1 Customer

1.1 Instance Variables

1. `customerID` (**String**): a unique 4-digit number to identify customers.
2. `balance` (**double**): the balance in customer's account. (Default:0.0)
3. `status` (**boolean**): TRUE if the account is active and the customer is in good standing, FALSE otherwise. When this status is FALSE, account is frozen and no action is possible from the customer side. (Default: TRUE)

1.2 Constructors

`Customer`(**int** customerID, **double** balance): The status of a newly created customer is always TRUE unless manually changed by LaundryRoom.

1.3 Methods

1. `getCustomerID`
2. `getBalance`
3. `getStatus`
4. `setStatus`
5. `addMoney` (**double** amountToAdd → **void**): Add money to the account.

2 Machine (Abstract)

2.1 Instance Variables

1. `name` (**String**): a name to identify the machine.
2. `pricePerUse` (**double**): price for using this machine
3. `inUse` (**boolean**): TRUE if the machine is in use, FALSE otherwise.
4. `status` (**boolean**): TRUE if the machine is in good status; FALSE if the machine cannot be used (e.g. reserved, under maintenance, electricity outage, etc.)
5. `customerID` (**String**): the ID of the customer that is currently using this machine. Empty string if this machine is free.

2.2 Methods

1. `getName`
2. `getPricePerUse`
3. `getInUse`
4. `getStatus`
5. `getCustomerID`

6. `setPricePerUse`
7. `setStatus`
8. `setCustomerID`
9. **Abstract** `start`(**String** customerID, **String** mode→**String**)
10. **Abstract** `pause`(→**void**): Pause request by customers. Customers cannot stop a washing machine, but can pause a dryer.
11. **Abstract** `pause`(**String** `staffAccessPassword`→**void**): Pause request by staff. Can immediately pause any machine if the password is correct.

3 WashingMachine extends Machine

3.1 Instance Variables

1. `washMode` (**String**): The mode of wash. Empty string if this washing machine is not in use. Reset to empty string when wash is done.
2. `pausePassword` (**String**): A password to the pause function of this machine. A staff member can pause a wash in progress if they enters the password correctly.

3.2 Constructors

WashingMachine(**String** `machineID`, **double** `pricePerUse`, **String** `paucePassword`): Create a washing machine in the system. DEFAULT: `inUse` = FALSE, `status` = TRUE, `customerID` = "", `washMode` = "".

3.3 Methods

1. `getTemperature`
2. `setTemperature`
3. `start`(**String** CustomerID, **String** temperature→**String**): Try to start the machine. If successful, set the instance variables accordingly and return empty string. If unsuccessful, return the an error message explaining why the machine cannot be started.
4. `pause`(→**String**): return the error message "Wash cannot be paused before it's done. Please see a staff member for assistance."
5. `pause`(**String** `staffAccessPassword`→**String**): If the password is correct, pause the machine and set instance variables accordingly, return empty string. If the password is not correct, return "Invalid password".

4 Dryer extends Machine

4.1 Instance Variables

1. `temperature` (**String**): the temperature level set by customer before starting the dryer. VALUES: "High", "Medium", "Low", "Room". Empty string if the dryer is not in use.

4.2 Constructors

Dryer(**String** `machineID`, **double** `pricePerUse`): Create a washing machine in the system. DEFAULT: `inUse` = FALSE, `status` = TRUE, `customerID` = "", `temperature` = "".

4.3 Methods

1. `getWashMode`
2. `setWashMode`
3. `start`(**String** CustomerID, **String** mode→**String**): Try to start the machine. If successful, set the instance variables accordingly and return empty string. If unsuccessful, return the an error message explaining why the machine cannot be started.
4. `pause`(→**String**): Pause the dryer and set instance variables accordingly. Return empty string.
5. `pause`(**String** `staffAccessPassword`→**String**): Call the customer pause method.

5 LaundryRoom

5.1 Instance variables

1. customerList (List<Customer>)
2. machineList (List<Machine>)

5.2 Methods

1. `customerAddMoney(Customer customer, double toAdd → void)`: try to add money to Customer's account.
2. `startMachine`(**Customer** customer, **String** machineName → **void**): try to find the machine by machineName and start the machine. If successful, deduct the price from customer's account.
3. `pauseMachine`(**String** machineName → **void**): try to find the machine by its name and pause it.
4. `pauseMachine`(**String** password, **String** machineName → **void**): forced pause on machines by staff.