# Lab: Inheritance and Multiple Contracts

## Master | Agent contract

Create a master contract, that:

* Can create agents
* Can give orders to a given agent
  + The agent must be one created or approved by this master
* The master contract owner can tell the master to create or approve an agent
* Can query an agent by asking if the order is done
* Create an agent contract
  + It must have a method to receive and query an order from master and only from it.
* For this case, create one agent which can accept only one order - wait 15 seconds. The order is done (the query function returns true) if 15 seconds have passed since the order placement.

## Inheritance

Write a **SafeMath** **contract** (don’t google for the solution!)

* The contract has **methods to safely add, subtract and multiply** two numbers
* The methods should throw if an integer overflow occurs!

Write an **Owned contract**

* Which knows its owner
* Has method to change the owner (called from current owner)
* Implements an access modifier

Write a **contract** that inherits **SafeMath** and **Owned** and uses their methods

* The contract should hold one **int256 state variable**
* Has a method to **change the state variable** automatically by these rules:
  + Method is **called by the owner**
  + The state is incremented by **now % 256**
  + The state is **multiplied by the amount of seconds since the last state change** (initially 1)
  + The current **block gas limit is subtracted** from the state

## Improve SafeMath

* Take your **SafeMath** contract and
  + Turn it to a library
* Take the contract that uses it
  + Modify it so that it uses the new library
  + Use the “using x for y” directive