Name: Mohan (Dhar   Poly-Id: 0520191

Programming – Blue Book

- Place the answer to the following question in your Blue Book.
- **Comments** are **not** required in the blue book!
  However, if you think they will help us understand your code,
  feel free to add them.
- **#includes** are not required in the blue book.
- Read the question *carefully!*

12. [50 pts] One of the most important jobs in our country is the baker. He makes all those treats that we crave and that provide us with the energy to study for (and write) exams! **You will implement a class** to represent this important national resource. (Note, you are only implementing the Baker class.)

What do bakers do?

- Make treats! The baker has to create treats on demand, e.g.
  `theBaker.bakes(“Twinkie”);`
  - Each treat will be created **on the heap** so it can have a long shelf-life.
  - The Treat class has
    - a constructor that takes a string which is that name of the Treat
    - an output operator that displays the name.
    - Anything else Treats have is a trade secret. (Ok, they do support copy control.)
- Deliver the treats to a company, who will in turn package them and sell them (to us!).
  - This requires that he pass the collection off to the bakery company. After handing the collection over, he is back to having nothing. All those treats, and in fact the container itself that held them now belongs to the company. E.g.
    `aCompany.receives(theBaker.delivers());`

Just to keep life entertaining, we will want to support copy control for our baker.

- Deep copy, of course.
- To keep this exam to a reasonable length, you only have to implement the assignment operator. For the other copy control functions, just provide their prototypes in the class and assume someone else will write them for you.

And naturally you should provide a reasonable output operator displaying him and his products. Se the sample test code and output on the next page.

And finally, let's have an equality operator. We will consider two bakers to be “equal” if they currently have the same number of treats.

So, what do *you* have to implement? **Just the Baker class.**

- You don't have to worry about the bakery company or even defining the Treat class..
  (Well, worry all you like, but you are not writing those classes.)
- You can also assume that Treat provides any needed operators or constructors.
- The baker has:
  - a constructor ✓
  - bakes method ✓
  - delivers method ✓
  - output operator ✓
  - equals operator ✓
  - copy control ✓