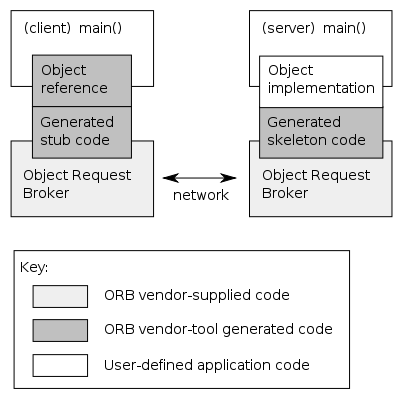
**Aim:** To develop a Medical Store using CORBA.

**Theory:**

* CORBA(**Common Object Request Broker Architecture**) enables communication between software written in different languages and running on different computers.
* CORBA normalizes the method-call semantics between application objects residing either in the same address-space (application) or in remote address-spaces (same host, or remote host on a network.
* CORBA uses an interface definition language (IDL) to specify the interfaces that objects present to the outer world. CORBA then specifies a *mapping* from IDL to a specific implementation language like C++ or Java
* The CORBA specification dictates there shall be an ORB through which an application would interact with other objects.
* This is how it is implemented in practice:

1. The application simply initializes the ORB, and accesses an internal *Object Adapter*, which maintains things like reference counting, object (and reference) instantiation policies, and object lifetime policies.
2. The Object Adapter is used to register instances of the *generated code classes*. Generated code classes are the result of compiling the user IDL code, which translates the high-level interface definition into an OS- and language-specific class base for use by the user application. This step is necessary in order to enforce CORBA semantics and provide a clean user process for interfacing with the CORBA infrastructure.



* In addition to providing users with a language and a platform-neutral remote procedure call (RPC) specification, CORBA defines commonly needed services such as transactions and security, events, time, and other domain-specific interface models.

**Steps to implement CORBA:**

**1. Create** **store.idl file**

module MedicalStore{

interface medintf{

string check\_medicine(in string med\_id);

string process\_medicine(in string med\_id, in long quantity);

long total\_price();

string pay\_bill();

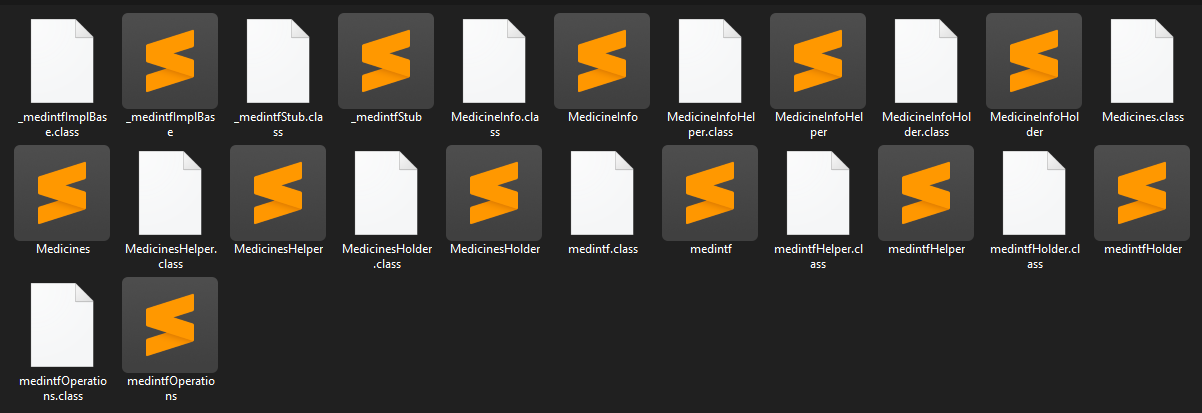
};

};

**2. Compile the idl using the command:**

idlj -fall store.idl

**3. The above command generates the following files:**

****

**4. Create Client.java file.**

import MedicalStore.\*;

import org.omg.CosNaming.\*;

import org.omg.CosNaming.NamingContextPackage.\*;

import org.omg.CORBA.\*;

import java.util.\*;

class Client{

    public static void main(String[] args){

        try{

            int i;

            String j;

            Scanner sc=new Scanner(System.in);

            String s,b,a;

            ORB orb=ORB.init(args,null);

            org.omg.CORBA.Object objRef =

            orb.resolve\_initial\_references("NameService");

            NamingContext ncRef = NamingContextHelper.narrow(objRef);

            NameComponent nc=new NameComponent( "Medicines" , "" );

            NameComponent path[] = {nc} ;

            medintf lbref=medintfHelper.narrow( ncRef.resolve(path) );

            do{

                int p=0;

                System.out.println( ++p +".Check medicines");

                if(lbref.total\_price() > 0){

                            System.out.println( ++p +".Proceed to Payment");

                }

                System.out.println(++p +".Exit");

                System.out.print("Enter choice: ");

                i=sc.nextInt();

                switch(i){

                    case 1:System.out.print("Enter medicine name: ");

                    j=sc.next();

                    s=lbref.check\_medicine(j);

                    System.out.println(s);

                    System.out.println("If you want to  confirm type 'yes' else 'no'");

                    b=sc.next();        a=b.toLowerCase();

                    if(a.equals("yes")){

                         System.out.print("Enter number of quantities: ");

                         int y=sc.nextInt();

                         s=lbref.process\_medicine(j,y);

                         if(s.equals("")){

                             s="Sorry medicines not in stock!!!";

                         }

                         System.out.println(s);

                    }

                    break;

                    case 2:int pay=lbref.total\_price();

                    System.out.println("Total amount to be paid: "+pay);

                    System.out.println("If you want to pay type 'yes' else 'no' ");

                        b=sc.next();        a=b.toLowerCase();

                    if(a.equals("yes")){

                         s=lbref.pay\_bill();

                         System.out.println(s);

                    }

                    break;

                    default:

                    if(lbref.total\_price() > 0){

                        System.out.println("Your cart amount: "+lbref.total\_price());

                     System.out.print("Do you want to pay? ");

                     b=sc.next();       a=b.toLowerCase();

                     if(a.equals("yes")){

                                 s=lbref.pay\_bill();

                                 System.out.println(s);

                     }

                    }

                    break;

                }

            }while(i!=3);

        }catch(Exception e){

            e.printStackTrace();

        }

    }

}

**5. Create Server.java file.**

import org.omg.CosNaming.\*;

import org.omg.CosNaming.NamingContextPackage.\*;

import org.omg.CORBA.\*;

import java.util.\*;

import MedicalStore.\*;

class Server extends \_medintfImplBase{

    static Map<String,Integer> medicine\_list;

    static int[] prices;

    static int total;

    static int store;

    public Server(){

        total=0;

        medicine\_list=new HashMap<String,Integer>();

        prices=new int[10];

        medicine\_list.put("Paracetamol",15);

        medicine\_list.put("Lyrica",20);

        medicine\_list.put("Ibuprofen",20);

        medicine\_list.put("Codiene",20);

        medicine\_list.put("Cymbalta",20);

        medicine\_list.put("Ativan",20);

        medicine\_list.put("Losartan",20);

        medicine\_list.put("Actidone",20);

        medicine\_list.put("Lexapro",20);

        medicine\_list.put("Lyrica",20);

        prices[0]=90;   prices[1]=150;  prices[2]=70;

        prices[3]=120;  prices[4]=24;   prices[5]=98;

        prices[6]=140;  prices[7]=274;  prices[8]=210;

        prices[9]=50;

    }

  // checking for availabilty for medicines

    public String check\_medicine(String med\_id){

        store=-1;

        int i=0;

        if(!medicine\_list.containsKey(med\_id)   )

            {

                        return "Medicine not present";

            }

      String s="";

        for(String ik: medicine\_list.keySet() ){

            if( ik.equals(med\_id) )

            {

                s="Medicine present with total available quantity: "+ medicine\_list.get(med\_id) +"\nPrice:"+prices[i];

                store=i;

                break;

            }

            i++;

        }

        return s;

    }

  // add to cart

    public String process\_medicine(String med\_id,int quantity){

        int q=  medicine\_list.get(med\_id);

        String s="";

        if(q>=quantity){

            total+= prices[store] \* quantity;

            medicine\_list.put( med\_id, medicine\_list.get(med\_id) - quantity );

            System.out.println("Medicine "+med\_id+" ordered with quantities "+quantity);

            s= "Medicine added to cart";

        }

      return s;

    }

  //return total prices

  public int total\_price(){

        return total;

    }

    public String pay\_bill(){

        System.out.println("Payment successful with amount "+total);

        total=0;

        return "Payment successful!!";

    }

    public static void main(String[] args){

        try{

        ORB orb = ORB.init(args, null);

        Server lbRef=new Server();

        orb.connect(lbRef);

        org.omg.CORBA.Object objRef = orb.resolve\_initial\_references("NameService");

        NamingContext ncRef = NamingContextHelper.narrow(objRef);

        NameComponent nc=new NameComponent( "Medicines" , "" );

        NameComponent path[] = {nc} ;

        ncRef.rebind(path,lbRef);

        System.out.println("Server started!!");

        Thread.currentThread().join();

        }catch(Exception e){

            System.err.println(e);

        }

    }

}

**6. Compile all files using the command:**

javac \*.java

**7. Start ordb from terminal using command:**

orbd -ORBInitialPort 1050&

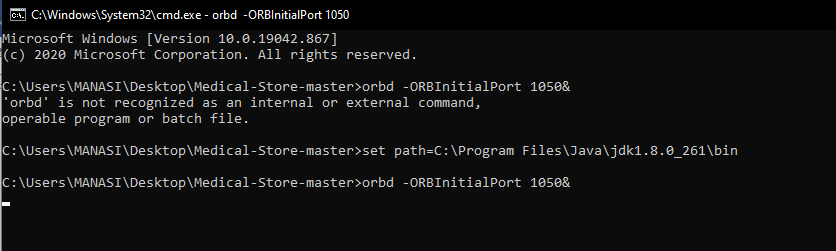
**8. Start Server using the command:**

java Server -ORBInitialPort 1050 -ORBInitialHost localhost&

**9. Start Client using the command:**

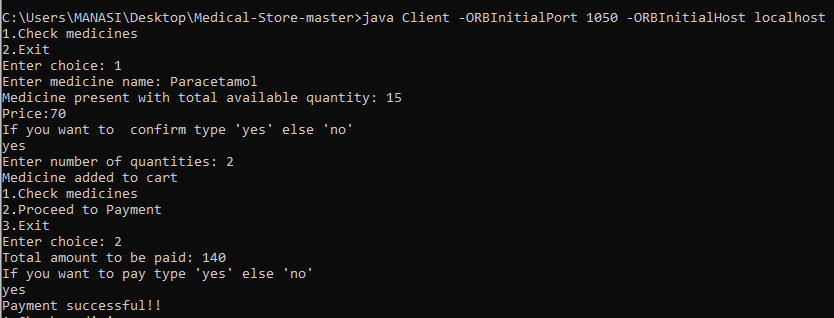
java Client -ORBInitialPort 1050 -ORBInitialHost localhost

**Output:**

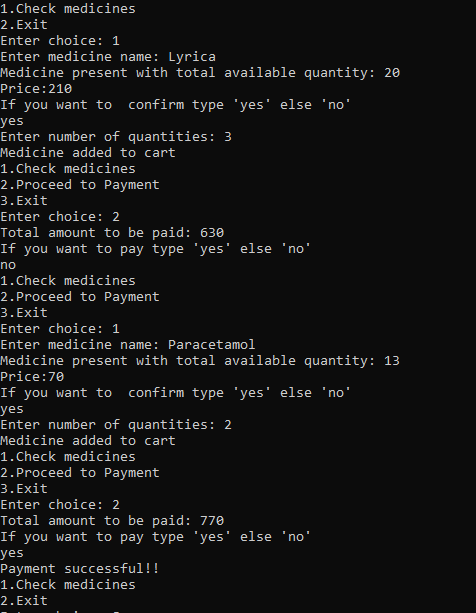
****

****

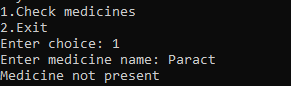
**Payment for single medicine**

****

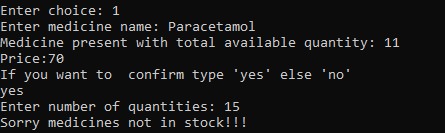
**Purchasing multiple medicines**

****

**Invalid Medicine**

****

**Medicine out of stock**

****