General guidelines:
1- This assignment should be done in teams of two to three max. Everyone in the team participates equally as stipulated in the syllabus. Read the syllabus for clarity.
2- Due date is February 01st, 2026 end of day.
3- You have to upload the source files and screenshots of the outputs to the blackboard.
4- Use Java to solve the assignments.
5- You have to use a diagram drawing tool for drawing the UML diagrams (e.g., Microsoft Visio). Visio should be free for Trent students, download it using your Trent username. You can also use web-based tools such as https://www.draw.io/ or https://www.lucidchart.com/

Question#1:

Briefly answer the following questions:

a) What are the important categories of architectural structures, briefly explain each category with at least one example.
b) What is an architectural pattern? Briefly describe a commonly used architectural pattern.
c) In the Introduction to Software Architecture lectures, we discussed that one of the issues not considered good practice is having too many inter-module calls between modules. Why is that? What are two good design principles that an architect should follow to avoid having a lot of inter-module calls?

Question#2:

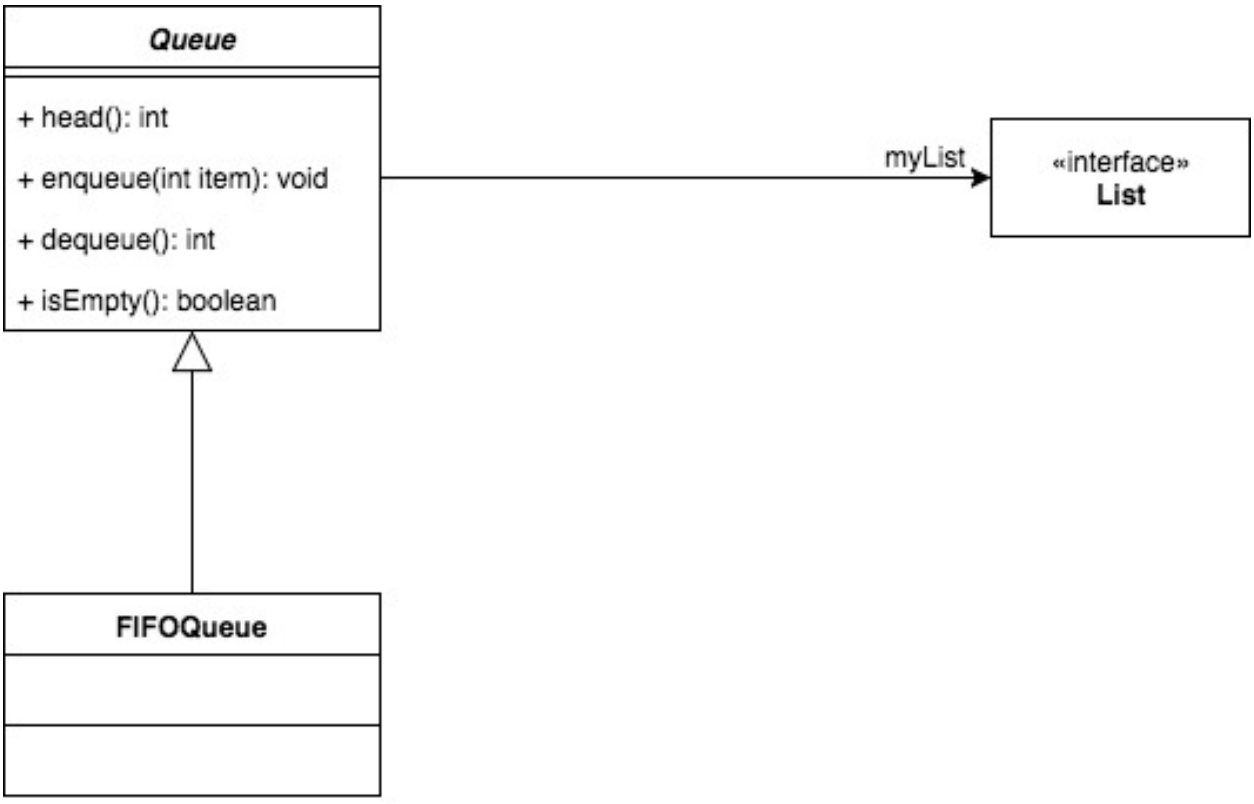
The List interface is defined as follows:

```
interface  List {
        int count();  //return the current number of elements in the list
        Object get(int index);  //return the object at the index in the list
        Object first();  //return the first object in the list
        Object last();   //return the last object in the list
        boolean include(Object obj);  //return true is the object in the list
        void append(Object obj); //append the object to the end of the list
        void prepend(Object obj);  //insert the object to the front of the list
        void delete(Object obj);  //remove the object from the list
        void deleteLast();  //remove the last element of the list
        void deleteFirst();  //remove the first element of the list
        void deleteAll();  //remove all elements of the list
}
```

(a) Write a class adapter (ArrayListAdapter.java) to adapt Java ArrayList to the List interface.

(b) Write a main program to test ArrayListAdapter through List interface.

(c) The following UML class diagram shows implementation of Bridge pattern for the Queue data structure. A queue enters elements to a list (the List interface shown above). A FIFOQueue ensures that the first element entered in the queue (through the enqueue() operation) is also the first element to exit the queue (when dequeue() operation is called).



Write program to implement the abstract class Queue and the concrete class FIFOQueue.

Use the following main program to test your implementation, assuming you use the ArrayListAdapter class that adapt Java ArrayList to the List interface.

```
List myList = new ArrayListAdapter();
Queue myQueue = new FIFOQueue(myList);
for (int i = 0; i < 100; i++) {
    int n = Math.random() * 100;
    myQueue.enqueue(n);
    System.out.print(n + " ");
}
System.out.println("FIFO output:");
while(! myQueue.isEmpty() {
   int n = myQueue.head();
   System.out.print(n + " ");
   myQueue.dequeue();
}
```

(d) Complete the UML diagram shown in (c) to include your solution for (a).

Question#2: Design and implement a Java program using Abstract Factory and Singleton design patterns.

The program displays date and time in one of the following two locations: Tokyo and Hawaii.

The following is how the program works. Then the program continuously asks the user to give one of the following commands, and performs the corresponding task. Note that the program gets the current date and time from the system clock (use the appropriate Java date and time operations for this).
'td': display Tokyo date
'tt': display Tokyo time
'hd': display Hawaii date
'ht': display Hawaii time
'q': quit the program.

- In the program, there should be 2 product hierarchies: “DateObject” and “TimeObject”. Each hierarchy should have locations described above. Calculate Hawaii and Tokyo date and time based on the current time. Hawaii time is current time – 5. Tokyo time is current time + 14.
- Implement the “DateTimeFactory” subclasses as singletons
- Draw a UML class diagram for the program.