PROGRAMMING PROJECT #1

CIS 2353 - PROF. JOHN P. BAUGH – OAKLAND COMMUNITY COLLEGE – OR

Points: _____ / 100

OBJECTIVES

- To apply knowledge of the `Comparable` interface to a problem
- To understand and override the `toString` and `equals` methods

INSTRUCTIONS

For this assignment, you will create a Pizza class.

To maintain consistency for my grading, please place your Pizza and PizzaDemo classes (.java files) in a package called proj1

That means don't call your package p1, Proj1, MyProject, Prog1, Snuggles, Dumptruck, buffalo, or myPizzaisAwesome. Call the package **proj1**.

The class will contain the following fields:

- `crust`
  - The type of crust for the pizza
  - Valid values are based on an **enumerated type** (NOT A String), `CrustType`:
    - `Plain`
    - `Butter`
    - `Garlic`
    - `GarlicButter`
    - `Cheese`
- `toppings`
  - An ArrayList of strings, representing each of the `toppings`
  - The ArrayList may be empty, in which case, it's just a sauce pizza (no cheese, just sauce on bread)
  - The available toppings that may be included are cheese, onion, green pepper, ham, pineapple, pepperoni, ground beef, Italian sausage, and anchovies
- `size`
  - The size of the pizza
  - Valid values are based on an **enumerated type** (not a String, integer, or anything else), `SizeType`:
    - `Small`
    - `Medium`

        - `Large`
        - `XLarge`
        - `XXLarge`
        - `Party`

The class will contain the following methods:

- `Pizza( )`
    - Constructor – initializes the fields to their default values
        - `crust` set to `Plain`
        - `toppings` is empty
        - `size` set to `Small`
- `Pizza(crust, toppings, size)`
    - Constructor – initialize the fields to the values passed in by the client
- Getter and setter methods for each of the field names
- `addTopping(String topping)`
    - call this on a Pizza object to have an individual topping added
- `toString( )`
    - overridden from the Object class
    - This method should return a string containing a reference to the value in the following format:
      `This pizza has a` *`crust`* `crust and the following toppings:`
      `topping1`
      `topping2`
      `etc…`
    - In the case of no toppings it should say **none** after the "following toppings:" part of the output.
    - An actual example might be:
      `This pizza has a Butter crust and the following toppings:`
      `cheese`
      `pepperoni`
      `ham`
      `onion`

- `equals( )`
    - overridden from the Object class
    - A pizza is **equal** to another pizza if they have the same crust, *and* the same ***number*** of toppings
- `compareTo(Pizza otherPizza)`
    - implemented from the interface `Comparable`
    - Specifically, `Comparable<Pizza>` should be used for the interface
    - The method returns the following:
        - -1 if the current pizza, Pizza A is "less than" the other pizza, Pizza B
            - Go by the **number of toppings** first (if Pizza A has fewer toppings than Pizza B, then return -1)
            - If they have the same number of toppings, then check the crust:
              `Cheese > GarlicButter > Garlic > Butter > Plain`
        - 0 if the Pizzas have the same number of toppings and the same crust (same as if equals() returns true)
            - You can use this to your advantage 😉

        - 1 if the current pizza is “greater than” the other pizza
            - Look at the instructions under the -1 above
            - This time if Pizza A (current) has more toppings; *or* if it has the same number of toppings but a superior crust
- As a reminder, when you create your project, make sure the package the Pizza and PizzaDemo live in is called **proj1** and not anything else.

Your PizzaDemo class contains main, and simply creates a few Pizza objects and tests them against one another (e.g., equals, compareTo)

DELIVERABLES

- Turn in a **zipped up *folder***, containing **just the .java files**. **I don’t need the .class files**.
    - I need only the .java source files, which are found (in an Apache NetBeans folder structure) under the src folder.
- Also, **include screenshots of your program working,** placed inside the zip file that you turn in**.**

Also, as a another reminder: make sure your package is called **proj1**.