[Publisher] Należy **napisać listę** (typ kolekcji, można utworzyć klasę pochodną z ArrayList), która powiadamia o każdej zmianie która w niej zaszła. Mianowicie, wysyła informacje o tym, które elementy zostały zmienione (listę zmienionych elementów), oraz czas, kiedy nastąpiła ta zmiana. W szczególności, należy przeciążyć metody `int Add(object value)`, `void Clear()`, oraz indexer `object this[int index]`.

[Subscriber 1] Należy **napisać** klasę **ListListener** która nasłuchuje, czy zaszła jakakolwiek zmiana w działaniu listy. Jeśli zmiana nastąpiła, należy wypisać wszystkie informacje jak poniżej:

```
Event fired at 11/28/2023 11:30:23 AM
Changed elements are:
1
2
```

[Subscriber 2] Należy **napisać** klasę **ListListenerSaveToFile** która nasłuchuje, czy zaszła jakakolwiek zmiana w działaniu listy. Jeśli zmiana nastąpiła, należy zapisać informacje o zmianach do pliku podanego jako parametr konstruktora.

[Testing Console Application] Następnie należy **napisać program** który testuje utworzoną klasę. W programie należy:

- utworzyć listę oraz obiekt, który nasłuchuje zmian które w niej zachodzą
- utworzyć obiekt który będzie zapisywał zmiany zachodzące w liście do pliku list.txt
- dodać 3 dowolne elementy do listy (jeden po drugim)
- zmodyfikować drugi element używając indeksera
- zatrzymać wykonywanie programu na sekundę: Thread.Sleep(1000);
- usunąć wszystkie elementy listy
- zrezygnować z nasłuchiwania zmian
- dodać 1 dowolny element

Przykładowy wynik działania programu:

```
----adding elements----
Event fired at 11/28/2023 11:40:13 AM
Changed elements are:
1
SAVING TO FILE...
Event fired at 11/28/2023 11:40:13 AM
Changed elements are:
2
SAVING TO FILE...
Event fired at 11/28/2023 11:40:13 AM
Changed elements are:
3
SAVING TO FILE...
----indexer----
Event fired at 11/28/2023 11:40:13 AM
Changed elements are:
0
SAVING TO FILE...
----clear----
Event fired at 11/28/2023 11:40:15 AM
Changed elements are:
1
0
3
SAVING TO FILE...
```

Podpowiedzi:

1. Event

Utworzyć klasę `ListEventArgs` przechowującą datę+czas (DateTime) oraz listę zmienionych elementów (ArrayList) – 2 składniki. Patrz slajd 19.

2. Publisher

Dziedzicząc po klasie ArrayList należy utworzyć nową klasę, zawierającą (patrz slajd 20):

- delegat ListChangeHandler przyjmujący obiekt oraz event (obiekt klasy utworzony w punkcie 1)
- konkretny egzemplarz event-u utworzonego w punkcie 1
- metodę OnListChanged która odpala event
- przeciążone metody: Add, Clear oraz indexer
  - w metodach tych należy stworzyć instancję eventu oraz odpowiednio ustawić jego pola: datę+czas oraz listę zmodyfikowanych elementów a następnie uruchomić metodę OnListChanged

3. Subscriber 1

Odpowiednik klasy znajdującej się na slajdzie 21. Powinien zawierać:

- Konstruktor, przyjmujący obiekt zmodyfikowanej listy.
- Metodę Subscribe
- Metodę implementującą delegowaną funkcjonalność: np. `void ListChanged(object sender, ListEventArgs e)` która będzie wypisywać na ekranie żądane informacje.
- Metodę Unsubscribe/Detach – działającą tak samo jak Subscribe tylko odłączającą nasłuchiwanie (patrz slajd 11)

4. Subscriber 2

Klasa bardzo podobna do Subscriber 1 tylko zapisująca do pliku. Zapisywanie do pliku można prześledzić na slajdze 10 i 27.