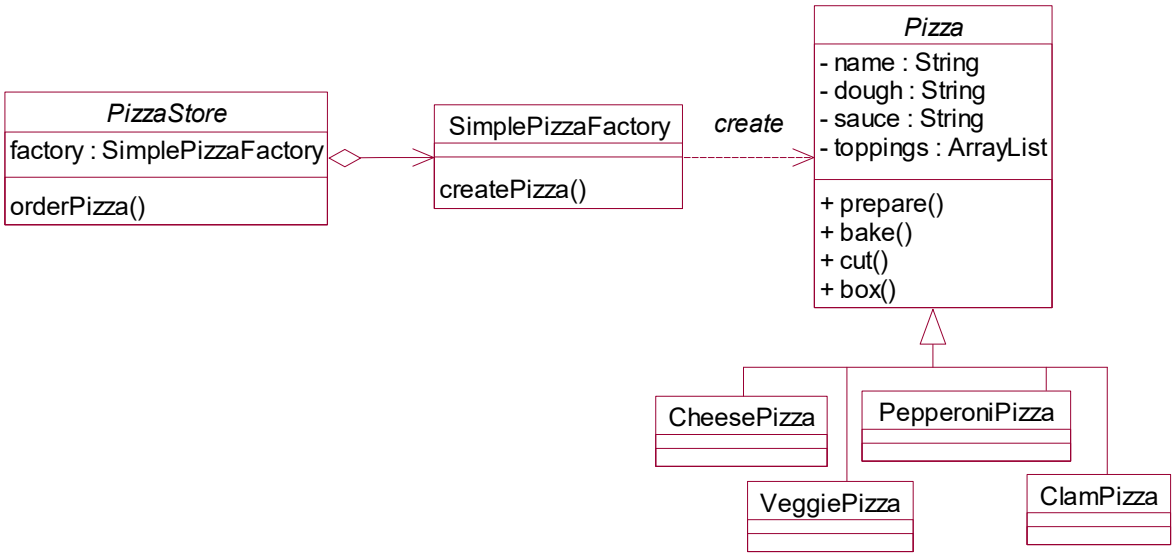
FACTORY PATTERN LAB

I> SIMPLE PIZZA FACTORY

1. Mở Eclipse, tạo một Java Project với tên **PizzaSF1**
2. Tạo các cấu trúc kế thừa lớp **Pizza** như lược đồ lớp sau:



Pizza Class:

```
public abstract public class Pizza {
   protected String name;
   protected String dough;
   protected String sauce;
   protected ArrayList<String> toppings = new ArrayList<String>();
   public String getName() {
      return name;
   }
   public void prepare() {
      System.out.println("Preparing " + name);
   }
   public void bake() {
      System.out.println("Baking " + name);
   }
   public void cut() {
      System.out.println("Cutting " + name);
   }
   public void box() {
      System.out.println("Boxing " + name);
   }
   public String toString() {
      // code to display pizza name and ingredients
      StringBuffer display = new StringBuffer();
      display.append("---- " + name + " ----\n");
      display.append(dough + "\n");
      display.append(sauce + "\n");
      for (int i = 0; i < toppings.size(); i++) {
         display.append(toppings.get(i) + "\n");
```

```
        }
        return display.toString();
    }
}
```

CheesePizza Class:

```
public class CheesePizza extends Pizza {
    public CheesePizza() {
        name = "Cheese Pizza";
        dough = "Regular Crust";
        sauce = "Marinara Pizza Sauce";
        toppings.add("Fresh Mozzarella");
        toppings.add("Parmesan");
    }
}
```

ClamPizza Class:

```
public class ClamPizza extends Pizza {
    public ClamPizza() {
        name = "Clam Pizza";
        dough = "Thin crust";
        sauce = "White garlic sauce";
        toppings.add("Clams");
        toppings.add("Grated parmesan cheese");
    }
}
```

VeggiePizza Class:

```
public class VeggiePizza extends Pizza {
    public VeggiePizza() {
        name = "Veggie Pizza";
        dough = "Crust";
        sauce = "Marinara sauce";
        toppings.add("Shredded mozzarella");
        toppings.add("Grated parmesan");
        toppings.add("Diced onion");
        toppings.add("Sliced mushrooms");
        toppings.add("Sliced red pepper");
        toppings.add("Sliced black olives");
    }
}
```

PeperoniPizza Class:

```
public class PepperoniPizza extends Pizza {
    public PepperoniPizza() {
        name = "Pepperoni Pizza";
        dough = "Crust";
        sauce = "Marinara sauce";
        toppings.add("Sliced Pepperoni");
        toppings.add("Sliced Onion");
        toppings.add("Grated parmesan cheese");
    }
}
```

3. Tạo lớp **SimpleFactory** để tạo các đối tượng Pizza

```
public class SimplePizzaFactory {

   public Pizza createPizza(String type) {
      // TO DO
   }
}
```

4. Tạo lớp **PizzaStore** sử dụng **SimplePizzaFactory** để tạo các loại **Pizza**

```
public class PizzaStore {
   private SimplePizzaFactory factory;

   public PizzaStore(SimplePizzaFactory factory) {
      this.factory = factory;
   }

   public Pizza orderPizza(String type) {
      Pizza pizza;
      // TO DO

      pizza.prepare();
      pizza.bake();
      pizza.cut();
      pizza.box();
      return pizza;
   }
}
```

5. Viết lớp test và thực hiện:

```
public class PizzaTestDrive {
   public static void main(String[] args) {
      SimplePizzaFactory factory = new SimplePizzaFactory();
      PizzaStore store = new PizzaStore(factory);

      Pizza pizza = store.orderPizza("cheese");
      System.out.println("We ordered a " + pizza.getName() + "\n");

      pizza = store.orderPizza("veggie");
      System.out.println("We ordered a " + pizza.getName() + "\n");
   }
}
```

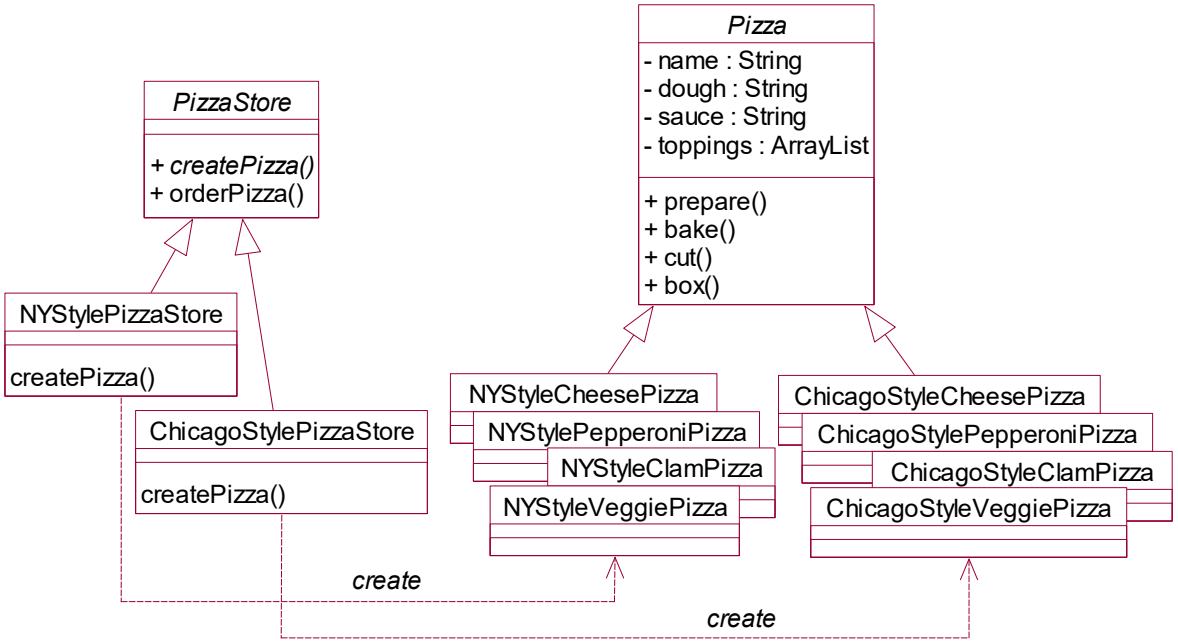
II) FACTORY METHOD

Bài toán:

- Giả sử công ty muốn mở nhiều cửa hàng bán Pizza ở nhiều thành phố khác nhau như Chicago và Newyork, …
- Từng cửa hàng bán các bánh Pizza được làm theo kiểu khác nhau phụ thuộc vào nơi cửa hàng đặt và hương vị địa phương khác biệt của từng vùng.

Dùng factory method để tạo các loại Pizza theo hương vị khác nhau của từng vùng

1. Mở Eclipse, tạo một Java Project với tên **PizzaFM**
2. Tạo các cấu trúc kế thừa lớp **Pizza** như lược đồ lớp sau:



Pizza class giống như trên

NYStyleCheesePizza

```
public class NYStyleCheesePizza extends Pizza {
   public NYStyleCheesePizza() {
      name = "NY Style Sauce and Cheese Pizza";
      dough = "Thin Crust Dough";
      sauce = "Marinara Sauce";
      toppings.add("Grated Reggiano Cheese");
   }
}
```

NYStyleClamPizza

```
public class NYStyleClamPizza extends Pizza {
   public NYStyleClamPizza() {
      name = "NY Style Clam Pizza";
      dough = "Thin Crust Dough";
      sauce = "Marinara Sauce";
      toppings.add("Grated Reggiano Cheese");
      toppings.add("Fresh Clams from Long Island Sound");
   }
}
```

NYStyleVeggiePizza

```
public class NYStyleVeggiePizza extends Pizza {
   public NYStyleVeggiePizza() {
      name = "NY Style Veggie Pizza";
      dough = "Thin Crust Dough";
      sauce = "Marinara Sauce";
      toppings.add("Grated Reggiano Cheese");
      toppings.add("Garlic");
      toppings.add("Onion");
      toppings.add("Mushrooms");
      toppings.add("Red Pepper");
   }
}
```

NYStylePeperoniPizza

```
public class NYStylePepperoniPizza extends Pizza {
   public NYStylePepperoniPizza() {
      name = "NY Style Pepperoni Pizza";
      dough = "Thin Crust Dough";
      sauce = "Marinara Sauce";
      toppings.add("Grated Reggiano Cheese");
      toppings.add("Sliced Pepperoni");
      toppings.add("Garlic");
      toppings.add("Onion");
      toppings.add("Mushrooms");
      toppings.add("Red Pepper");
   }
}
```

ChicagoStyleCheesePizza

```
public class ChicagoStyleCheesePizza extends Pizza {
   public ChicagoStyleCheesePizza() {
      name = "Chicago Style Deep Dish Cheese Pizza";
      dough = "Extra Thick Crust Dough";
      sauce = "Plum Tomato Sauce";
      toppings.add("Shredded Mozzarella Cheese");
   }
   public void cut() {
      System.out.println("Cutting the pizza into square slices");
   }
}
```

ChicagoStyleClamPizza

```
public class ChicagoStyleClamPizza extends Pizza {
   public ChicagoStyleClamPizza() {
      name = "Chicago Style Clam Pizza";
      dough = "Extra Thick Crust Dough";
      sauce = "Plum Tomato Sauce";
      toppings.add("Shredded Mozzarella Cheese");
      toppings.add("Frozen Clams from Chesapeake Bay");
   }
   public void cut() {
      System.out.println("Cutting the pizza into square slices");
   }
}
```

ChicagoStyleVeggiePizza

```
public class ChicagoStyleVeggiePizza extends Pizza {
   public ChicagoStyleVeggiePizza() {
      name = "Chicago Deep Dish Veggie Pizza";
      dough = "Extra Thick Crust Dough";
      sauce = "Plum Tomato Sauce";
      toppings.add("Shredded Mozzarella Cheese");
      toppings.add("Black Olives");
      toppings.add("Spinach");
      toppings.add("Eggplant");
   }

   public void cut() {
      System.out.println("Cutting the pizza into square slices");
   }
}
```

ChicagoStylePeperoniPizza

```
public class ChicagoStylePepperoniPizza extends Pizza {
   public ChicagoStylePepperoniPizza() {
      name = "Chicago Style Pepperoni Pizza";
      dough = "Extra Thick Crust Dough";
      sauce = "Plum Tomato Sauce";
      toppings.add("Shredded Mozzarella Cheese");
      toppings.add("Black Olives");
      toppings.add("Spinach");
      toppings.add("Eggplant");
      toppings.add("Sliced Pepperoni");
   }

   public void cut() {
      System.out.println("Cutting the pizza into square slices");
   }
}
```

PizzaStore class dùng factory method `createPizza()`:

```
public abstract class PizzaStore {

   public abstract Pizza createPizza(String item);

   public Pizza orderPizza(String type) {
      Pizza pizza;
      // TO DO

      System.out.println("--- Making a " + pizza.getName() + " ---");
      pizza.prepare();
      pizza.bake();
      pizza.cut();
      pizza.box();
      return pizza;
   }
}
```

NYStylePizzaStore

```
public class NYStylePizzaStore extends PizzaStore {
   public Pizza createPizza(String item) {
      // TO DO
   }
}
```

ChicagoStylePizzaStore

```
public class ChicagoStylePizzaStore extends PizzaStore {
   public Pizza createPizza(String item) {
      // TO DO
   }
}
```

Viết lớp test và thực hiện:

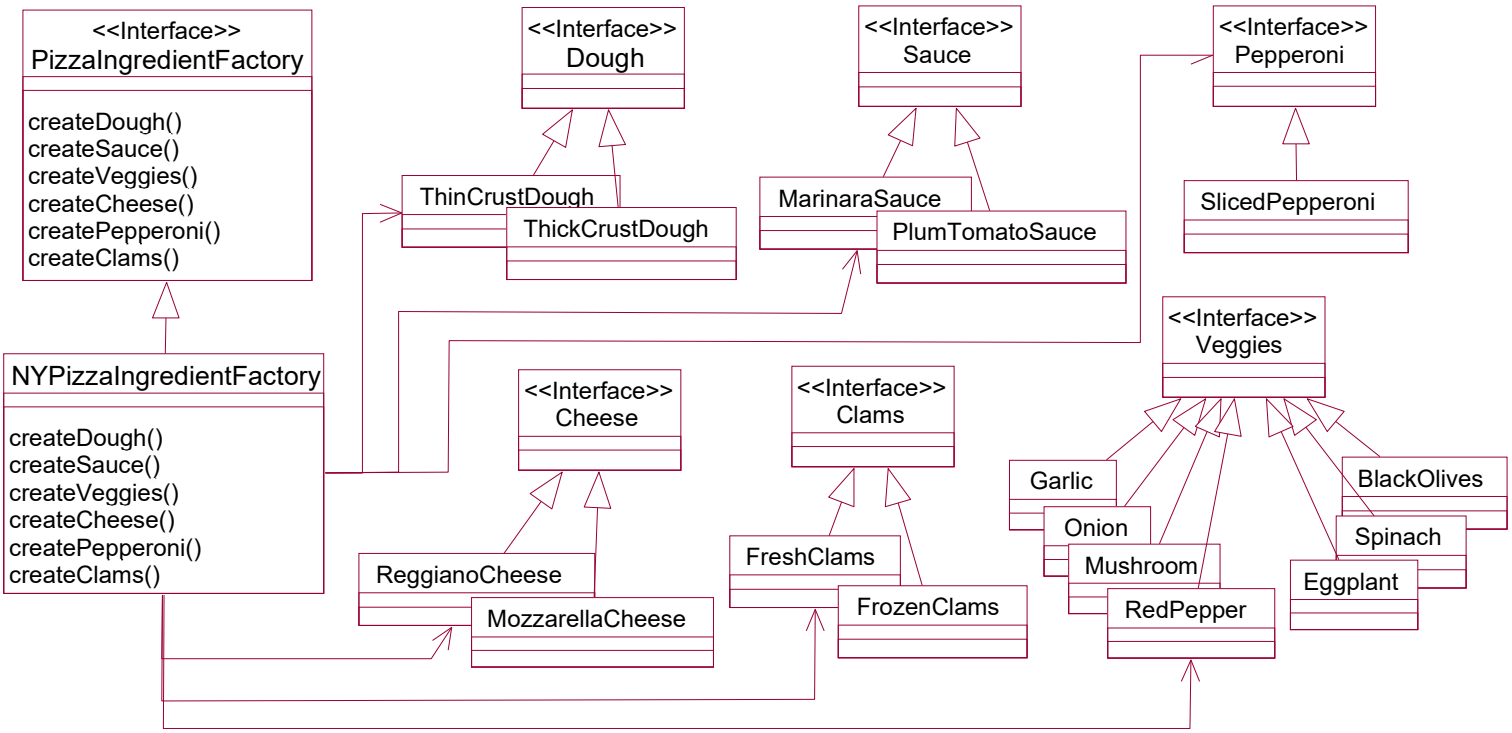
```
public class PizzaTestDrive {

   public static void main(String[] args) {
      PizzaStore nyStore = new NYStylePizzaStore();
      PizzaStore chicagoStore = new ChicagoStylePizzaStore();

      Pizza pizza = nyStore.orderPizza("cheese");
      System.out.println("Ethan ordered a " + pizza.getName() + "\n");

      pizza = chicagoStore.orderPizza("cheese");
      System.out.println("Joel ordered a " + pizza.getName() + "\n");

      pizza = nyStore.orderPizza("clam");
      System.out.println("Ethan ordered a " + pizza.getName() + "\n");

      pizza = chicagoStore.orderPizza("clam");
      System.out.println("Joel ordered a " + pizza.getName() + "\n");

      pizza = nyStore.orderPizza("pepperoni");
      System.out.println("Ethan ordered a " + pizza.getName() + "\n");

      pizza = chicagoStore.orderPizza("pepperoni");
      System.out.println("Joel ordered a " + pizza.getName() + "\n");

      pizza = nyStore.orderPizza("veggie");
      System.out.println("Ethan ordered a " + pizza.getName() + "\n");

      pizza = chicagoStore.orderPizza("veggie");
      System.out.println("Joel ordered a " + pizza.getName() + "\n");
   }
}
```

III) Abstract FACTORY

Bài toán:

- Cho phép bán các loại Pizza cho nhiều vùng khác nhau như Chicago và Newyork, …
- Các loại Pizza có thành phần chất liệu dough, cheese, sauce, veggies, … được làm khác nhau phụ thuộc vào vùng nơi cửa hàng đặt.

Dùng Abtract Factory Pattern để tạo các họ chất liệu (dough, sauce, cheese, …) khác nhau theo vùng NY, Chicago, …



1. Mở Eclipse, tạo một Java Project với tên **PizzaAF**
2. Tạo các interface và lớp thành phần nguyên liệu của **Pizza**: **Cheese**, **Sauce**, **Clams**, …

Cheese

```
public interface Cheese {
   public String toString();
}
```

ReggianoCheese

```
public class ReggianoCheese implements Cheese {
   public String toString() {
      return "Reggiano Cheese";
   }
}
```

MozzarellaCheese

```
public class MozzarellaCheese implements Cheese {
   public String toString() {
      return "Shredded Mozzarella";
   }
}
```

3. Tạo các lớp factory để tạo các thành phần nguyên liệu của Pizza cho NY và Chiacago:

PizzaIngredientFactory

```
public interface PizzaIngredientFactory {
   public Dough createDough();
   public Sauce createSauce();
   public Cheese createCheese();
   public Veggies[] createVeggies();
   public Pepperoni createPepperoni();
   public Clams createClam();
}
```

NYPizzaIngredientFactory

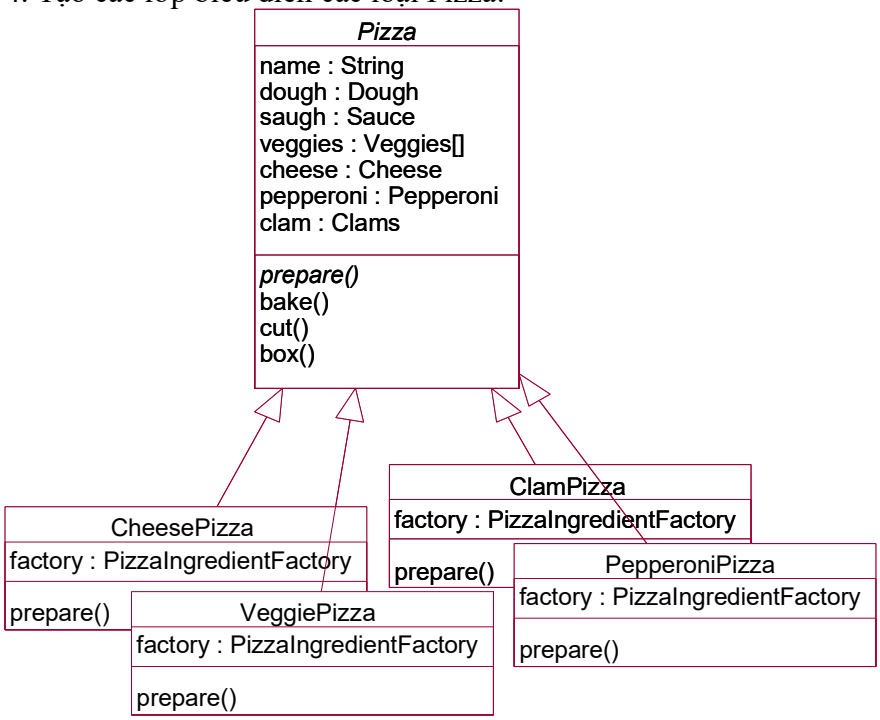
```
public class NYPizzaIngredientFactory implements PizzaIngredientFactory {
   public Dough createDough() {
      return new ThinCrustDough();
   }

   public Sauce createSauce() {
      return new MarinaraSauce();
   }

   public Cheese createCheese() {
      return new ReggianoCheese();
   }

   public Veggies[] createVeggies() {
      Veggies veggies[] =
             { new Garlic(), new Onion(), new Mushroom(), new RedPepper() };
      return veggies;
   }

   public Pepperoni createPepperoni() {
      return new SlicedPepperoni();
   }

   public Clams createClam() {
      return new FreshClams();
   }
}
```

ChicagoPizzaIngredientFactory

```
public class ChicagoPizzaIngredientFactory implements PizzaIngredientFactory {
   public Dough createDough() {
      return new ThickCrustDough();
   }

   public Sauce createSauce() {
      return new PlumTomatoSauce();
   }

   public Cheese createCheese() {
      return new MozzarellaCheese();
   }
```

```
    public Veggies[] createVeggies() {
        Veggies veggies[] = { new BlackOlives(), new Spinach(), new Eggplant() };
        return veggies;
    }

    public Pepperoni createPepperoni() {
        return new SlicedPepperoni();
    }

    public Clams createClam() {
        return new FrozenClams();
    }
}
```

4. Tạo các lớp biểu diễn các loại Pizza:



Pizza Class

```
public abstract class Pizza {
    protected String name;
    protected Dough dough;
    protected Sauce sauce;
    protected Veggies veggies[];
    protected Cheese cheese;
    protected Pepperoni pepperoni;
    protected Clams clam;

    abstract void prepare();

    public void bake() {
        System.out.println("Bake for 25 minutes at 350");
    }
```

```
    public void cut() {
        System.out.println("Cutting the pizza into diagonal slices");
    }

    public void box() {
        System.out.println("Place pizza in official PizzaStore box");
    }

    public void setName(String name) {
        this.name = name;
    }

    public String getName() {
        return name;
    }

    public String toString() {
        StringBuffer result = new StringBuffer();
        result.append("---- " + name + " ----\n");
        if (dough != null) {
            result.append(dough);
            result.append("\n");
        }
        if (sauce != null) {
            result.append(sauce);
            result.append("\n");
        }
        if (cheese != null) {
            result.append(cheese);
            result.append("\n");
        }
        if (veggies != null) {
            for (int i = 0; i < veggies.length; i++) {
                result.append(veggies[i]);
                if (i < veggies.length-1) {
                    result.append(", ");
                }
            }
            result.append("\n");
        }
        if (clam != null) {
            result.append(clam);
            result.append("\n");
        }
        if (pepperoni != null) {
            result.append(pepperoni);
            result.append("\n");
        }
        return result.toString();
    }
}
```

CheesePiza

```
public class CheesePizza extends Pizza {
   private PizzaIngredientFactory ingredientFactory;

   public CheesePizza(PizzaIngredientFactory ingredientFactory) {
      this.ingredientFactory = ingredientFactory;
   }

   public void prepare() {
      System.out.println("Preparing " + name);
      dough = ingredientFactory.createDough();
      sauce = ingredientFactory.createSauce();
      cheese = ingredientFactory.createCheese();
   }
}
```

ClamPizza

```
public class ClamPizza extends Pizza {
   private PizzaIngredientFactory ingredientFactory;

   public ClamPizza(PizzaIngredientFactory ingredientFactory) {
      this.ingredientFactory = ingredientFactory;
   }

   public void prepare() {
      System.out.println("Preparing " + name);
      dough = ingredientFactory.createDough();
      sauce = ingredientFactory.createSauce();
      cheese = ingredientFactory.createCheese();
      clam = ingredientFactory.createClam();
   }
}
```

PepperoniPizza

```
public class PepperoniPizza extends Pizza {
   private PizzaIngredientFactory ingredientFactory;

   public PepperoniPizza(PizzaIngredientFactory ingredientFactory) {
      this.ingredientFactory = ingredientFactory;
   }

   public void prepare() {
      System.out.println("Preparing " + name);
      dough = ingredientFactory.createDough();
      sauce = ingredientFactory.createSauce();
      cheese = ingredientFactory.createCheese();
      veggies = ingredientFactory.createVeggies();
      pepperoni = ingredientFactory.createPepperoni();
   }
}
```

VeggiePizza

```
public class VeggiePizza extends Pizza {
   private PizzaIngredientFactory ingredientFactory;

   public VeggiePizza(PizzaIngredientFactory ingredientFactory) {
      this.ingredientFactory = ingredientFactory;
   }

   public void prepare() {
      System.out.println("Preparing " + name);
      dough = ingredientFactory.createDough();
      sauce = ingredientFactory.createSauce();
      cheese = ingredientFactory.createCheese();
      veggies = ingredientFactory.createVeggies();
   }
}
```

5. Tạo PizzaStore

PizzaStore

```
public abstract class PizzaStore {
   protected abstract Pizza createPizza(String item);

   public Pizza orderPizza(String type) {
      Pizza pizza = createPizza(type);
      System.out.println("--- Making a " + pizza.getName() + " ---");
      pizza.prepare();
      pizza.bake();
      pizza.cut();
      pizza.box();
      return pizza;
   }
}
```

NYPizzaStore

```
public class NYPizzaStore extends PizzaStore {
   protected Pizza createPizza(String item) {
      Pizza pizza = null;
      PizzaIngredientFactory ingredientFactory = new NYPizzaIngredientFactory();

      if (item.equals("cheese")) {
         pizza = new CheesePizza(ingredientFactory);
         pizza.setName("New York Style Cheese Pizza");
      } else if (item.equals("veggie")) {
         pizza = new VeggiePizza(ingredientFactory);
         pizza.setName("New York Style Veggie Pizza");
      } else if (item.equals("clam")) {
         pizza = new ClamPizza(ingredientFactory);
         pizza.setName("New York Style Clam Pizza");
      } else if (item.equals("pepperoni")) {
         pizza = new PepperoniPizza(ingredientFactory);
         pizza.setName("New York Style Pepperoni Pizza");
      }
      return pizza;
   }
}
```

ChicagoPizzaStore

```
public class ChicagoPizzaStore extends PizzaStore {
   protected Pizza createPizza(String item) {
      Pizza pizza = null;
      PizzaIngredientFactory ingredientFactory = new ChicagoPizzaIngredientFactory();

      if (item.equals("cheese")) {
         pizza = new CheesePizza(ingredientFactory);
         pizza.setName("Chicago Style Cheese Pizza");
      } else if (item.equals("veggie")) {
         pizza = new VeggiePizza(ingredientFactory);
         pizza.setName("Chicago Style Veggie Pizza");
      } else if (item.equals("clam")) {
         pizza = new ClamPizza(ingredientFactory);
         pizza.setName("Chicago Style Clam Pizza");
      } else if (item.equals("pepperoni")) {
         pizza = new PepperoniPizza(ingredientFactory);
         pizza.setName("Chicago Style Pepperoni Pizza");
      }
      return pizza;
   }
}
```

6. Tạo lớp test

PizzaTestDrive

```
public class PizzaTestDrive {
   public static void main(String[] args) {
      PizzaStore nyStore = new NYPizzaStore();
      PizzaStore chicagoStore = new ChicagoPizzaStore();

      Pizza pizza = nyStore.orderPizza("cheese");
      System.out.println("Ethan ordered a " + pizza + "\n");

      pizza = chicagoStore.orderPizza("cheese");
      System.out.println("Joel ordered a " + pizza + "\n");

      pizza = nyStore.orderPizza("clam");
      System.out.println("Ethan ordered a " + pizza + "\n");

      pizza = chicagoStore.orderPizza("clam");
      System.out.println("Joel ordered a " + pizza + "\n");

      pizza = nyStore.orderPizza("pepperoni");
      System.out.println("Ethan ordered a " + pizza + "\n");

      pizza = chicagoStore.orderPizza("pepperoni");
      System.out.println("Joel ordered a " + pizza + "\n");

      pizza = nyStore.orderPizza("veggie");
      System.out.println("Ethan ordered a " + pizza + "\n");

      pizza = chicagoStore.orderPizza("veggie");
      System.out.println("Joel ordered a " + pizza + "\n");
   }
}
```