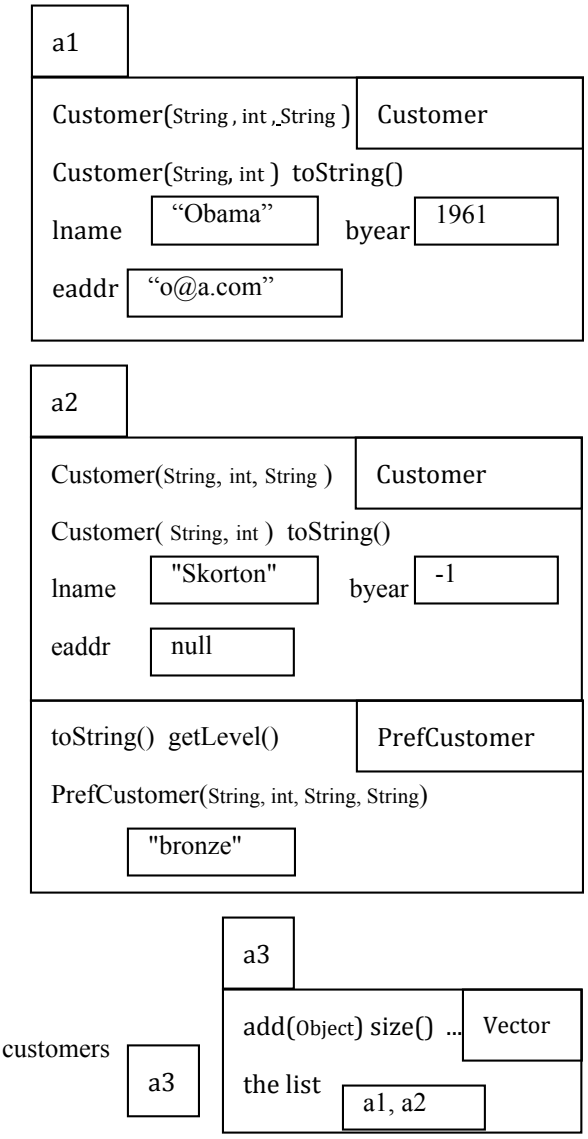
1.



2. Methods in Customer

```
/** Constructor: a customer with last name lname,
     birth year y (-1 if unknown), and email address a
    (null if unknown). */
public Customer(String lname, int y, String a) {
      this.lname= lname;
      byear= y;
      eaddr= a;
      customers.add(this);
}

/** Constructor: customer with last name n, birth year y
     (-1 for unknown), and unknown email address */
public Customer(String n, int y) {
     this(n, y, null);
}

/** = String repr. of customer (the last name) */
public String toString() {
      return lname;
}
```

Methods in PrefCustomer

```
/** Constructor: instance with last name lname,
     birth year y (use -1 if unknown),
     email address addr (use null if unknown), and
     level lev, -- one of "bronze", "silver", "gold". */
public PrefCustomer(String lname, int y,
                    String addr, String lev) {
   super(lname, y, addr);
   level= lev;
}

/** = the level of this preferred customer –one of
       "bronze", "silver" and "gold" */
public String getLevel() {
      return level;
}

/** = String representation … in the form
         <last name>, <level>
     e.g. "gries, silver",  "lee, gold"  */
public String toString() {
   return super.toString() + ", " + level;
}
```

3a.

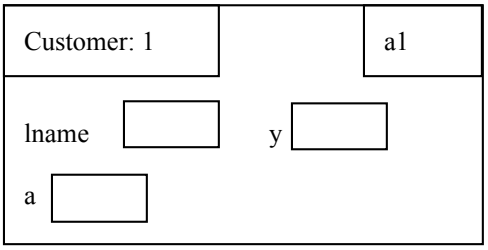
```
/** See prelim for the spec. */
public Customer(String addr) {
   customers.add(this);
   eaddr= addr;
   int k= addr.indexOf("@");

   if (!Character.isDigit(addr.charAt(k-1))) {
      lname= addr.substring(0,k);
      byear= -1;
      return;
   }

   lname= addr.substring(0,k-4);
   byear= Integer.parseInt(addr.substring(k-4,k));
}
```

3b. The two wrapper classes mentioned in the table are `Integer` and `Character`. One purpose of a wrapper class is to provide objects that wrap a value of the corresponding primitive type so they can be viewed as objects. Another purpose (you had to mention only one) is to house useful static methods and fields that deal with items of the corresponding primitive type.

4a. Make a field static if there should be only one copy of it (instead of it appearing in each object). This would be the case for a variable that is going to contain information about all objects, for example.

4b. `Customer.customers.size()`

4c.



4d. // Swap C1 and C2
```
Customer t= c1;
c1= c2;
c2= t;
```

4e. The instance variable is declared in a class definition. The static variable is declared in a class definition. The parameter is declared within the parentheses of the header of a method. The local variable is declared within the body of a method.