Monday
15-6-2020

DP Assignment-5

UNIT-5

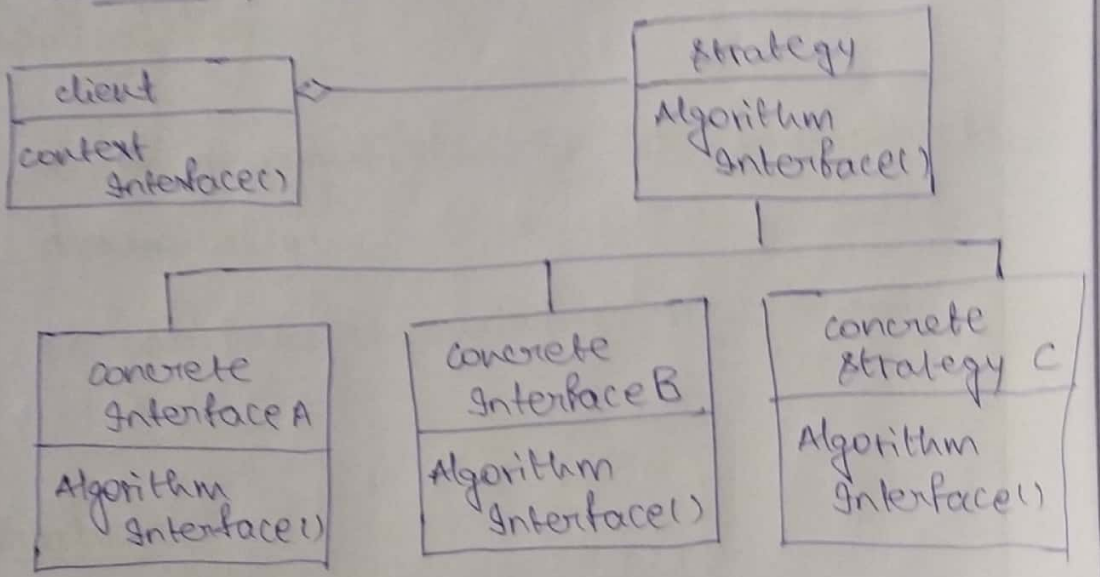
N.V.S.K. Kalyani
17H41A05H4
3rd B.tech CSE-C

2 marks

1A) Consequences of State pattern:-

* Localize state-specific requests.
* primary Interface for clients.
* State objects can be shared.

2A) Structure of strategy pattern:-



3A) Implementation of state pattern:-

* who controls the Iteration
* who defines the traversal algorithm
* How robust is the Iterator
* Iterators may have priviledge access.
* Iterators for Composite.

4A) Intent of strategy pattern:-

Define a family of algorithm Encapsulate each one and make them Interchangable. Strategy lets the algorithm vary independently from client

that use it.

Applicability:-

* Many related classes differ only in their
  behaviour.
* you need different varients of an Algorithm.
* An Algorithm uses date that client shouldn't
  known about.

5A) Design patterns provides a standard termino-
-logy and are specific to particular Scenario.
Ex:- A singleton design pattern signifies use of
single object so all develops familar with single
design pattern will make use of single object and
they can tell each other that program is follo-
-wing a singleton pattern.

174N1A0584

10 Marks

1A)

TEMPLATE METHOD

Intent :- Define the selection of algorithm in an algorithm operation, determing some steps to subclassed.

Motivation:-

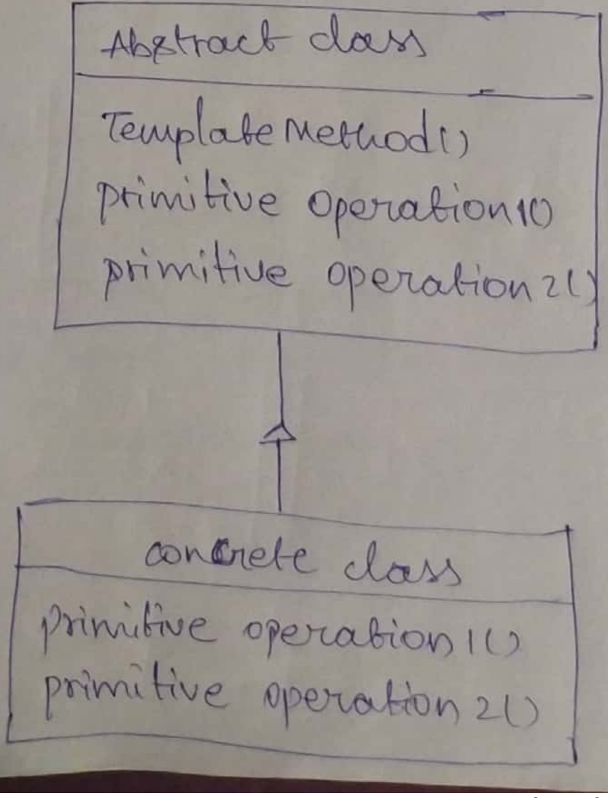
* consider an application framework that provide application and Document classes.

* Ex:- A Drawing application defines draw documen subclass a spread sheet application defines sprea sheet Application & Document subclasses.

Applicability:-

* To Implement the Invarient parts of an Algorith
* To control subclasses Extension.

Structure:-



Scanned with CamScanner

participants :-

1. Abstract class (Application)
2. concrete class (My application)

Collaboration :-

concrete class relies an abstract class to implement the invarient steps of an algorithm

Consequences:-

* concrete operations
* Concrete Abstract class operations.
* primitive operations
* Factory methods

Implementation:-

* Minimizing primitive operations
* Naming conventions

known uses :-

* Defining Templates
* processing the software application

Related patterns:-

* Factory method
* Strategy

17UMIA05?4

2a) Visitor pattern:-

Intent: Represents an operation to be performe on the Elements of an object structure

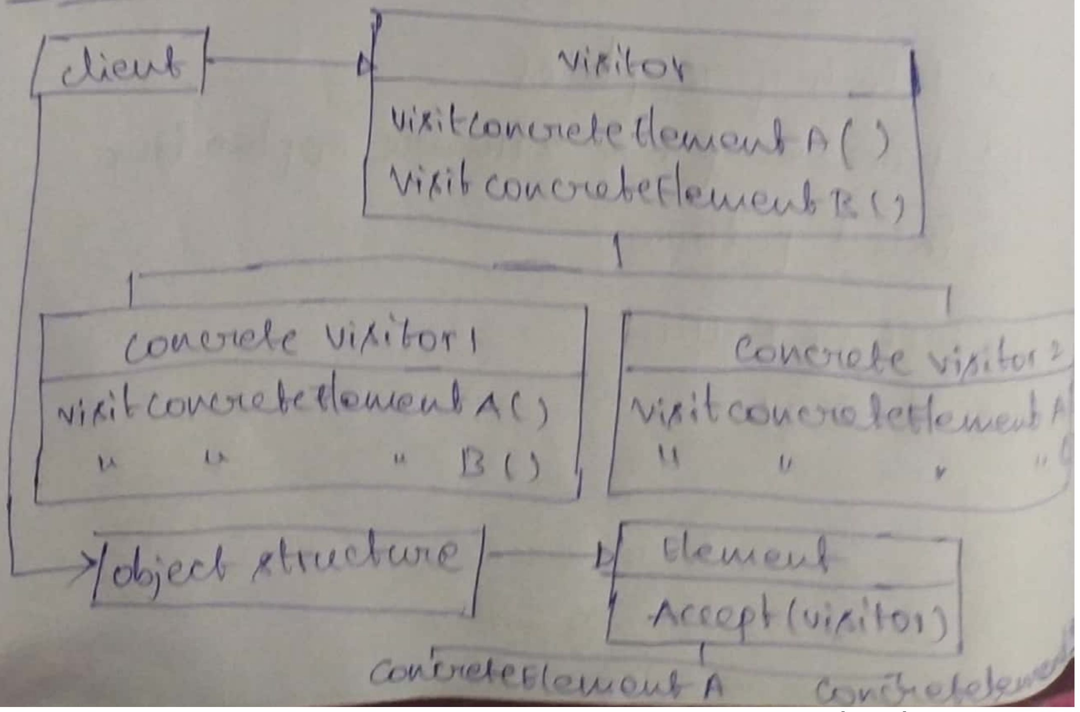
Motivation:-

* This diagram show part of the Node class - hierarchy.
* The problem here is that distributing all the operations across the various node class leads to a system that hard to understand maint and change.

Applicability:-

* An object structure contains many classes of objects with differing Interfaces.
* The class defining the object structure rare change.

Structure:-



participants:-

1. visitor (Node visitor)
2. concrete visitor
3. Element (Node)
4. concrete Element (AssignmentNode, VariableRefNode)

collaboration:-

* A client that uses the visitor pattern must
* create a concrete visitor object and then traverse the object structure, visiting each Element with the visitor.

consequences:-

* visitor makes adding new operations Easy.
* A visitor gather related operations & seperate unrelation ones.
* visiting across class hierarchies.

Implementation:-

* Double dispatch
* who is responsible for traversing the object structure.

Sample code:-

```
public abstract class visitor
{
    Concrete Element A();
    Concrete Element B();
}
```

```
public class concrete visitor Extends visitor
{
  public void concrete Element()
   {
      //          public abstract class Element
   }  ———————→   { public ~~void~~ accept (visitor);
}                }
public void concrete Element Extends Elements
{
  public void Accept()
   {
     //
   }
}
public class client
{
   visitor v;
   //Implement the code
}
```

Known uses:-

* Arthimetic calculations
* Traversing
* compiler

Related patterns:-

* composite
* Interpreter