```java
package lab4;
/*
 * Banh, Alex
 * CECS 277
 * Professor Phuong Nguyen
 * 2 October, 2019
 */
import java.util.ArrayList;
/**
 * Stores multiple products and cash in the machine
 * @author alexb
 *
 */

public class VendingMachine {
    ArrayList<Product> products;
    public CoinSet coins;
    public CoinSet currentCoins;

    ArrayList<Product> productList;
    double change;
    /**
     * Default vending machine constructor
     */
    public VendingMachine() {
        products = new ArrayList<Product>(0);
        coins = new CoinSet();
        currentCoins = new CoinSet();
        productList = new ArrayList<Product>(0);
        change = 0;
    }
    /**
     * Add a new product to the vending machine
     * @param p The product to add
     * @param quantity How much of the product to add
     */
    public void addProduct(Product p, int quantity) {
        for(int i = 0; i < quantity; i++)
            products.add(p);

        if (!productList.contains(p))
            productList.add(p);
    }
    /**
     * Returns a product if you have enough cash
     * @param p Product to buy
     * @return Return the product you bought
     * @throws PriceException Not enough cash to buy it
     */
    public Product buyProduct(Product p) throws PriceException {

            Product tempProduct = null;
            for (int i = 0; i < products.size(); i++) {
                if(products.get(i).equals(p))
                {
                    if (currentCoins.getValue(currentCoins) < products.get(i).getPrice())
                    {
```

```
58                      System.out.println("Insufficent money");
59                      break;
60                  }
61                  else
62                  {
63                      tempProduct = products.get(i);
64                      products.remove(i);
65                      change = currentCoins.getValue(currentCoins) - tempProduct.getPrice();
66                      Coin cashEarned = new Coin(tempProduct.getPrice(), "Cash");
67                      coins.addCoin(cashEarned);
68                      currentCoins.removeAllCoins(currentCoins);
69
70                      ArrayList<Product> removedProd = new ArrayList<Product>(0);
71                      removedProd.addAll(products);
72                      products = removedProd;
73
74                      if(!products.contains(tempProduct))
75                          productList.remove(tempProduct);
76
77                      break;
78                  }
79              }
80          }
81          return tempProduct;
82      }
83
84  }
```