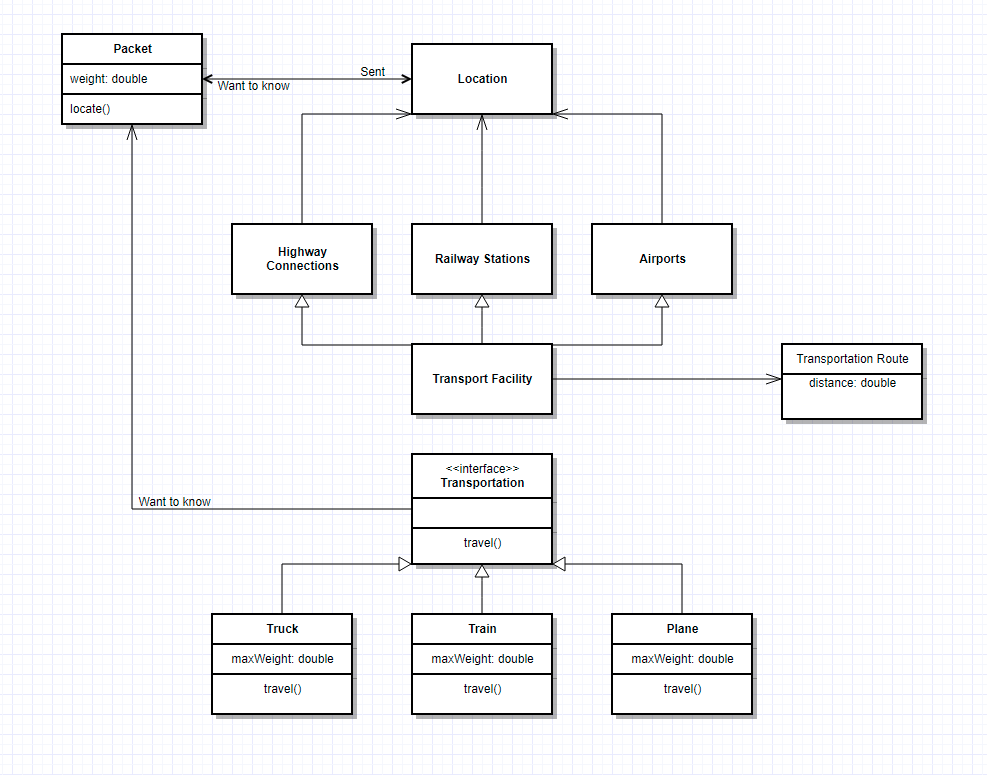
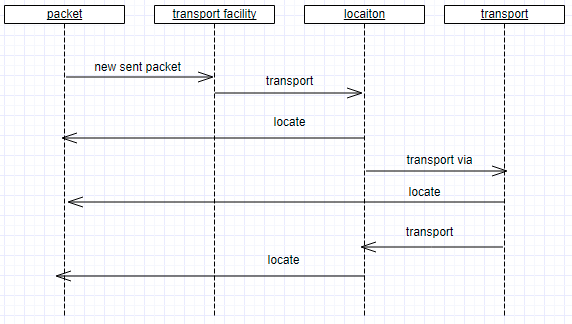
Problem 1:





Problem 2:

public class A

{

public void functionA(B b)

{

}

}

public class B

{

public B()

{

}

}

public class C extends A

{

public void functionC(D d)

{

}

}

public class D

{

private F[] classF;

ArrayList<F> bList = new ArrayList<F>();

public D()

{

classF = new F[2];

classF[0]= new F();

classF[1]= new F();

bList.add(new B();

}

}

public class E extends C

{

}

public class F

{

private D[] classD;

ArrayList<D> dList = new ArrayList<D>();

public F()

{

classD = new D[5];

classD[0]= new B();

classD[1]= new B();

classD[2]= new B();

classD[3]= new B();

classD[4]= new B();

}

}

Problem 3

* The reason you could be having trouble is because Hexadecimal in in base16 there are inbuilt memory allocation limits in java.
* The inbuilt memory allocation limits could be reached causing the program to crash.
* You could use a composition which is a “has-a” kind of relationship rather than the “is-a” relationship that is inheritance. So instead of extending the Integer class you would have an instance of it.