Lösningsförslag till övning 2.

Uppgift 1

```
public static String[] bornThisYear(Person[] people, int year) {
    if (people.length == 0) {
        return new String[0];
    }
    int count = 0;
    for (int i = 0; i < people.length; i = i + 1) {
        if (people[i].getBirthYear() ==  year)
            count = count = count + 1;
    }
    String[] result = new String[count];
    int j = 0;
    for (int i = 0; i < people.length; i = i + 1) {
        if (people[i].getBirthYear() ==  year) {
            result[j] = people[i].getName();
            j = j + 1;
        }
    }
    return result;
}

public static ArrayList<String> bornThisYear(Person[] people, int year) {
    ArrayList<String> result = new ArrayList<String>();
    for (int i = 0; i < people.length; i = i + 1) {
        if (people[i].getBirthYear() ==  year)
            result.add(people[i].getName());
    }
    return result;
}

public static ArrayList<String> bornThisYear(ArrayList<Person> people, int year) {
    ArrayList<String> result = new ArrayList<String>();
    for (Person p : people) {
        if (p.getBirthYear() ==  year)
            result.add(p.getName());
    }
    return result;
}

public static ArrayList<String> bornThisYear(ArrayList<Person> people, int year) {
    ArrayList<String> result = new ArrayList<String>();
    Iterator<Person> p =  people.iterator();
    while (p.hasNext()) {
        Person obj = p.next();
        if (obj.getBirthYear() ==  year)
            result.add(obj.getName());
    }
    return result;
}
```

```
  public static void removeNames(ArrayList<Person> people,  ArrayList<String>  names) {
        Iterator<String> n =  names.iterator();
        while (n.hasNext()) {
              String name = n.next();
              Iterator<Person> p =  people.iterator();
              while (p.hasNext()) {
                    if (p.next().getName() ==  name)
                    p.remove();
              }
        }
}
```

Uppgift 2

a)

```
public static void uniqueWords(Array_List<String> words) {
    ArrayList<String> unique = new ArrayList<String>();
    for (String w : words) {
        if (!unique.contains(w))
            unique.add(w);
    }
    for (String w : unique)
        System.out.println(w);
}// uniqueWords
```

b)

```
public static void nrOfOccurrence(ArrayList<String> words) {
    TreeMap<String, Integer> occurence = new TreeMap<String, Integer>();
    for (String w : words) {
        if (!occurence.containsKey(w))
            occurence.put(w,1);
        else
            occurence.put(w,occurence.get(w)+1);
    }
    Iterator i = occurence.entrySet().iterator();
    while (i.hasNext()) {
        Map.Entry entry = (Map.Entry) i.next();
            System.out.println(entry.getKey() + ": " + entry.getValue());
    }
}// nrOfOccurrence
```

c)

```
public static void positiosOfOccurrence(ArrayList<String> words) {
    TreeMap<String, ArrayList<Integer>> occurence =
                                        new TreeMap<String,ArrayList<Integer>>();
    int pos = 0;
    for (String w : words) {
        pos = pos +1;
        if (!occurence.containsKey(w)) {
            ArrayList<Integer> al = new ArrayList<Integer>();
            al.add(pos);
            occurence.put(w,al);
        }
        else {
            ArrayList<Integer> al = occurence.get(w);
            al.add(pos);
        }
    }
    Iterator i = occurence.entrySet().iterator();
    while (i.hasNext()) {
        Map.Entry entry = (Map.Entry) i.next();
        System.out.print(entry.getKey() + ": ");
        ArrayList al = (ArrayList) entry.getValue();
        Iterator ai = al.iterator();
        while(ai.hasNext()) {
            System.out.print(ai.next());
            if (ai.hasNext())
                System.out.print(", ");
        }
        System.out.println();
    }
}// positiosOfOccurrence
```