P2W5 Opdracht TodoList

We gaan een programma schrijven waarmee we onze Todo's kunnen opvolgen. Een TodoItem heeft een omschrijving, prioriteit en deadline. Alle TodoItems worden een beheerd door de klasse TodoList. Je mag géén TodoItem kunnen aanmaken of aanspreken buiten de klasse TodoList.

Deze opdracht gaat zowel over Collections en (Linked)List , maar behandeld ook leerstof van de voorgaande weken (Exception, Inner Classes, Date API)

1. Maak een klasse TodoList aan
2. Voorzie daarin een private inner klasse TodoItem met volgende implementatie
    a. Drie private eigenschappen "description" (`String`), "priority" (`int`) en "deadline" (`LocalDate`). Enkel voor de eigenschap "deadline" moet een accessor methode voorzien worden, voor alle andere eigenschappen niets
    b. Een constructor die een "description" (`String`), "priority" (`int`) en "deadline" (`String`) als parameter meekrijgt. De deadline wordt ingegeven in het formaat 'dag/maand/jaar'. Die verschillende datum-onderdelen moet je zelf kunnen achterhalen uit deze String en dan omzetten naar een manier waarop je een LocalDate aangemaakt kan worden.
    TIP: `String.split` en `Integer.parseInt`
    c. Als de opgegeven deadline een ligt vóór de huidige datum, gooi dan een `IllegalArgumentException`
    d. Maak een `toString`() methode die een TodoItem als volgt afdrukt:
```
1     Uitblazen na examens P2                  2015-01-19
{priority: 5 plaatsen, links uitgelijnd}
{description: 40 plaatsen, links uitgelijnd}
{deadline}
```
    Gebruik de methode `String.format` om deze afdruk mooi te krijgen.
3. Zorg ook voor een geneste enum in de klasse TodoList. De naam van die enum is "TodoItemOrder" en zijn mogelijke waarden zijn "HIGH_TO_LOW", "LOW_TO_HIGH" en "DEADLINE"
4. Maak in de klasse TodoList een private eigenschap todoItems aan. Het datatype van deze eigenschap is `LinkedList`. Initialiseer deze eigenschap in de constructor
5. In deze klasse TodoList maak je een ook methode "addTodoItem" aan:
    a. Deze gaat op basis van de parameters "description" (`String`), "priority" (`int`) en "deadline" (`String`) een nieuw TodoItem aanmaken en deze op de juiste plaats in de lijst van todoItems inserteren
    b. De juiste plaats bepaal je doordat de `LinkedList` moet ingevuld wordt op basis van de prioriteit. De hoogste prioriteit (= cijfer 1) moet het eerste element zijn, een lagere prioriteit (bv. 10) moet daarna ingevoegd worden
    c. Gebruik géén `sort`, maar zoek steeds op waar het nieuw in te voeren element in de lijst moet tussengevoegd worden

6. Maak tot slot een methode ‘getAllTodoItems’ waarin je – op basis van een parameter van het type ‘TodoItemOrder’ een `String` samenstelt. Toon van elk TodoItem de uitvoer van zijn toString methode
    a. Bij de optie ‘HIGH_TO_LOW’ en ‘LOW_TO_HIGH’ geef je alle items van hoogste naar laagste respectievelijk van laagste naar hoogste prioriteit terug als `String`. Je mag de lijst zelf NIET hersorteren (**TIP:** gebruik `Iterator`)
    b. Bij de optie ‘DEADLINE’ zorg je ervoor dat je het TodoItem dat als eerste een deadline heeft eerst getoond wordt, enzovoort... Ook hier mag je de originele lijst niet sorteren, maar niets belet je om een copy van de originele lijst in een gewone array te nemen en bijvoorbeeld die te sorteren ☺.
    Probeer ook gebruik te maken van een anonieme inner class om de sorteringslogica te schrijven zodanig dat je de klasse TodoItem niet hoef aan te passen.

De code van de main-methode krijg je. Met deze code moet je ook de verwachte uitvoer verkrijgen.

```
public static void main(String[] args) {
    TodoList ourList = new TodoList();
    ourList.addTodoItem("Java OO Technieken leren", 5, "8/1/2015");
    ourList.addTodoItem("Java Geavanceerde OO Technieken leren", 10, "16/3/2015" );
    ourList.addTodoItem("Java game binnenleveren", 8, "10/3/2015");
    ourList.addTodoItem("Uitblazen na examens P2", 1, "19/1/2015");

    System.out.println("===== Alle todo-item van hoogste prioriteit naar laagste");
    System.out.println(ourList.getAllTodoItems(TodoList.TodoItemOrder.HIGH_TO_LOW));
    System.out.println("===== Alle todo-item van laagste prioriteit naar hoogstse");
    System.out.println(ourList.getAllTodoItems(TodoList.TodoItemOrder.LOW_TO_HIGH));
    System.out.println("===== Alle todo-item gesorteerd volgens deadline");
    System.out.println(ourList.getAllTodoItems(TodoList.TodoItemOrder.DEADLINE));
    System.out.println("===== Alle todo-item van laagste prioriteit naar hoogstse");
    System.out.println(ourList.getAllTodoItems(TodoList.TodoItemOrder.LOW_TO_HIGH));

    ourList.addTodoItem("Blokken voor Java Basisbegrippen", 1, "7/11/2014");
}
```

Verwachte uitvoer

```
===== Alle todo-item van hoogste prioriteit naar laagste
1     Uitblazen na examens P2                  2015-01-19
5     Java OO Technieken leren                 2015-01-08
8     Java game binnenleveren                  2015-03-10
10    Java Geavanceerde OO Technieken leren    2015-03-16

===== Alle todo-item van laagste prioriteit naar hoogstse
10    Java Geavanceerde OO Technieken leren    2015-03-16
8     Java game binnenleveren                  2015-03-10
5     Java OO Technieken leren                 2015-01-08
1     Uitblazen na examens P2                  2015-01-19

===== Alle todo-item gesorteerd volgens deadline
5     Java OO Technieken leren                 2015-01-08
1     Uitblazen na examens P2                  2015-01-19
8     Java game binnenleveren                  2015-03-10
10    Java Geavanceerde OO Technieken leren    2015-03-16

===== Alle todo-item van laagste prioriteit naar hoogstse
10    Java Geavanceerde OO Technieken leren    2015-03-16
8     Java game binnenleveren                  2015-03-10
5     Java OO Technieken leren                 2015-01-08
1     Uitblazen na examens P2                  2015-01-19

Exception in thread "main" java.lang.IllegalArgumentException: Deadline kan
niet voor vandaag liggen
      at be.kdg.Todo.TodoList$TodoItem.<init>(TodoList.java:28)
      at ...
```