

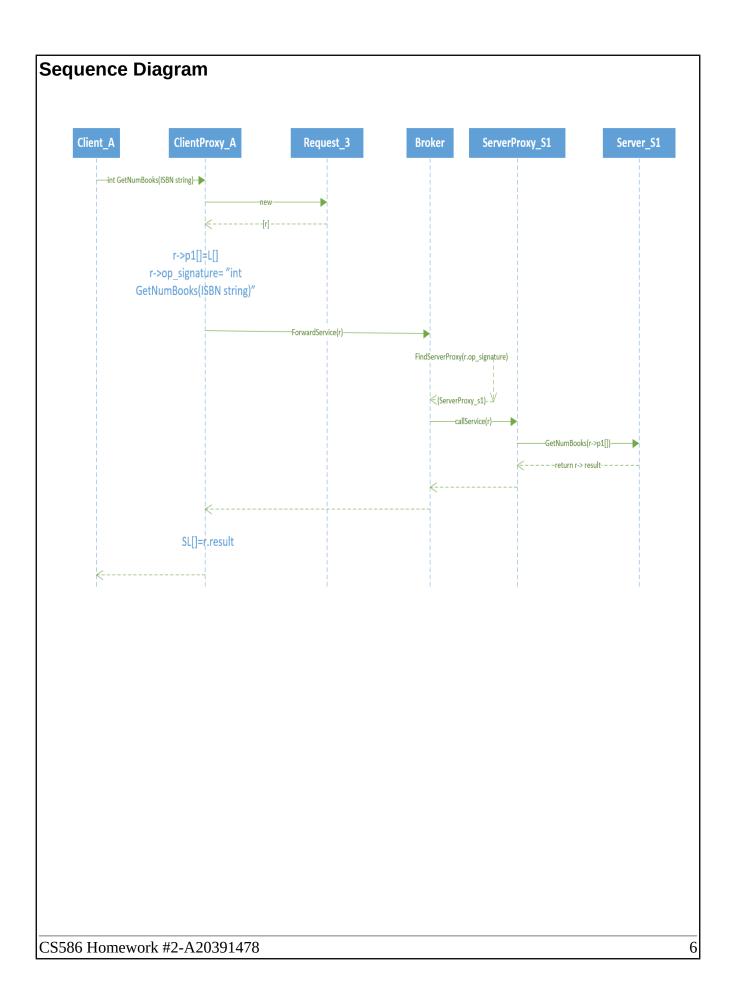
Table of Contents

Problem #1	3
Client-Broker-Server pattern	
Class Diagram	3
Pseudo-code	
Sequence Diagram	
Problem #2	
Adapter Design Pattern	
Association-Based Version	
Class Diagram	8
Pseudo-code	10
Inheritance-Based Version	12
Class Diagram	
Pseudo-code	
Problem #3	
Abstract Factory Pattern	16
Class diagram	16
Pseudo-code	
Sequence Diagram	18

Problem #1 Client-Broker-Server pattern Class Diagram -string p1 -string p1 -string p1 -string p1 -int result int result -int p2 -string op_signature -ServerProxy *sp -ServerProxy spList[][] request *r +Register(ServerProxy sp, string signature) +Unregister(ServerProxy sp) +FindServerProxy(string op_signature) -Broker *brk -Broker *brk +ForwardService(Request r) -request *r -Request *r +callService(Request r) -server_S1 *svr -server_S2 *svr +void AddBooks(ISBN string, +void InsertBooks(ISBN string, +callService(Request r) +callService(Request r) warehouse int, n int) warehouse int) +void DeleteBooks(ISBN string, +void RemoveBook(ISBN string, warehouse int, n int) +int GetNumBooks(ISBN string) warehouse int) +int GetNumBooks(ISBN string) +int IsBook(ISBN string) +int IsBook(ISBN string) +void AddBooks(ISBN string, +void InsertBook(ISBN string, warehouse int, n int) warehouse int) +void DeleteBooks(ISBN string, +void RemoveBook(ISBN string, -ClientProxy *cp -ClientProxy *cp warehouse int, n int) warehouse int) +int GetNumBooks(ISBN string) +int GetNumBooks(ISBN string) +int IsBook(ISBN string) +int IsBook(ISBN string) 3 CS586 Homework #2-A20391478

```
Pseudo-code
Class Broker
       ServerProxy *sp
       ServerProxy spList[][]
       request *r
       //Functions
       Register(ServerProxy sp, string signature)
               sp → Serverproxy spList[] // add server to serverproxy spList
               add signature to serverproxy spList[]
       unregister(serverproxy sp)
              remove sp from serverproxy spList
       FindServerProxy(string op_Signature)
       //For every server in the serverproxy spList
               if serverproxy spList[op_signature]
              return Serverproxy
               end if
       ForwardService(request r)
               server proxy p1=Findserverproxy(r.op_signature)
                      if(p1!=null)
                              sp \rightarrow callservice(r)
                      end if
       }
       Class clientproxy_A
               void AddBooks(string ISBN,int warehouse, int n)
               r= Request_1
               r → op_signature = "void AddBooks(string,int,int)"
              r \rightarrow p1 = ISBN
              r \rightarrow p2=warehouse
              r \rightarrow p3=n
              b \rightarrow Forwardservice(r)
       Void RemoveBook( ISBN, int warehouse)
              r=Request_2
               r → op_signature="Void RemoveBook(string,int)"
```

```
r \rightarrow p1 = ISBN
                r \rightarrow p2=warehouse
                b \rightarrow ForwardService(r)
        int GetNumBooks(string ISBN)
                r=Request_3
                r → op_signature="int GetNumBooks(string)"
                r \rightarrow p1 = ISBN
                b \rightarrow ForwardService(r)
                 return r \rightarrow result
        }
        int IsBook(string ISBN)
                r=Request_4
                r → op_signature= "int IsBook(string)"
                r \rightarrow p1 = ISBN
                b \rightarrow ForwardService(r)
                 return r \rightarrow result
        }
Class ServerProxy_S2
        Server_s2 *svr
        CallService(Request r)
                 if(r → op_signature="Void InsertBoot(string,int)")
                         S2 \rightarrow insert the book
                else if(r → op_signature="Void RemoveBook(string,int)")
                         S2 \rightarrow \text{remove the book}
                else if(r → op_signature="int GetNumBooks (string)")
                         return r \rightarrow result = S2 \rightarrow GetNumBooks(r.n)
                else if(r → op_signature="int IsBook(string)")
                         return r \rightarrow result = S2 \rightarrow IsBook(r.n)
```

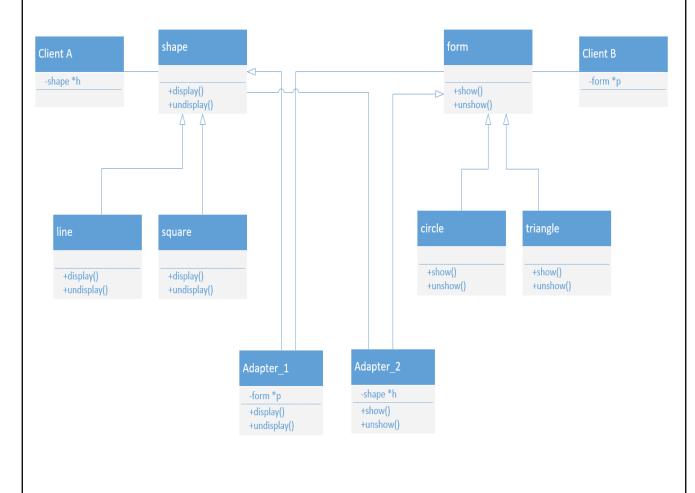


Problem #2

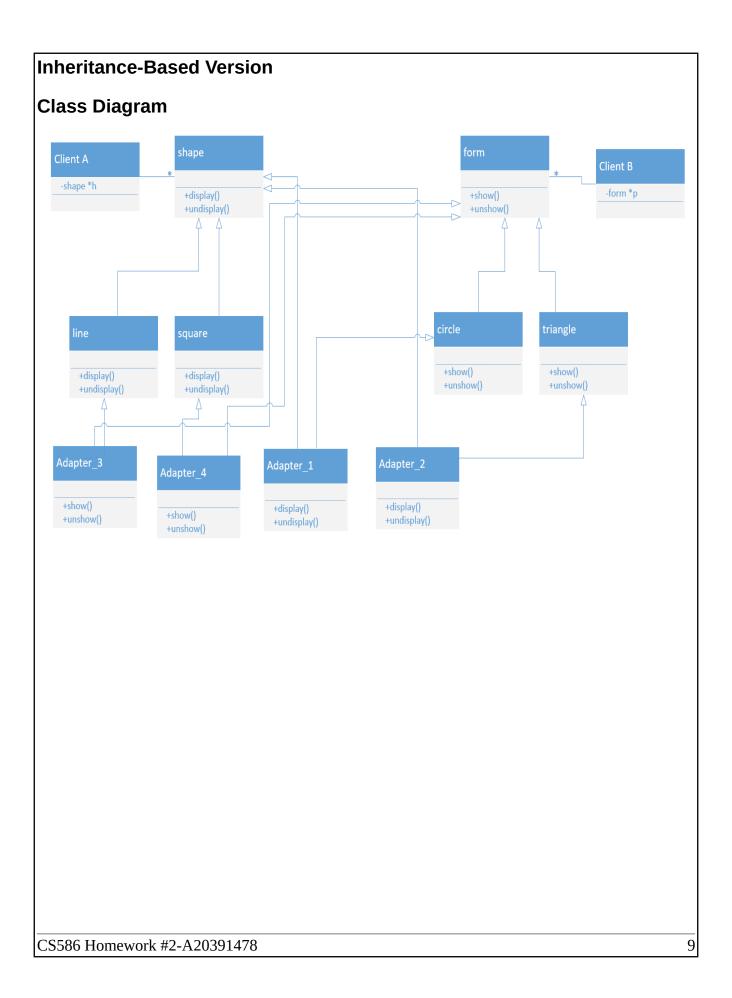
Adapter Design Pattern

Association-Based Version

Class Diagram



```
Pseudo-code
class Adapter_1
        Form *p
        display()
                p \rightarrow show()
        undisplay()
                p \rightarrow unshow()
class Adapter_2
        Shape *h
        show()
        {
                h \rightarrow display()
        unshow()
                h \rightarrow undisplay()
        }
```



```
Pseudo-code
Class Adapter_1
      display()
             show()
      undisplay()
             unshow()
Class Adapter_2
      display()
             show()
      undisplay()
             unshow()
Class Adapter_3
      show()
             display()
      unshow()
             undisplay()
Class Adapter_4
      show()
             display()
      unshow()
             undisplay()
CS586 Homework #2-A20391478
                                                                                          10
```

Problem #3 **Abstract Factory Pattern** Class diagram -s1:Server1 -s2:Server2 +sort() +search() +sort() +search() MergeSort +getsort() +getsearch() -search() -search() +sort() +sort() +getsort() +getsort() +getsearch() +getsearch() CS586 Homework #2-A20391478 11

```
Pseudo-code
AbstractFactory

class clientARequest
{
    getsort()
    {
        return new heapsort // return object of HeapSort
    }
    getsearch()
    {
        return new binary search // return object of BinarySearch
    }
}

class clientBRequest
{
    getsort()
    {
        return new mergesort // return object of MergeSort
    }
    getsearch()
    {
        return new mergesort // return object of LinearSearch
    }
}
```

