Q1) a) abstract class is a class [redacted]
that can't be intantiated
and meant to be super class
interface same as abstract class
can't be intantiated (meant to
be implemented - not extend -

abstract class	interface
can't be intantiated (no object)	can't be intantiated (no object)
can be a reference	can be a reference
other clases extend it	othe classes implement it (othe interfaces extend it

b) Encapsulation: hide the data belonging to an object
(privat, protected)

Abstraction: the main idea of it → know what the objet do not
how it does it, so it hide unncessery details

inheritance: the main idea of it is to reduce duplicate code
between classes (objects) the have mutual features
such as data, methods

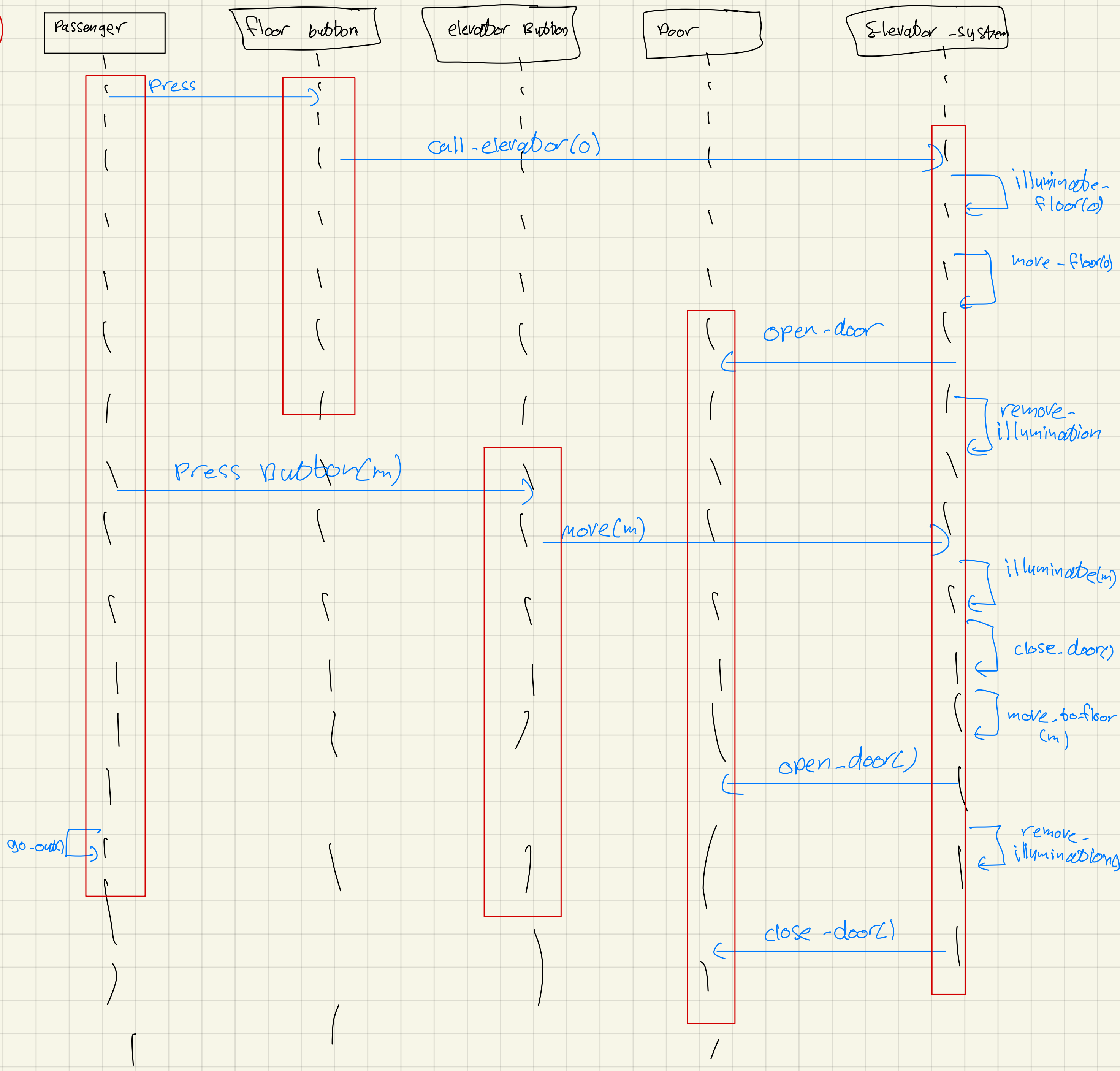
$$\begin{tikzcd}
 & \text{Animal} \arrow[dl, no head] \arrow[dr, no head] & \\
\text{fish} & & \text{reptile}
\end{tikzcd}$$

polymorphism: one interface multiple methods (implementation)
so when 2 classes inherite same method
they can over ride it (dynamic binding for non-static)

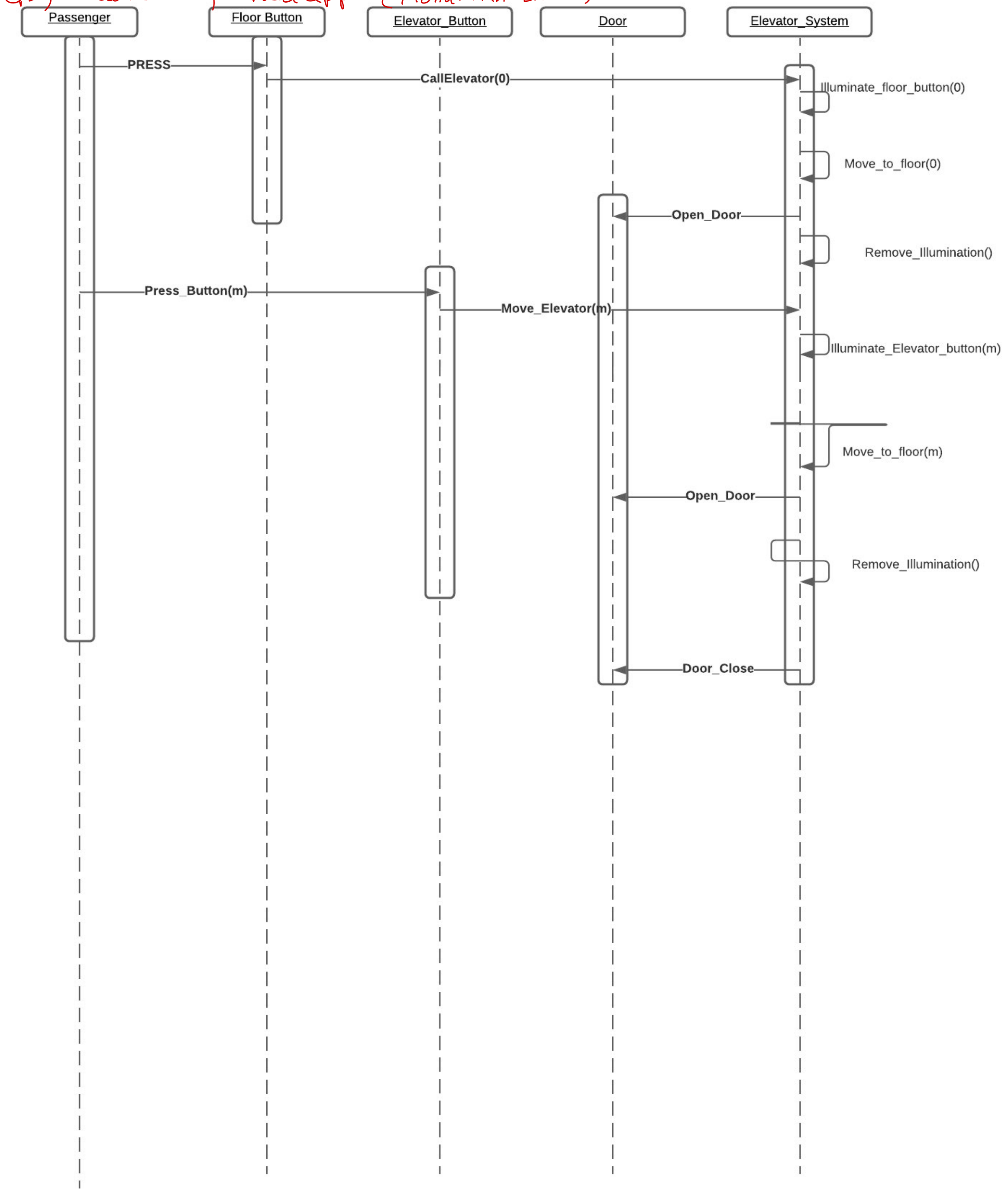
Q1) c)

```
public class exam{
    . -
    public void exam-time( int time)
    {

    }
overload    pubic void exam-time(int tim, int eac-q) (different
                                                      proto type (signture)
      {
       }
}

public class Midterm extends exam{
    @override
    public void exam-time(int time) (must be same
                                     name and signture}
override      {                      and the access
                                     modifier cant be
       }                             lower than super's
```

Q2)


Passenger
floor button
elevator Button
Door
Elevator -system
press
call-elevator(0)
illuminate-floor(0)
move-floor(0)
open-door
remove-illumination
Press Button(m)
move(m)
illuminate(m)
close_door()
move_to_floor (m)
open_door()
go-out()
remove-illuminations
close-door()

Q2) done using lucid.app (Mohammad Snab10)

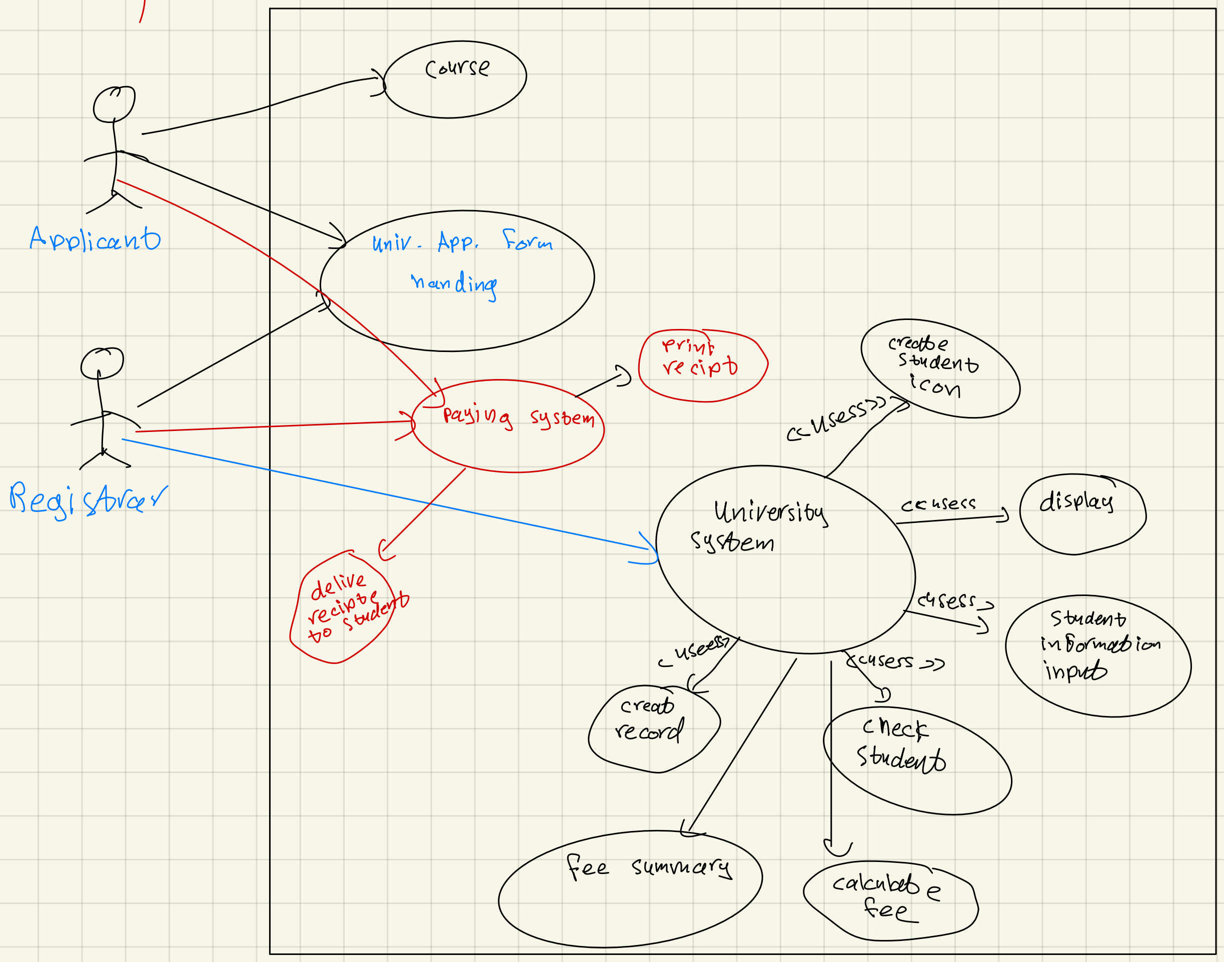


Q3)

```
public class Person
{
    private string name
}

public Staff extends Person
{
   private int ID
}

 public class Passenger extends Person
  {
      ...
      Array list < Reservation>  P_ reservations;

  }

  public class Driver extends Staff {
      ..
  }

   public class Hostess extends Staff {
      ..
   }


   public class Reservation {
        Seat    Reserved_seat;
   }

   public class seat {
      private int no;
      private int isReserved
      private int isTaken;
   }

    public class Buss {                    public class Buss Company {
      Driver   buss_driver ;               Arraylist< Buss> company_Busss ;
      Hostess  buss_hostess;
      Array list<Trip>  buss_trips;
    }

     public class trip {

        Schedule   trip_schedule;

     }
```

```
public class RegularTrip extends Trip{

      }

public class ExpressTrip extends Trip{

      }

public class Schedule{

   Arraylist<serviceStop> service_s ;

}

public class ServiceStop {

   Schedule name_Schedule;
}

public class BussStation extends ServiceStop{

}

public class RestArea extends ServiceStop{

}
```

Q4)


Applicant
Registrar
Course
Univ. App. Form handing
paying system
print recipt
delive recipte to student
University system
create student icon
<<Usess>>
<<usess
display
<usess>
student information input
<users>>
check student
calculate fee
fee summary
<useess>
creat record