```
            int lastLanes = count - lastHeat;
            numHeats = count / numLanes;
            if(lastLanes > 0)
               numHeats++;
            int heats = numHeats;

            //place heat and lane in each swimmer's object
            int j = 0;

            for(int i = 0; i < lastLanes; i++) {
               Swimmer sw = swmrs[i];

               sw.setLane(lanes[j++]);
               sw.setHeat(heats);
               if(j >= numLanes) {
                  heats--;
                  j=0;
               }
            }

            //Add in last partial heat
            if(j < numLanes)
               heats--;
            j = 0;
            for(int i = lastLanes-1; i<count; i++) {

               Swimmer sw = swmrs[i];
               sw.setLane(lanes[j++]);
            }
            //copy from array back into Vector

            Swimmers = new Vector();
            for(int i = 0; i < count; i++)
               Swimmers.addElement(swmrs[i]);
}
```

This makes the entire array of seeded Swimmers available when we call the *getSwimmers* method.

Circle Seeding

The CircleSeeding class is derived from StraightSeeding, so it copies in the same data.

```
public class CircleSeeding extends StraightSeeding {

    public CircleSeeding(Vector sw, int lanes) {
        super(sw, lanes);  //straight seed first
        seed();
    }
//--------------
    protected void seed() {
        int circle;

        super.seed();      //do straight seed as default
```