Observer pattern

Improve lab7 by using observer pattern to report when a spy's resistance is broken. Add the following abstract class definitions:

```
class Observer;
class Subject {
public:
    virtual void Attach(Observer *o) = 0; // Set
    virtual void Notify() = 0;
};

// Abstract interface of Observer
class Observer {
public:
    virtual void Update() = 0;
};
```

Derive a Spy class from Person and Subject and change interrogate so that when spy's resistance is broken the registered observer is notified. Leave the rest of the spy's as it was in lab7.

Derive a Judge class from Person and Observer. When Judge gets an update the name of the judge is printed plus a text "I'm sending a spy to prison". Note that Judge inherits functionality from Person so you need to implement only constructor and Update().

Change the beginning of the main():

```
Person agent("James Bond");
Spy spy("Emilio Largo", "William Johnson", 3);
Spy spy2("Ernst Blofield", "John Keats", 5);
Judge judge("Judge Barnhill");
spy.Attach(&judge);
spy2.Attach(&judge);
```

Example output (partial):

```
Nice to meet you. My name is: James Bond
Who are you?
My name is: William Johnson
Who are you?
My name is: William Johnson
Who are you?
[My name is: Judge Barnhill
 I'm sending a spy to prison!]
My name is: Emilio Largo
My alias is: William Johnson
```