Adapter Pattern Example

```
// Abstract Target
class AbstractPlug {
public:
  void virtual RoundPin(){}
  void virtual PinCount(){}
};

// Concrete Target
class Plug : public AbstractPlug {
public:
  void RoundPin() {
    cout << " I am Round Pin" << endl;
  }
  void PinCount() {
    cout << " I have two pins" << endl;
  }
};


// Abstract Adaptee
class AbstractSwitchBoard {
public:
  void virtual FlatPin() {}
  void virtual PinCount() {}
};


// Concrete Adaptee
class SwitchBoard : public AbstractSwitchBoard {
public:
  void FlatPin() {
        cout << " Flat Pin" << endl;
  }
  void PinCount() {
        cout << " I have three pins" << endl;
  }
};
```

```
// Adapter
class Adapter : public AbstractPlug {
public:
  AbstractSwitchBoard *T;
  Adapter(AbstractSwitchBoard *TT) {
        T = TT;
  }
  void RoundPin() {
        T->FlatPin();
  }
  void PinCount() {
        T->PinCount();
  }
};


// Client code
void _tmain(int argc, _TCHAR* argv[])
{
  SwitchBoard *mySwitchBoard = new SwitchBoard;//Adaptee

// Target = Adapter(Adaptee)
  AbstractPlug *adapter = new Adapter(mySwitchBoard);
  adapter->RoundPin();
  adapter->PinCount();
}
```