17-6-25. DESIGN PATTERNS:--7. A design pattern in Programming is a reusable Solution to a common Problem that occur during Software development of design sido ansusted yearwoods de desert -> These Patterns Provide structured approach to solve design & development issues. -7 used to exeate move maintainable, flexible & Scalable code Design Pattexns Slobal Points of acres to it Creational Patterns Structural Pattern Behavioral Pattern - Singerton - Adaptes - Observe - Composite - Stratergy - Factoxy - Builder [i] (seational pattern: These pattern four on object execute mechanism. -> Provide ways to execute object in a matther that is flexible and maintanable. -> Based on application specific requirements we choose a object exection mechanism. [2] Structural Pattern: -> These Patterns focus on class & objects to execte large structures while keeping them flexible and efficient. to ensure that classes and objects

[3] Behavisol Pattesn:-- used for intraction of communication between object and classes. -7 Provide solution for officiently manage flow of Control, behaviour between object. * (Seational Pattesn: Til Singerton Pattexn: --7 Ensures that only one instance of a class is executed of provide global point of access to it. -7 used when we want to have single shared instance of class throught the application. ways to execute singleton class :-[1] Eagex intialisation [4] Double char [i] Laty intialisation [5] Bill Pugh Solution [3] Synchronized block. [6] Frum Mair class: - , vages alliones maid miles no Lossy see Public class main mains as on missex souls Public static void main (String ages []) ou class 4 sb }cus DB Connection obj = DB Connection get Instancecy; especial bas message that most of su samp in of a scholar of congression estagest would not

```
Eages Intialisation :-
Public dass DB Connection
Private Static DB Connection canobic et = new DB Connectiona
   Private DR Connection () { 3
     Private Static DB Connection get Instances
        xeturn conobiect;
 Laty intialization:
 Public class DB Connection
     PXIVATE Static DB Connection conObject,
     Private DB Connection C) [].
     Public static DB Connection get Instances
              if (conobiat = = null)
                       con Object = new DB Connections.
            return conobiect
      4
   Synchronization Block:-
  Public class DB Comection
    Private static DB Connection conobicet
       Private DBConnection CD { ].
        Synchronized Public Static DB connection get Instances
              if (conobject = = nui)
                      Con Object = new DB Connection ()
              se toxn conobject
```

```
Double check locking system :-
   Public class DB Connection
     · Private static volatile DB Connection con = new DB connection
       Private DB Connection 0 } }
       Poblic Static DB Connection get Instance()
             Synchronize d (DB Connection . class)
                    if (con = = nun)
                        Con = new DB connection co
              Seturn con.
  Big Push singleton: (uses cagex
    Public class DBC
   Private DBC CD { }
    Private static class DBHelper
        Private Static final DB = INSTANCE_OBJECT= New DB
     Public static DB ger Instances
          return DDHERES. INSTANCE_OBJECT
    ENUM:-
    Public enum DBC
       INSTANCE
       public static do Something cof
```

Buildes Pattern: -- Buildex Pattorn is a excational design Pattorn that allows you to construct complex objects step - by-step Serexating the Const xuction logic from the representation -7 useful when objects have many optional fields -> Avoid telescoping Constructors. with builder Patton Without Builder Pattern Person. Java. Pexson. Java: -Public class Person Public class Person 1 Private String name; Private int age; private String name; Private Person (Builder builder) Private int age; this name = buider name Public Pexson (String name, int age) this age = builder age this . name = name . Public void display col this. age = age; 5.05 (name+ ...). Public void display c) Public Static class Builded Private String name S.O.P ("Name" + name + . .) Private int asc Public Buildon set Name (String & -this. name = name; I Cst. Java: return this. Public class Test Public Buider set Age (int age) 5 -this. 000 = age Public static void main (String angse) setusn this Person P=new Person ("Vixat", 35) Public Person buildent return new Person (this P. display (). Inside main Person Penew Person . Builderc) · setName ("Kohli") · Set ABC (35) · build co -P. displayez

```
tactoxy Pattern :-
  - A creational design pattern that provides on interform
   to execute objects in a superclass but allows
  subclass to alter the type of objects that
   will be exected.
   -7 Avoid using new discetty in the client
   -> The will use factory Pattern when we need to exem
     object based on input ox condition
  _ Code :-
   Public intextace shape {
    void drawcz;
  Private Person ( Billion & ...
   Public class circle implements shape {
          Public void draw co of
  system.out. Println ("Dxawing a circle).
  Public class square implements shape of
        Public void draw () of
     2 System. out. Println (" Drawing a senare")
   TOWN TO SOURCE STORY
Public class share Factory
    Public share ger Share (Soving type)
  to sind a se if (type. equal s I prove (ase ("elxcie)) {
               - Ketush new Circleco.
             y else if Ctype. equals Ignore Case (square) {
            Keturn new Square O.
```

```
MOIN Java
Public class Main
     Public static void main (String [] angs)
       Shape Factory factory = new Shape Factorye
           Shape si = factory, gershape ("circie")
   51. d&aw C)
            shape se = factory, get shape (square)
            52. 48awc 2.
Structral Pattern :-
 Adaptex
 -7 Adapter Pattern is a structural Pattern that allows
   object with in compatible intexfaces to work together
   by convexting one intexface into another.
 -> Bridge the gap between two in compatible interfaces.
          Client -> Target' (exported intexface)
            Adaptes MANAGER
                         A dapter (incompatible class)
  Code:
  class old Printer 2
         Public void PaintOld Co &
           System . out . Paintle ( old printer)
   3
```

```
intexface Printer &
       Void Printez
 class PrinterAdapter implements Printer 2
       Psivate OldPsintes oldPsintes
       Public Printer Adapter (Old Printer) {
      this old Printer = old Printer
      Public Void Print co {
         old Printer. Printold co;
  Public class Test 2
      Public static void main (String () 20195) {
     old printer old = new old Printercy;
      Printer adapter = new printer Adapter (01);
           adaptes. Printcy.
[2] Composite Pattern:-
 -7 used to treate individual objects and groups of
   objects in a uniform way.
      Component (intexface)
                Composite
                 sold principle of the states
     Contains List < Comforent >
```

```
Public intesface Employee 2
       void show Details C.).
3
public class Developer implements Employee {
       Private String name; commission man
        Public Developer (String name) {
         this name = name;
       Public void show Petails () 2
             System. out . pxintln ("Develope": " + name);
 Public class Manages implements Employee 2
       Private String name;
      Private list < Employee7 team = new Assay List < > Co;
       Public Manages (String name) {
             this. name = name;
      3
      Public void add (Employee emp) {
             team. add (emp);
      7
      Public void show Petails () &
            System · out · Paintin ("Manager : ", +name);
             for (Employee e: team) [
                  c. Show Details Co.
   Public class Test &
       Public Static void main (String CJ axSS) [
             Developes devi = new Developes ("Rahui").
             Developer dev 2 = new Developer (" Kixan");
```

manager manager = new Manager ("virat") manages add (devi); manager. Show DetailsCo, Smen paisse station (Smon gairs) Yogosey Behav Public Void showPerails of { System. out. Pxintin ("Developes:" + name). troblic class Manages implements Employee f poisone most Kookoloko Kom snew Public emanages (Simo name) I ame

```
Behavisol Pattern -
observe Pattern
-7 where an object maintains a list of dependents
  and notified them automatically when its state
  changes.
  Public intexface observer 2
        void update (string message);
 Public class Follower implements observer &
       Private String name;
       Public Follower (String name) 2
             this name = name;
       Public void update (String message) [
           System. out . Printly (name);
       7
  Public interface Subject &
       void add Observer (Observer o);
       voi à remove Observer (observer o);
       void noti fyobserver (string msg);
 7
 imPoxt java. util. *;
 Public class channel implements subjected
    Private List cobservery observers = new Assay Listery
     Public void addobsorvor (observer o) {
             observs . add (0);
    3
```

Public class Test [Public Static void main (String [] axys) { channel chanel = new channeles; Atote Sti and w Follower fi = new Follower ("vivab) channel · add Observer (FD) channel . UPload ("Highlights"). void update (seving message); Public class Follower ; mplements observer f Species our printer from Poblic interface Subject of (6 tovassed) keyes do blo singe