime using tools similar to those in the STL `<functional>` programming library. You need to learn what are abstract classes and how to store functions in classes beforehand.

  Try to explain how virtual functions work and what are abstract funtions in C++.

- Consider the following abstract class:

  ```
  class Function {
  public:
    virtual ~Function() = 0;
    virtual double evaluate_at( double value ) = 0;
  };
  ```

  This class exports a single function, `evaluate_at` , that accepts a `double` as parameter and returns the value of some function evaluated at that point.

  Write a derived class of `Function`, `SimpleFunction`, whose constructor accepts a regular C++ function that accepts and returns a `double` and whose `evaluate_at` function returns the value of the stored function evaluated at the parameter.

- The composition of two functions F and G is defined as $F(G(x))$ – that is, the function F applied to the value of G applied to x. Write a class `CompositionFunction` derived from `Function` and whose constructor accepts two `Function *` pointers and whose `evaluate_at` returns the composition of the two functions evaluated at a point.
- The derivative of a function is the slope of the tangent line to that function at a point. The derivative of a function F can be approximated as $F'(x) \approx (F(x + \Delta x) – F(x - \Delta x))/2\Delta x$ for small values of Δx. Write a class `DerivativeFunction` derived from `Function` and whose constructor accepts a `Function *` pointer and a double representing Δx and whose `evaluate_at` approximates the derivative of the stored function using the initial value of Δx.
- What is slicing problem? Why defining `operator =` as virtual methods won't solve the slicing problem?