Example: The Trapezium Rule for Integration Again...

Using inheritance create a generic integrate function to solve

\[ I = \int_{a}^{b} f(x) \, dx \]

where \( f(x) \) is supplied as an argument to the function. Now use it to solve:

\[ I = \int_{0}^{\infty} \frac{1}{\sqrt{2\pi(1 + x)}} e^{-\frac{x^2}{4\kappa}} \, dx \]

with \( \kappa = 1, 2, \ldots, 10 \).

Solution:

Stage I:
This time we shall make a base class to act as an interface.

```cpp
class Base {
public:
    // pure virtual function operator
    virtual double operator()(double x) const = 0;
};
```

Now we need a function to inherit from the base class, that can be declared and has an implementation to the function prototype in the base class. For this example we would have

```cpp
// integrand function
class Function : public Base {
public:
    // parameter kappa
    double kappa;
    // function operator
```
double operator()(double x) const
{
    return exp(-(x*x)/(4.*kappa))/sqrt(2.*Pi*(1+x));
}
};

Stage II:
Now we may write the integrate function, which accepts the base class as an argument.

```cpp
// function to integrate with the trapezium rule
double trapezium(double a, double b, int n, const Base& f)
{
    // step size
    double h = (b-a)/n;
    // store running sum
    double sum = (f(a)+f(b))/2.;
    for(int i=1;i<n;i++) sum += f(a+i*h);
    return sum*h;
}
```

Now we have encapsulation, so that Function may have its own associated data types, and also polymorphism, since the trapezium function will work on any class that inherits Base and supplies the function ()

### Inheritance

5.1 Recall the Secant method for finding a root:

\[
x_{n+1} = x_n - \frac{f(x_n)}{f(x_n) - f(x_{n-1})} (x_n - x_{n-1})
\]

By editing previous code, or starting from scratch, use inheritance to create a generic Secant method solver to return a root. You may use the Base class declared above.

5.2 Now recall problem 3.6 from examples sheet 3, if you have already written a function or class to produce the value of the investment \(B\), how would you go about completing the problem making use of your generic root finder? Try it out in code...

### Templates

5.3 Write new versions of the trapezium rule and secant method using templates.

5.4 Write the Point class as a template. Search the internet to find the syntax for how to make the Circle class that inherits Point a template as well.

5.5 Add a swap function as a member to the Point template class.

5.6 Try to rewrite the MVector class from the courseworks as a template. What happens if you have overloaded addition or multiplication? Can you find anyway to work around errors that you may receive?