group:

1- esraa Mohamed Hashish 12

2- amira Mohamed Fathy 14

3- rewan Alaa Eldin 23

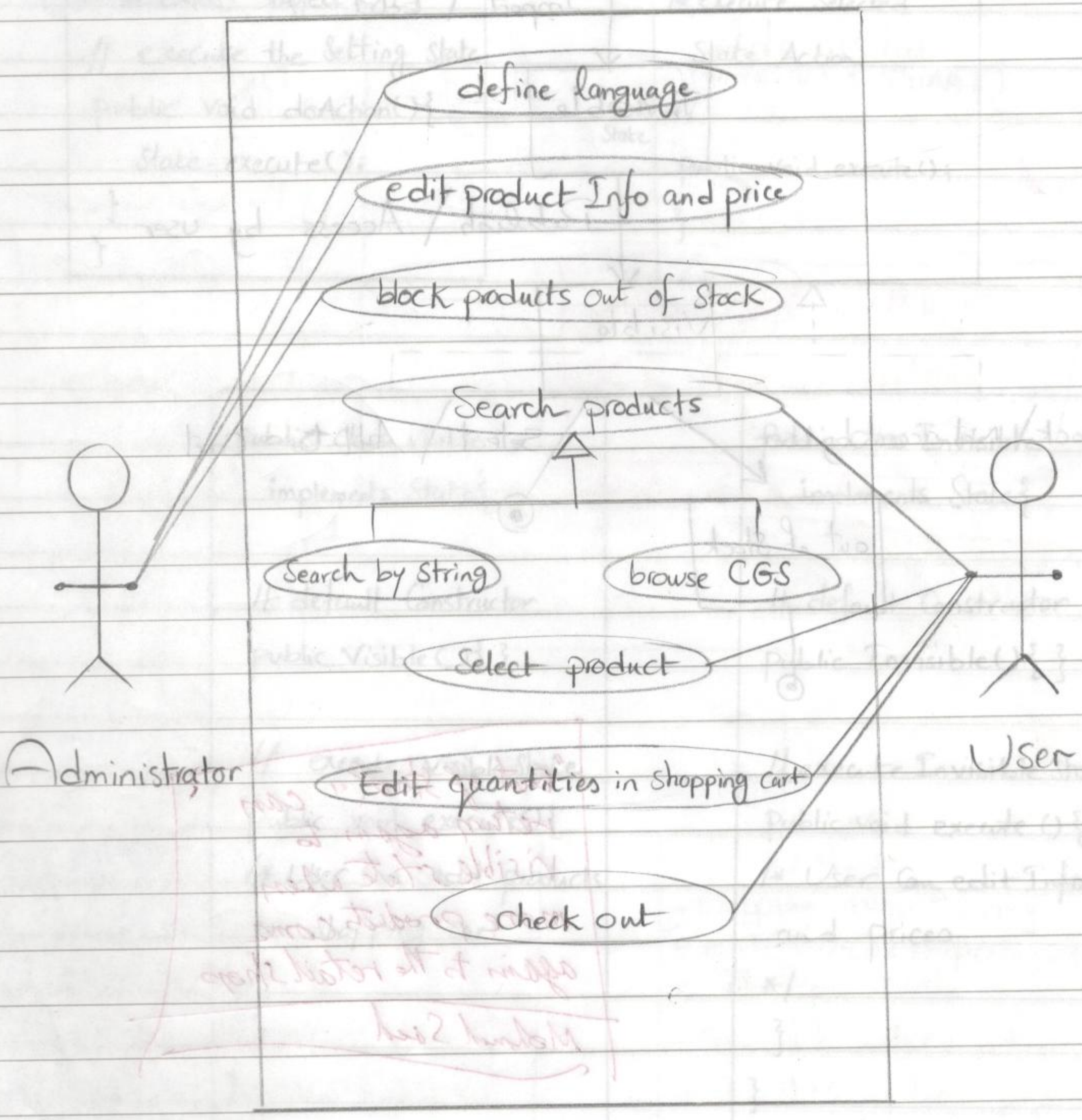
4- Mayar AbdElAziz 69

5- Amr mahmoud Bekheit 45

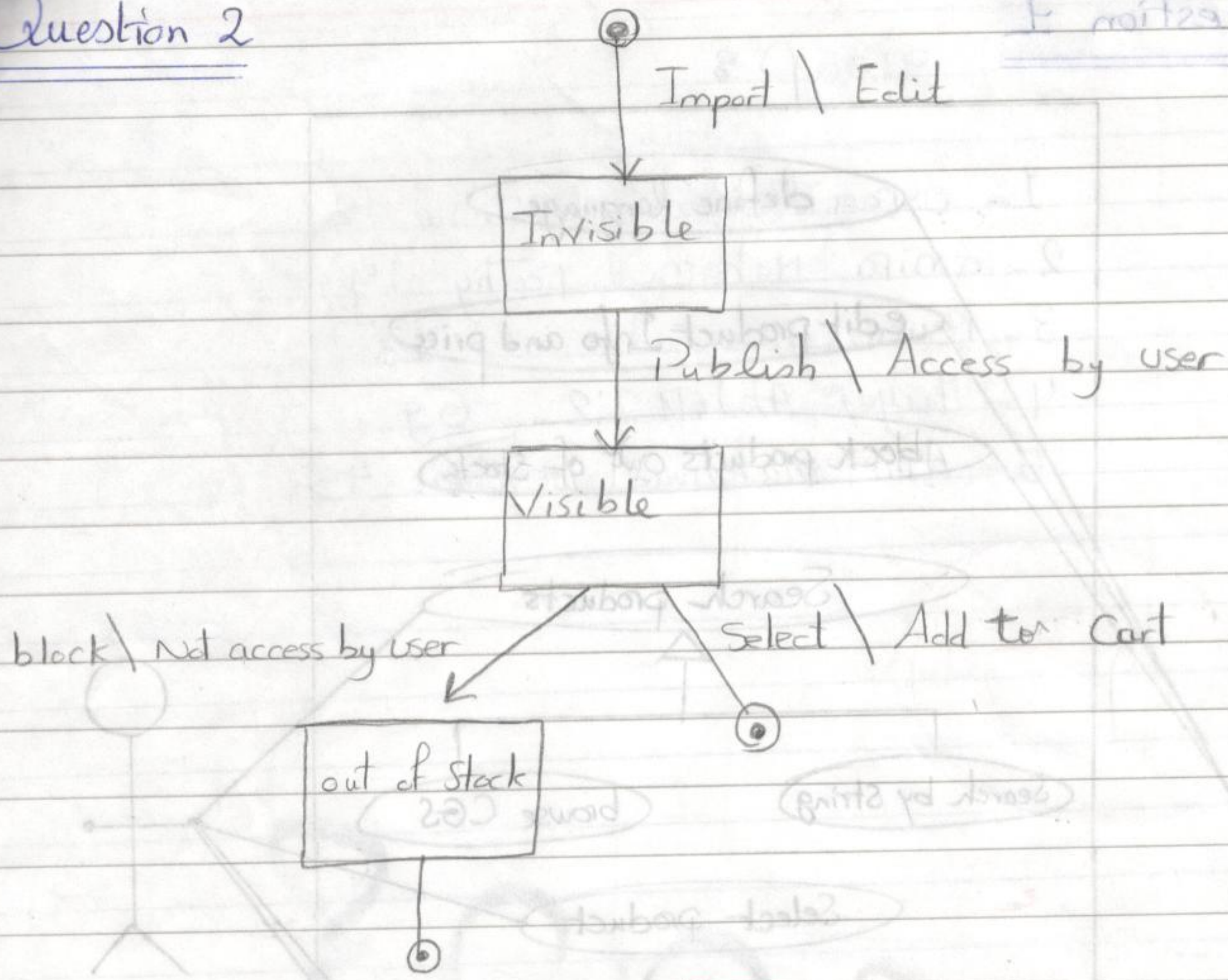
B

2008

Question 1

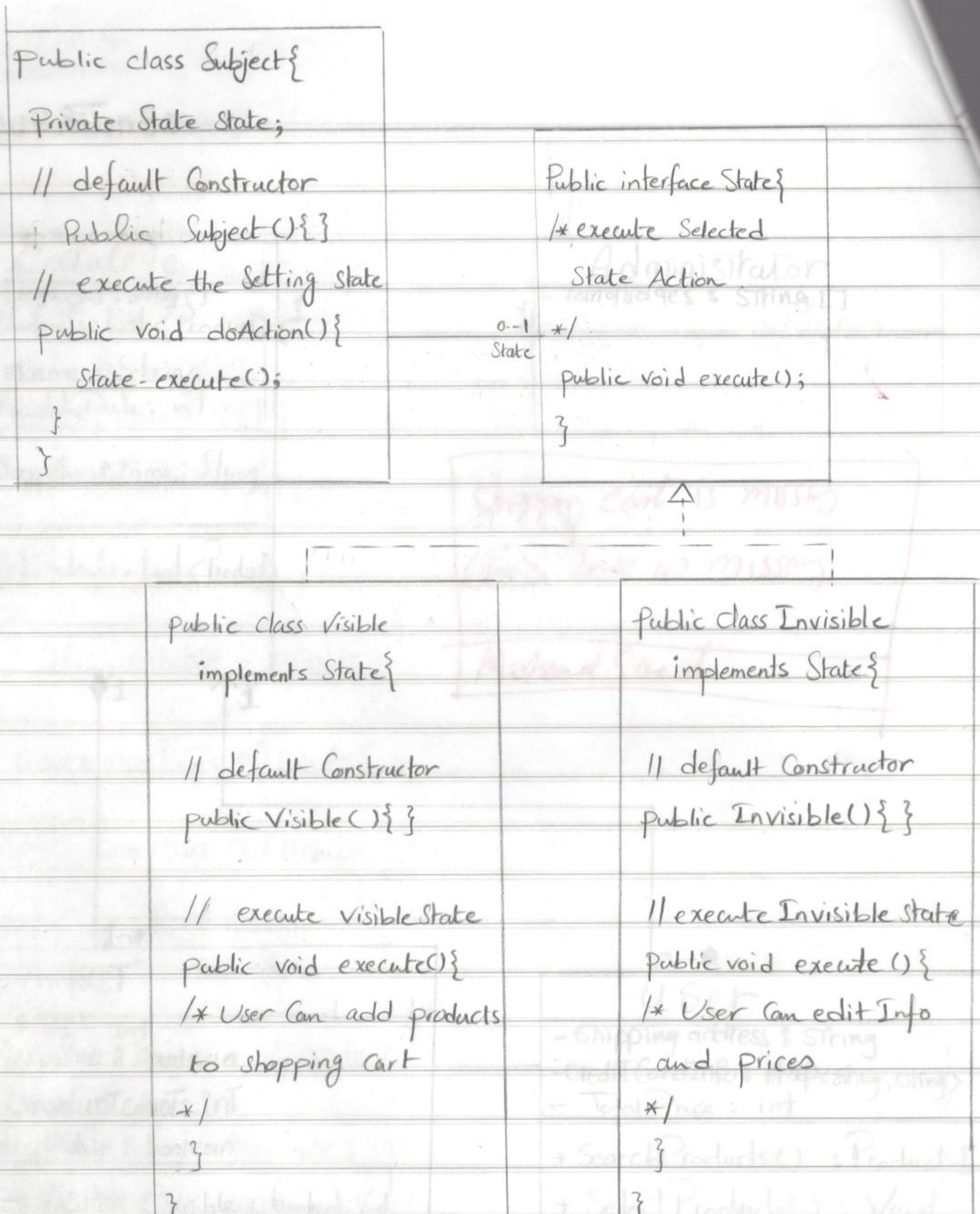

define language
edit product Info and price
block products out of Stock
Search products
Search by String
browse CGS
Select product
Edit quantities in shopping cart
Check out
Administrator
User

Question 2



"Out of stock" can return again to visible state when more products come again to the retail shops

Mohmed Saed



```
Public class Subject{
Private State state;
// default Constructor
 Public Subject(){}
// execute the Setting state
public void doAction(){
   state.execute();
 }
}
```

```
Public interface State{
/* execute Selected
  State Action
*/
 public void execute();
}
```

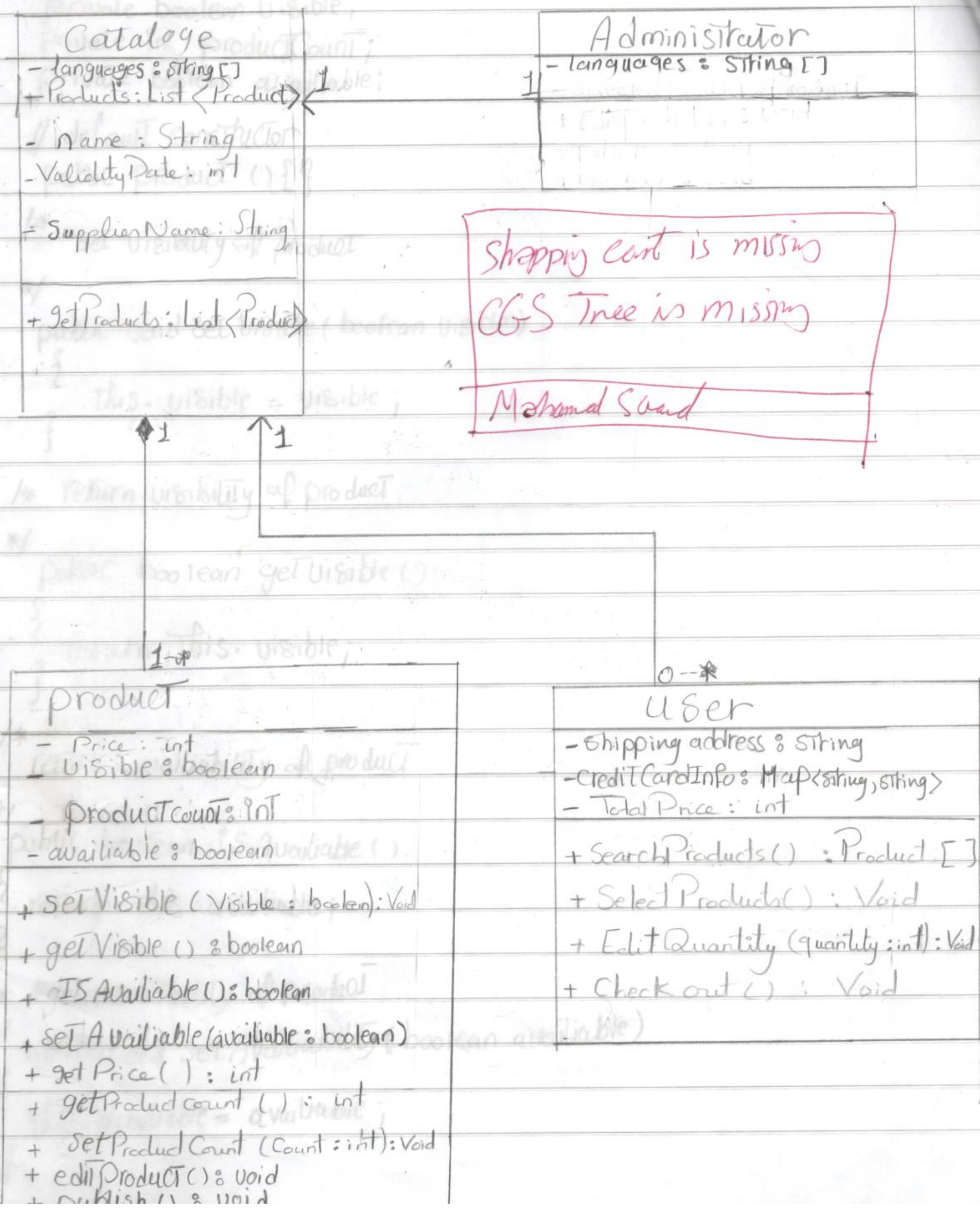
```
public Class Visible
  implements State{

 // default Constructor
 public Visible(){ }

 //  execute visible State
 public void execute(){
 /* User Can add products
 to shopping Cart
 */
 }
}
```

```
public Class Invisible
  implements State{

 // default Constructor
 public Invisible(){ }

 //execute Invisible state
 public void execute(){
 /* User Can edit Info
  and prices
 */
 }
}
```

We need Java code
not UML here

Mohamed Saad

Question 3:


Cataloge
- languages : string []
+ Products : List <Product>
- Name : String
- ValidityDate : int
- SupplierName : String
+ getProducts : List<Product>
1
1
1
Administrator
- languages : String []
1
1-*
product
- Price : int
- Visible : boolean
- ProductCount : int
- availiable : boolean
+ set Visible (Visible : boolean) : Void
+ get Visible () : boolean
+ IS Availiable () : boolean
+ set Availiable (availiable : boolean)
+ get Price ( ) : int
+ getProduct count ( ) : int
+ setProductCount (Count : int) : Void
+ edit ProducT () : void
+ publish () : void
0--*
user
- Shipping address : String
- CreditCardInfo : Map<String, String>
- Total Price : int
+ SearchProducts () : Product []
+ Select Products () : Void
+ EditQuantity (quantity : int) : Void
+ Check out () : Void

Shopping cart is missing

CGS Tree is missing

Mohamed Saad

Question 4

```
public class Product {

  private int price;
  private boolean visible;
  private int productCount;
  private boolean availiable;

  // default constructor
  public product (){}
 /*
    set visibility of product
 */
  public void setVisible( boolean visible )
   {
      this. visible = visible ;
   }

 /* return visibility of product
 */
  public boolean getVisible ()
  {
     return This. visible;
  }
/*
  return availiability of product
*/
 public boolean isAvailiable ()
 {
  return This. availiable ;
 }
/* set availiability of product
*/ public void setAvailiability( boolean availiable)
 {
  This. availiable = availiable ;
 }
```

```
eturn price
of product

ublic int getPrice(){
  return This.price;
}

return The number
of product

ublic int CountProduct(){
  return This.productCount;

set The number
of product

ublic Void SetProductCount (int productCount)
{
  This.productCount = ProductCount;
}

* edit product */
ublic Void EditProduct()
{ // edit The price,---etc
  This. setvisible(false);
}
 publish product */
ublic void publish(){
 This. set Visible (True);
 This. setAvailiable (True);

 block The product That is out of stock */
public Void block()
{
  This. set Availiable (false);
}
```

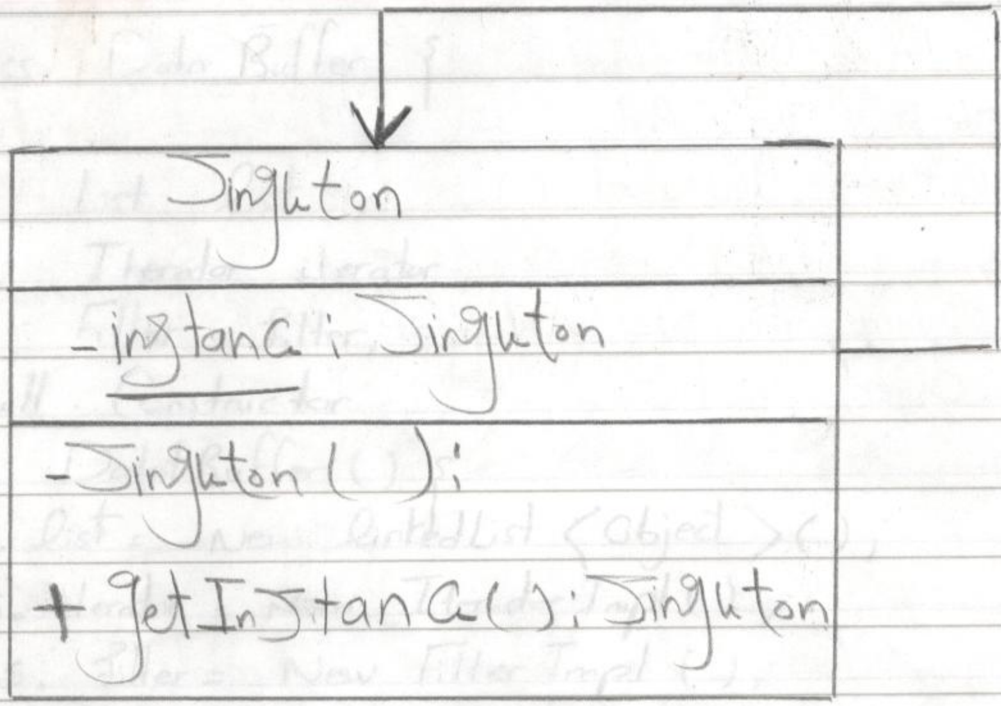
```
Public Class User {
 Private List < Product > list;
Private String Shipping Address ;
 Private Product []Products= New Product [20] ;
 Private Map < String, String> CreditCard Info;
 Private List < Product > Shopping Cart ;
 Private int totalPrice ;
 Private Cataloge cataloge ;
 Private Product Selected Item;
  // Default Constructor
 Public User ( String Shipping Address , Map<String, String> CreditCard){
  This. Shipping Address = Shipping Address ;
  This. CreditCardInfo = Credit Card;
  totalPrice = 0;
  Cataloge = New Cataloge ;
  Shopping Cart = New LinkedList <Products> ();

 /*
 Iterate in the list of Products from the Cataloge and
 */

 Public Product [] SearchProducts ( ) {
 list=Cataloge. get Products () ;
  //ittate through cataloge
 return Product;
}

/*
Select A Product from the Result Set and Add it in the
Shopping Cart if aVailable
*/
 Public Void SelectProducts ( ) {
 // Select the product from the Products Array

  // check Availability of the Product
  if (Selected Item. Is Available( )) {
        Shopping Cart. add ( Selected Item);
        total Price + = Selected Item. get Price() *
                   * Selected Item. get ProductCount () ;
  }
 the quantity of the Selected Product and Increment the total Price

ublic Void EditQuantity ( int quantity) {
   int Count ;
   Count = Selected Item. getProduct Count ();
   Selec ted Item. SetProductCount ( Count + quantity) ;
}

/*
 Pay for the purchased products
*/

 Public Void CheckOut ( ) {
   // Pay for the totalPrice using Credit Card
 }
```

```
pablic Class Administrator {

private Cataloge cataloge;
 private siting [] language;
// de fault consitructor
 public Administrator()
 {
   Cataloge = new Catalog ( language);
 }
```

```
public Class Cataloge {
 private string[] language;
 private List<product> products;
 private string name;
 private int ValidityDate;
 private string supplierName;
 // default Constructor
  public Cataloge() { }

// Constructor
  public Cataloge(string[] language){
   This.language = language
  }
/* return list of products
   Contained in The Cataloge
*/
 public List<products> get Products()
 {
   return This. products
 }
```

QUESTION 6 : a

Singleton Design Pattern UML Diagram



Singleton

-instance : Singleton

-Singleton( );

+getInstance( ): Singleton

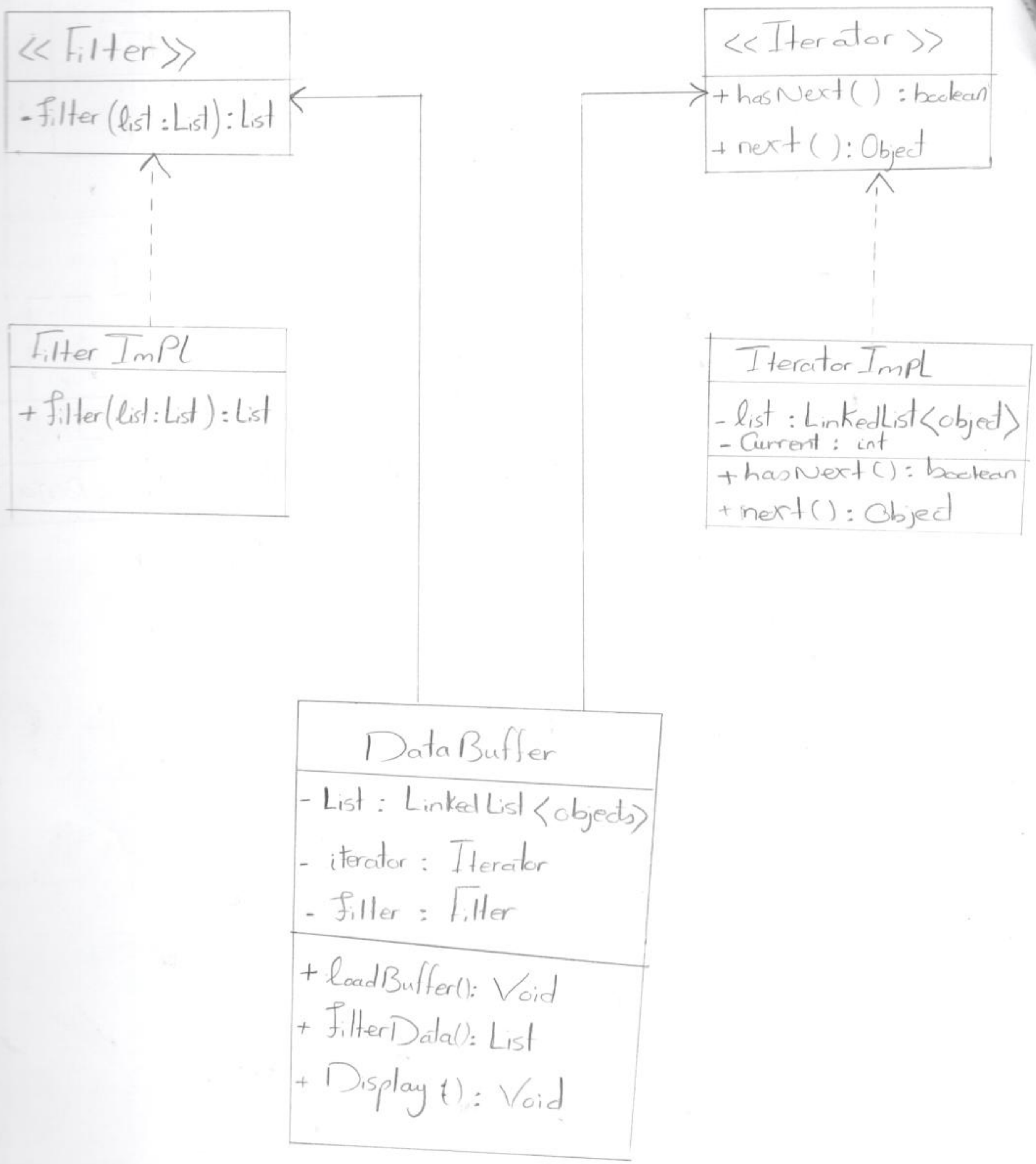
Question (5)

```
Public Class  Data Buffer {

    PriVate  List  list  ;
    PriVate  Iterator  iterator;
    PriVate  Filter  filter;
    // Default  Constructor
    Public  DatatBuffer () {
        This. list =  New  linkedList <Object> ();
        This. iterator = New  Iterator Impl () ;
        This. filler = New  Filter Impl () ;
    }

// load the Data from the memory
    Public Void  LoadBuffer () { } ;

// Returns the filltered data according to the Price

    Public  List<objects> FilterData () {

        // filter the Data and adds them to the list
        return list;
    }

// Display the the filltered data on the Screen
    Public Void  Display () { } ;
Public interface Iterator {
    // Checks if the list has anext element or Not
      Boolean hasNext ();

    // Return the Next element of the list
      Object next ();
}


 Public Class IteratorImpl implements Iterator {
      Private List list;
      Private int Current;

    // Default Constructor

      Public IteratorImpl ( List list <objects> ) {
      This. List = list;
      Current = 0;
    }

   // checks if the list has a Next Element
      Public Boolean hasNext ();

   // Returns the Next Element of the list
      Public object next () {
        return list. get (Current)
      }
```

```
Public interface Filter {
    Public List <Product> filterData (List< Product> Produc
}
Public class   PriceFilter implements Filter {
    @override
    Public List <Product> filterData( List <Product> Products){
      Iterator it = Products.getIterator( );
      List<Product> Selected = new List<Product> ( );
      while (it.hasNext( )){
          Product product = it.next( )
          if (Product.getPrice( ) < limit){
             Seleted.add(Product) }
          if (Selected.size( ) == 20)
               break;
      }
      return Selected;
  }
Public interface Iterator {
 Public boolean hasNext( );
 Public Product next( );
}
Public class ProductsList {
   Private Node first;
   Private class ProductsIterator implemints Iterator {
       boolean hasNext ( ) {
           if(first.next ≠ null) return true;
```

```
se return false

Product next ( ) {
   If ( hasNext( ) { first = first.next( ); return first.Item( );
   return null;
 }
Public Class Data Buffer {
 Private List < Product > Products, filteredProducts ;
 Private Filter filter = new Pric Filter ( ); // or any other
                                                Type of
                                                fiters
Public Void load Buffer ( ) {
      // load Data from DB
      // Set Products list
     // assumption: load first 100 Product and then
    // make filtering on it to select the top 20 Products

 }

ublic Void filter Data ( ) {
       filteredProducts = filter.filterData(Products);
  }

Public void display ( ) {
       // loop through filteredProducts list and display
     // them on the web Page
}
}
```

Question ⑤

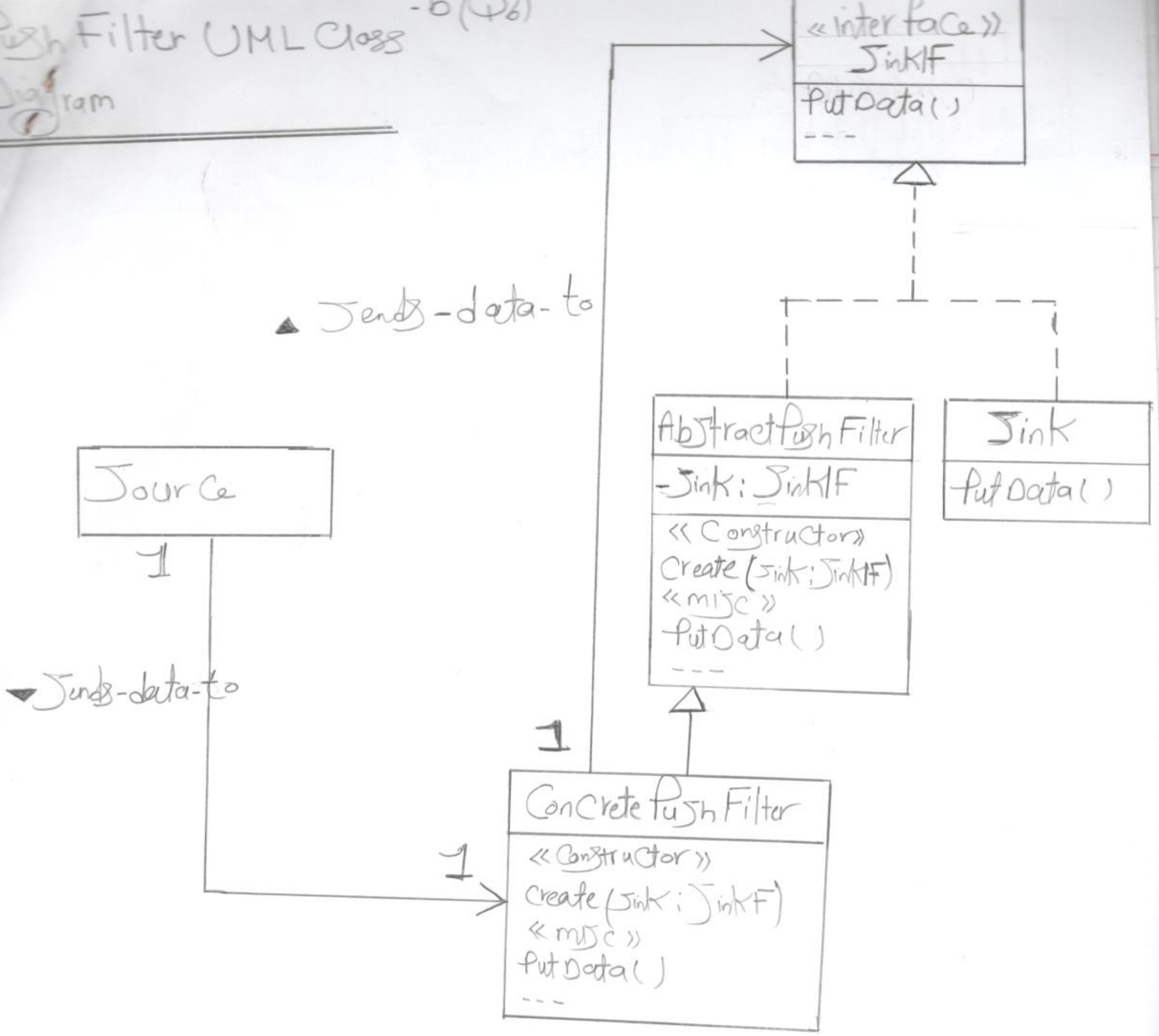


The design pattern used is ① Filter design pattern

② Iterator design pattern

Push Filter UML Class Diagram

-6(+6)


«interface»
SinkIF
PutData()

▲ Sends-data-to
Source
1
▼ Sends-data-to
AbstractPushFilter
-Sink: SinkIF
«Constructor»
Create(Sink: SinkIF)
«misc»
PutData()

Sink
PutData()
1
1
ConcretePushFilter
«Constructor»
Create(Sink: SinkF)
«misc»
PutData()
