**Dikshya Acharya, 639044**

**Final Project**

This project is a follow up and compilation from HW1, HW2, HW3, HW4 where application class does manipulations add/remove/contains with the List ADT. The application is user to register/withdraw for a college fair. It takes input from keyboard as first name (String), last name (String) and Student Id(int). However, the ADT that was working with a concrete item type i.e. dikshyaStudentType in past, now is able to take a generic data type.

"A generic type is a class or interface that is parameterized over types, meaning that a type can be assigned by performing generic type invocation, which will replace the generic type with the assigned concrete type. The assigned type would then be used to restrict values being used within the container, which eliminates the requirement for casting, as well as provides stronger type-checking at compile time."--Oracle

**HW1**

We created an application class that calls an instance of CS420ListAdt class to create an array of given size. It performs an operations add/remove itemType dikshyaStudentType while also making sure the itemType exists in the list or not.

**Additions in HW2:**

Additionally, we also have two classes, first one being interface class named CS420ListInterface, and the CS420ListAdt implements it and second class is a custom created exception class named CS420Exception that extends Exception. The CS420Exception class checks the exception when the total size of the class exceeds the limit with the message. Hence, a custom exception is created with size limit(int) and a message(String). The main application class uses try/catch to deal with these exceptions and the exception is thrown from the method named add in CS420ListADT class when the size of the class exceeds the limit. Along with the custom exception, the application class also catches other exceptions that are likely to occur when expecting input from user.

**Addition in HW3:**

**Using generics**

What differentiates this project from HW1 and HW2 is that, the HW3 uses the generics data types. We used the CS420ListAdt<T> class that implements CS420ListInterface<T> for storing the elements of generic data type. However, the type it was taking on HW1 and HW2 was dikshyaStudentType taking String and int which are concrete types, is now replaced by generic types.

**Addition in HW4: |**

**Adding design patterns**

In this part, I have added two design patterns into my program.

1. Singleton class: I have made my ADT as a singleton class that is the instance of my CS420ListADT can and will be instantiated only once at the application class. The instructions have been provided in CS420ADTList class about the additional steps in creating singleton.

2. Factory design pattern: In Factory pattern, we create object without exposing the creation logic to client and the client use the same common interface to create new type of object. The idea is to use a static member-function (static factory method) which creates & returns instances, hiding the details of class modules from user.

"A factory pattern is one of the core design principles to create an object, allowing clients to create objects of a library (explained below) in a way such that it doesn’t have tight coupling with the class hierarchy of the library."

To implement this design pattern, I have created an abstract class called dikshyaUniversity class with an abstract typeRegistered(), paysTuition() method and bunch of defined methods. I then have dikshyaItemType class and dikshyaEmployeeType class that extends dikshyaUniversity and provide their own implementation of typeRegistered() and paysTuition() methods. The subclasses derived from dikshyaUniversity inherits the method defined in dikshyaUniversity class. Then I made a dikshyaUniversityBaseFactory class with a createItemType() factory method. This has been declared as abstract method. I then made the factory method implementation in concrete subclass of dikshyaUniversityFactory class which will be the concrete creator of the application. Thus I wrote createItemType() method in dikshyaUniversityFactory and used switch statement to create a dikshyaUniversity object based on the parameter passed to the method. Then, in our application class we can observe the factory design pattern at work. Application makes a different call to the factory method based on what the user provides from the keyboard.

The key advantage that I have learned is how the factory method pattern helps to encapsulate the object creation code from client code. Also, by using this pattern we will have a centralized location for object creation code, which will make it easy for us to debug and troubleshoot.

**Addition in HW5: |**

Along with all the components from earlier assignments, I have implemented additional features like a new CS420QueueADT that implements CS420Queueinterface and reading a data file from the

“Queue is a linear structure which follows a particular order in which the operations are performed. The order is **F**irst **I**n, **F**irst **O**ut (FIFO).  “

The Queue is created using bounded implementation (i.e. Array) and the primary purpose is to facilitate the application. As mentioned earlier the application, lets the user (student/employee) of the university to register for an event. However, the size of the original list is limited to add only 10 itemType, thus the queue helps additional itemtype to sign up for the waitlist. And in case when any itemType is removed from the list then the queue removes its first itemType and that is added to the original List for event. User is made aware of all the conditions that comes across.

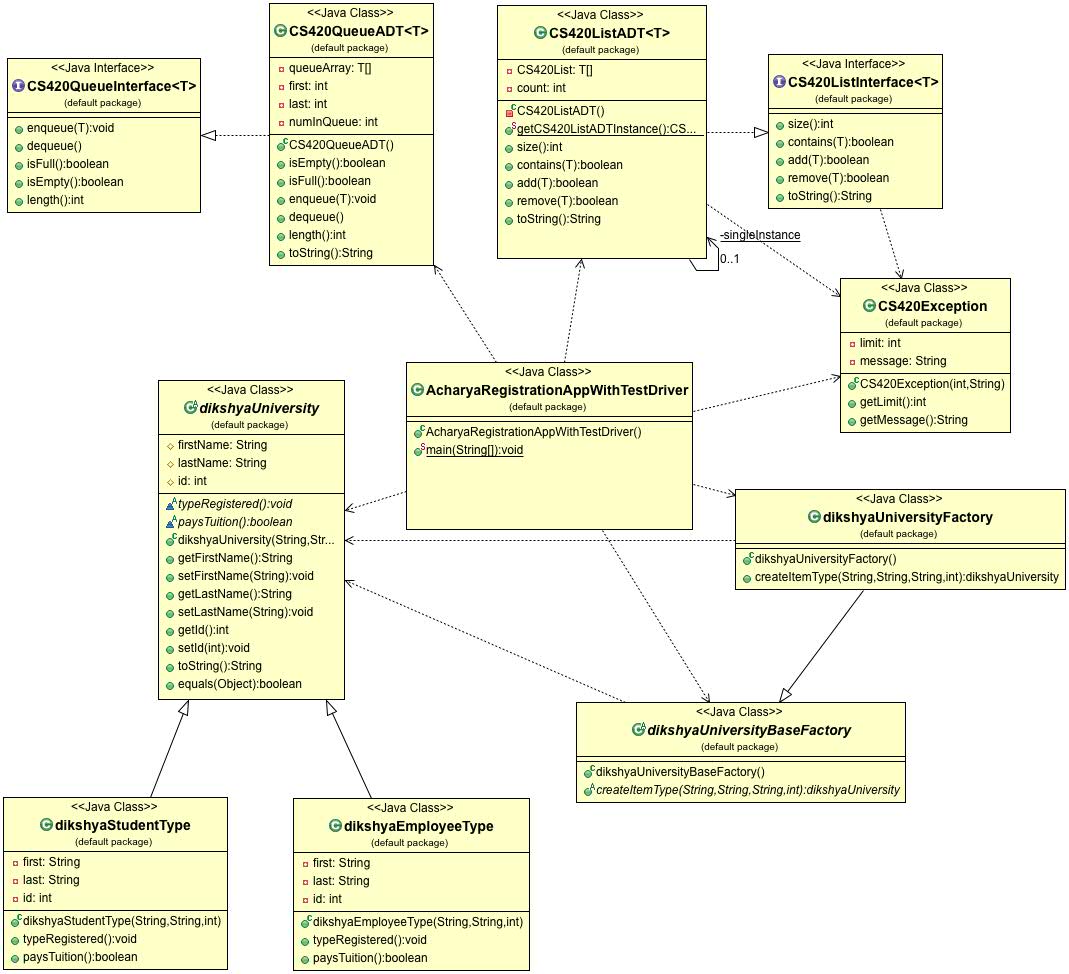
In short, the application flow goes as below:

1. A Generic List CS420ListADT is declared and initialized.
2. A generic Queue CS420QueueADT is declared and initialized.
3. Add onto the CS420ListADT by reading an external text file. The text file studentData1.txt has the list of names and Id of students and employees and it is stored into the list.
4. The addition ends as the list reaches the maximum limit of 10 and the custom exception is thrown, and the application step goes onto the next task.
5. Then the user is given a choice to sign up for waitlist in case, the space gets available later. That is where the enqueue method of queueADT starts working. Once the queue reaches maximum limited. The exception is thrown, and the flow of app continues on to the next step.
6. Then we provide an option for the withdrawal of name from the list ADT and at the same time once the space is available, the next itemType in queue is dequeued and added to the existing list. Also, the check is done for same id existing in the listADT.
7. The user selects the action as given
8. The list of registered users and the waiting list are shown, and the program terminates.

**Files included:**

1. AcharyaRegistrationAppWithTestDriver.java
2. AcharyaRegistrationAppWithTestDriver.class
3. dikshyaUniversity.java
4. dikshyaUniversity.class
5. dikshyaStudentType.java
6. dikshyaStudentType.class
7. dikshyaEmployeeType.java
8. dikshyaEmployeeType.class
9. CS420ListADT.java
10. CS420ListADT.class
11. CS420ListInterface.java
12. CS420ListInterface.class
13. CS420QueueADT.java
14. CS420QueueADT.class
15. CS420QueueInterface.java
16. CS420QueueInterface.class
17. CS420tException.java
18. CS420Exception.class
19. dikshyaUniversityBaseFactory.java
20. dikshyaUniversityBaseFactory.class
21. dikshyaUniversityFactory.java
22. dikshyaUniversityFactory.class
23. ProjectCompilation\_420\_OOD.doc
24. studentData.txt
25. DikshyaAcharya\_CS420\_ProjectCompilation.doc
26. HW5\_ClassDiagram.ucls
27. HW5\_UML\_DikshyaAcharya.jpg

**UML DIAGRAM for Project5**

****

**To run the application call:**

1.AcharyaRegistrationAppWithTestDriver.java, where the size of the ADT is bounded i.e however the program puts a limit to registration and thus is designed such that it will throw an exception when it reaches the given size limit with the help of custom exception. However, the app keeps on working as per the given directions.

**SAMPLE OUTPUT FOR THE APPLICATION SHOULD LOOK LIKE THIS:**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Welcome to CS-420Event Registration

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Total students/employees registered so far: 0

Barclay Liberty [554] added to the event list.

Employee registered.

Pays Tuition: false

Registered: 1

In waitlist: 0

Taylor Timothy [782] added to the event list.

Student registered.

Pays Tuition: true

Registered: 2

In waitlist: 0

Victoria Walker [977] added to the event list.

Student registered.

Pays Tuition: true

Registered: 3

In waitlist: 0

Tashya Martha [955] added to the event list.

Employee registered.

Pays Tuition: false

Registered: 4

In waitlist: 0

Latifah Rashad [679] added to the event list.

Employee registered.

Pays Tuition: false

Registered: 5

In waitlist: 0

Add error. Same Id exists

Cannot add to the list.

Registered: 5

In waitlist: 0

Elvis Gil [255] added to the event list.

Employee registered.

Pays Tuition: false

Registered: 6

In waitlist: 0

Add error. Same Id exists

Cannot add to the list.

Registered: 6

In waitlist: 0

Olympia May [923] added to the event list.

Student registered.

Pays Tuition: true

Registered: 7

In waitlist: 0

Add error. Same Id exists

Cannot add to the list.

Registered: 7

In waitlist: 0

Add error. Same Id exists

Cannot add to the list.

Registered: 7

In waitlist: 0

Paloma Nora [205] added to the event list.

Employee registered.

Pays Tuition: false

Registered: 8

In waitlist: 0

Orla Diana [595] added to the event list.

Student registered.

Pays Tuition: true

Registered: 9

In waitlist: 0

Troy Octavia [718] added to the event list.

Employee registered.

Pays Tuition: false

Registered: 10

In waitlist: 0

Maximum limit reached! 10 students registered.

The Event List registered:

Barclay Liberty [554]

Taylor Timothy [782]

Victoria Walker [977]

Tashya Martha [955]

Latifah Rashad [679]

Elvis Gil [255]

Olympia May [923]

Paloma Nora [205]

Orla Diana [595]

Troy Octavia [718]

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Welcome to the waitlist portal

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Please sign up for the waiting list.

Are you a [student] or [employee]: student

Enter the first name: Dikshya

Enter the last name: Acharya

Enter your id: 1286

Dikshya Acharya [1286] queued in the waitlist.

Add more? Enter: [1] to continue / [2] to stop: 1

Please sign up for the waiting list.

Are you a [student] or [employee]: employee

Enter the first name: Aarya

Enter the last name: Bhatt

Enter your id: 718

Aarya Bhatt [718] queued in the waitlist.

Add more? Enter: [1] to continue / [2] to stop: 1

Please sign up for the waiting list.

Are you a [student] or [employee]: student

Enter the first name: Paw

Enter the last name: wan

Enter your id: 1845

Paw wan [1845] queued in the waitlist.

Add more? Enter: [1] to continue / [2] to stop: 1

Please sign up for the waiting list.

Are you a [student] or [employee]: employee

Enter the first name: Kelly

Enter the last name: Khalisi

Enter your id: 1945

Kelly Khalisi [1945] queued in the waitlist.

Sorry the waitlist is full at the moment

Wait-List portal Closed.

First come, first serve.

The wait-list in queue:

Dikshya Acharya [1286]

Aarya Bhatt [718]

Paw wan [1845]

Kelly Khalisi [1945]

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Welcome to the withdraw portal for CS420 event!

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Are you a [student] or [employee]: student

Enter the first name: Victoria

Enter the last name: Walker

Enter your id: 977

Withdraw Successful

Adding the first person from the waitlist[CS420QueueADT] to eventlist[CS420ListADT]

Dikshya Acharya [1286] removed from the waitlist

Dikshya Acharya [1286] added to the event list.

Student registered.

Pays Tuition: true

Registered: 10

In waitlist: 3

Withdraw more? Enter: [1] to continue / [2] to stop: 1

Are you a [student] or [employee]: employee

Enter the first name: Tasya

Enter the last name: Martha

Enter your id: 657

Cannot withdraw, person not found.

Registered: 10

In waitlist: 3

Withdraw more? Enter: [1] to continue / [2] to stop: 1

Are you a [student] or [employee]: employee

Enter the first name: Barclay

Enter the last name: Liberty

Enter your id: 554

Withdraw Successful

Adding the first person from the waitlist[CS420QueueADT] to eventlist[CS420ListADT]

Aarya Bhatt [718] removed from the waitlist

Add error. Same Id exists

Cannot add to the list.

Adding the first person from the waitlist[CS420QueueADT] to eventlist[CS420ListADT]

Paw wan [1845] removed from the waitlist

Paw wan [1845] added to the event list.

Student registered.

Pays Tuition: true

Registered: 10

In waitlist: 1

Withdraw more? Enter: [1] to continue / [2] to stop: 2

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Event List

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

The total registered for the event: 10

Dikshya Acharya [1286]

Taylor Timothy [782]

Troy Octavia [718]

Tashya Martha [955]

Latifah Rashad [679]

Elvis Gil [255]

Olympia May [923]

Paloma Nora [205]

Orla Diana [595]

Paw wan [1845]

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Wait List

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

The total waitlisted for the event: 1

Kelly Khalisi [1945]

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

Thank you

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

----jGRASP: operation complete.