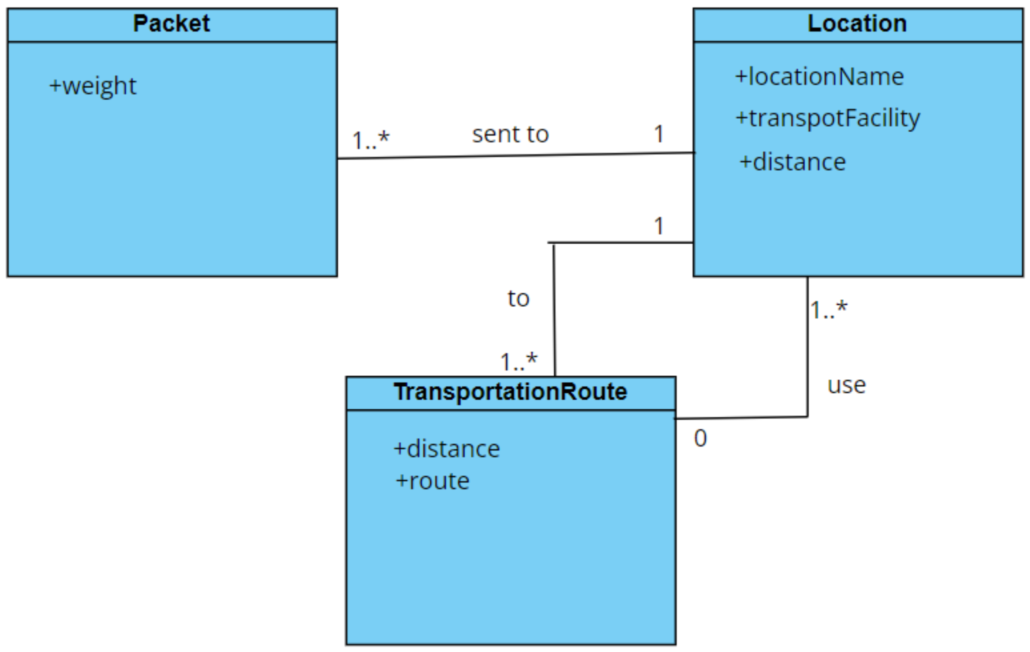
Problem 1 (20 points)

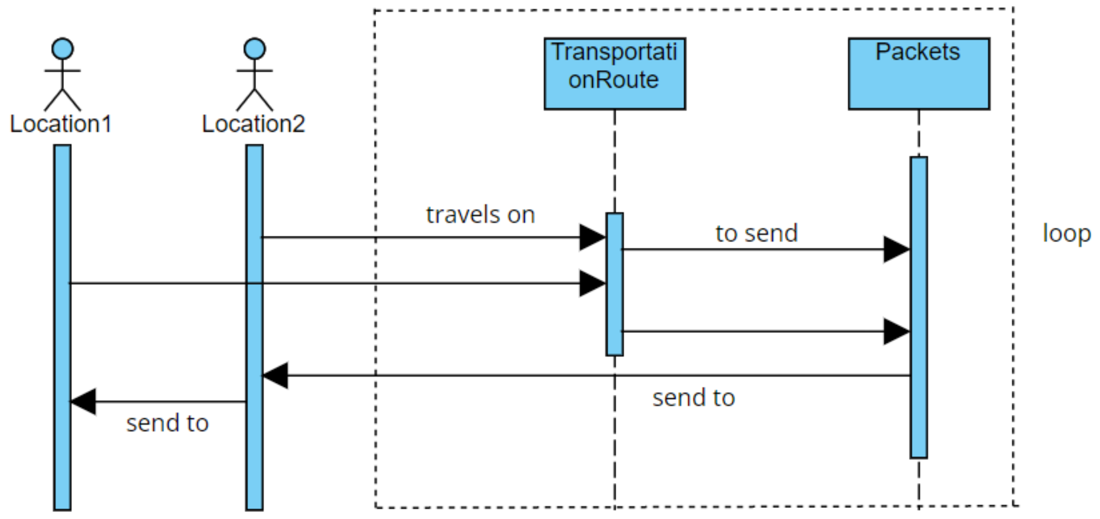
Assume the following requirements:

Packets are sent from one location to another. Packets have a certain weight. Locations are characterized by their transportation facilities, e.g. railway stations, airports and highway connections. Some locations are neighbored, i.e. there exists a direct transportation route between these locations. The transportation route between the locations has a certain length, i.e. the distance between the locations. Planes, trains, and trucks are used for transportation; each plane / train / truck may load a maximum packet weight. For each packet we want to know where it is, i.e. at which location or transport (plane, train, truck).

(i) Draw a class diagram for this problem; identify the semantic relationships and cardinalities. (10 points)

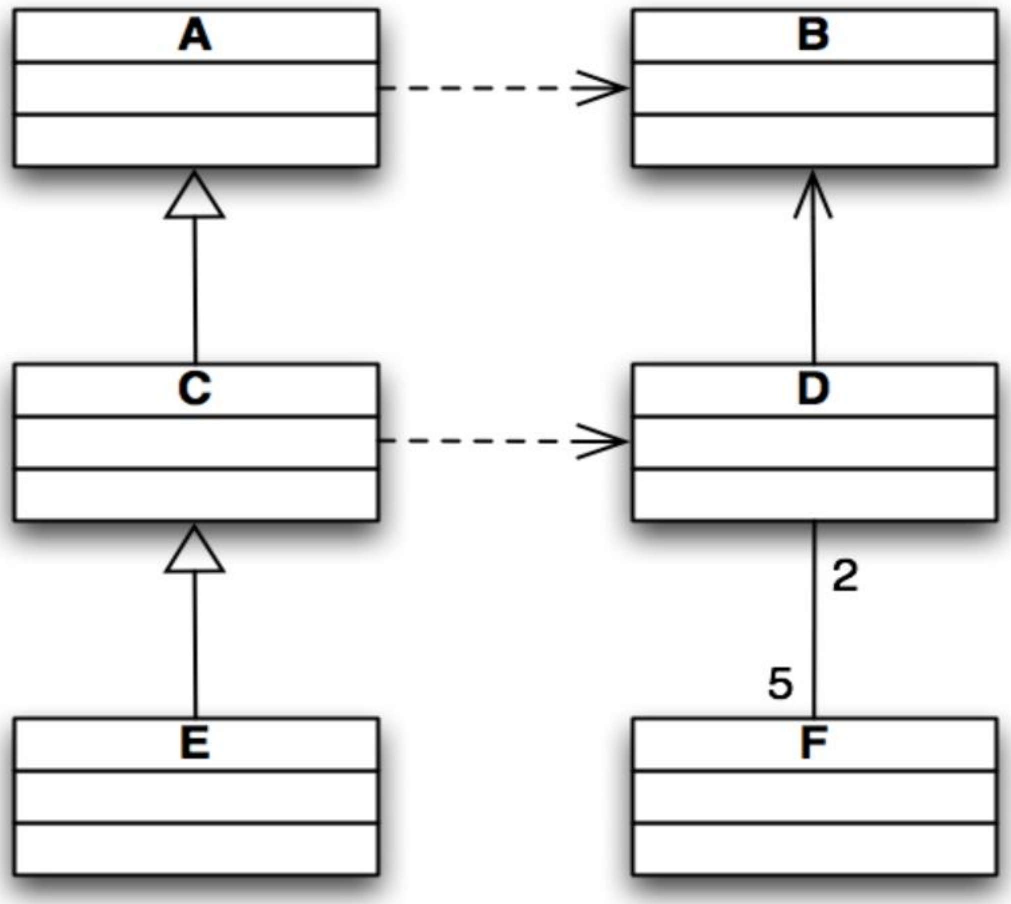


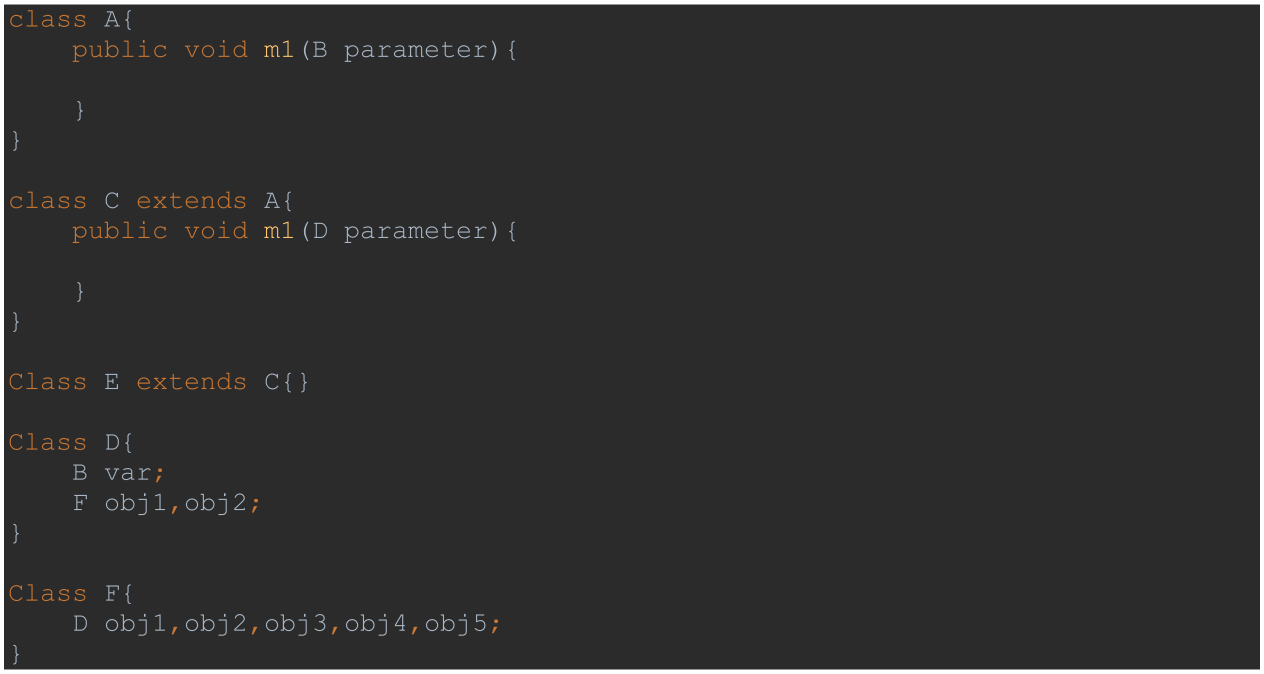
(ii) **Draw the sequence diagram corresponding to a packet being sent from one location to another. (10 points)**


Location1
Location2
TransportationRoute
Packets
travels on
to send
loop
send to
send to

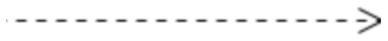
Problem 2 (10 points)

Given the following UML write the corresponding (skeleton) Java code.





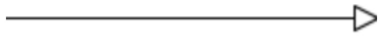
```
class A{
    public void m1(B parameter){

    }
}

class C extends A{
    public void m1(D parameter){

    }
}

Class E extends C{}

Class D{
    B var;
    F obj1,obj2;
}

Class F{
    D obj1,obj2,obj3,obj4,obj5;
}
```



Dependency

A is dependent on B

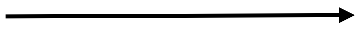
C is dependent on D



Inheritance

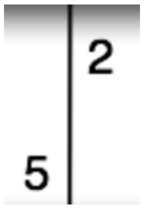
C inherits A

E inherits C



Association (two classes are linked to each other)

D is associated with B

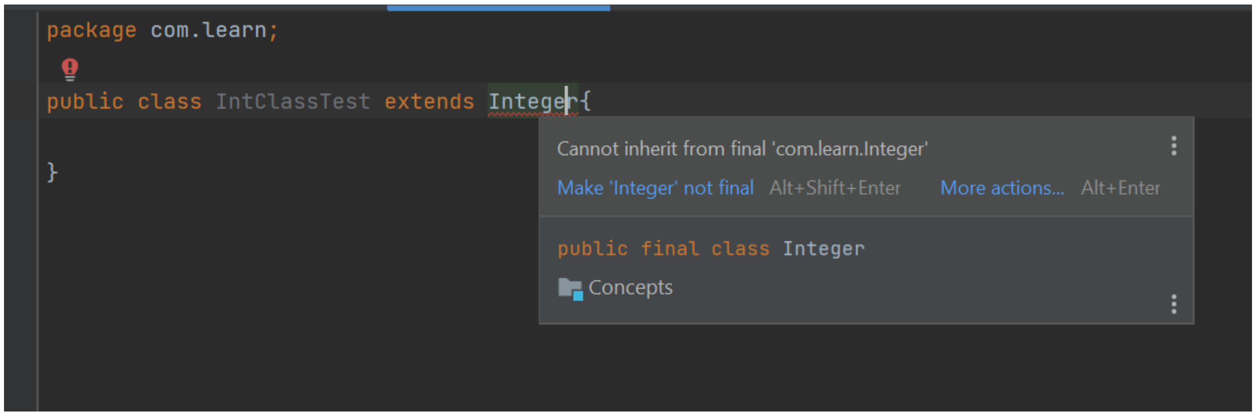


Multiplicity (how many objects of each class take part in the relationships

Problem 3 (20 points)

Integer is part of the Java API. Suppose you attempt to extend the Integer class and add a new method that returns the integer as a String that is written in hexadecimal.

(i) Explain why you're having trouble doing it. (5 points)



```
package com.learn;

public class IntClassTest extends Integer{

}
```

Cannot inherit from final 'com.learn.Integer'

Make 'Integer' not final Alt+Shift+Enter
More actions... Alt+Enter

```
public final class Integer
```

In the above screenshot I tried to inherit Integer class but got the error message saying cannot inherit from final

As Integer is a final class and if we declare any class as final, we are restricting the access, so other classes cannot be inherited or extend final classes.

(ii) Alright, so the people who wrote the code had their reasons to not want you to extend the class. Give an example of how things could go very wrong if they didn't do it this way. (5 points)

For example, if we take BankTransaction as an entity and it has withdraw and deposit methods.

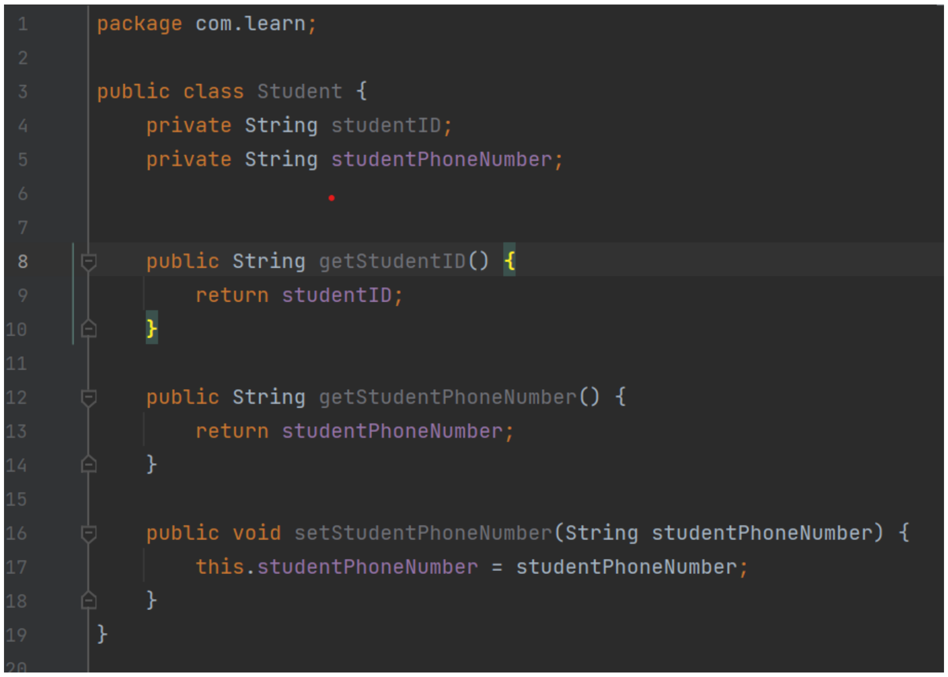
Withdraw method job is to take the money out of the account and the deposit method job is to add money to the bank.
Here BankTransaction is a superclass. Now if we want to change this superclass behavior for one person and write a new class SpecialBankTransaction which extends BankAccount class and change all method implementations
In SpecialBankTransaction class will change deposit method implementation to withdraw and withdraw implementation to deposit.
With these changes instead of one person's bank behavior change whole superclass behavior got changed and it will create a lot of chaos.
So, the change to a superclass can be avoided if the BankTransaction class was made final**.**

(iii)Recommend a solution to the problem that doesn't involve subclassing. (10 points)

a. No Setter methods

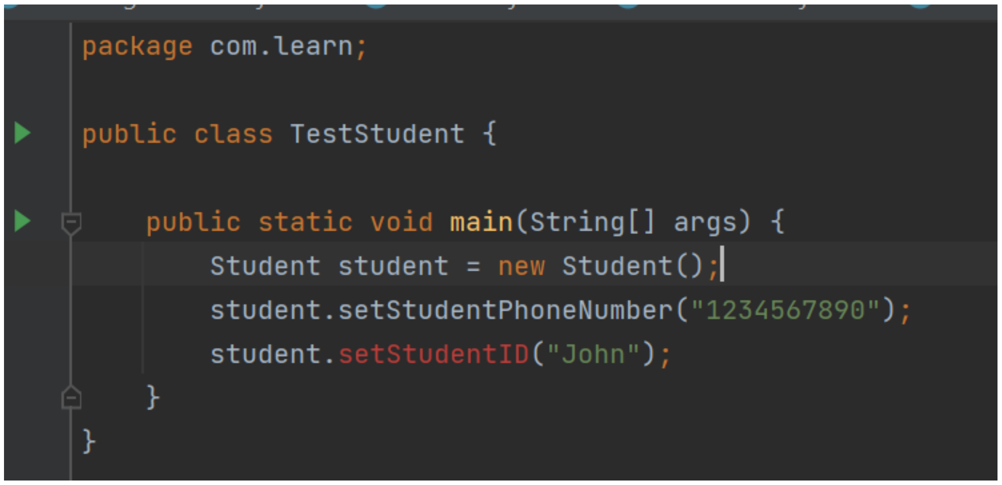
After defining the variables in the class, do not write setter methods, if there are no setter methods, the variables in the class cannot the modified.

Let's say I have a student class like below



```java
package com.learn;

public class Student {
    private String studentID;
    private String studentPhoneNumber;


    public String getStudentID() {
        return studentID;
    }

    public String getStudentPhoneNumber() {
        return studentPhoneNumber;
    }

    public void setStudentPhoneNumber(String studentPhoneNumber) {
        this.studentPhoneNumber = studentPhoneNumber;
    }
}
```

Now in my test class I am unable to modify the value of the variable



```java
package com.learn;

public class TestStudent {

    public static void main(String[] args) {
        Student student = new Student();
        student.setStudentPhoneNumber("1234567890");
        student.setStudentID("John");
    }
}
```

b. Use final and private

If you define the fields as final and private the value cannot be changed. Fields declared final cannot be accessed other than the defined classes. Final implies that the value cannot be changed thus locking the change.