WEEK 8

TUTORCLASS TC = NEW TUTORCLASS("WEEK8");

TODAY’S PLAN

- Unit Testing Demo
- Object Orientated Programming Explanation
    - Access to Variables Demo
- Immutable Set
- Timetable

UNIT TESTING

- Tests the Inputs and Outputs of a Function
  - Checks if the output is what is expected
  - Of course this only works on deterministic functions

UNIT TESTING

- Junit Library is used

UNIT TESTING – TEST BASED PROGRAMMING

- It forces you to consider what each functions input and output is long before
- Also makes you consider the interaction of edge cases
- Automates Testing → Don't have to enter everything manually

UNIT TEST

- Demo

OBJECT ORIENTATED PROGRAMMING

- Blueprint vs Instance

VARIABLE ACCESS

VARIABLE ACCESS

- Demo

IMMUTABLE SET

- Think of it as a set of Strings
  - [aa, AA, bb]
- Rule of Sets: All elements are unique

IMMUTABLE SET – IMPLEMENT

- Constructor – Creates a empty set
- Boolean isElement(String s) – Is s included in set
- Boolean superset(ImmutableSet subset) – are all elements of subset found within the current set
- Boolean isEqual(ImmutableSet other) – Are all elements of other found in the current set
- Void add(String s) – Append s to the set if s is not currently found in the set
- String toString() – Represent the set as a String in the following format [a, b, c]

IMMUTABLE SET – TESTS

- I have written for you a unit test that will test your code
    - Found on my website in this weeks folder

IMMUTABLE SET – IMPLEMENT

- Constructor – Creates a empty set
- Boolean isElement(String s) – Is s included in set
- Boolean superset(ImmutableSet subset) – are all elements of subset found within the current set
- Boolean isEqual(ImmutableSet other) – Are all elements of other found in the current set
- Void add(String s) – Append s to the set if s is not currently found in the set
- String toString() – Represent the set as a String in the following format [a, b, c]

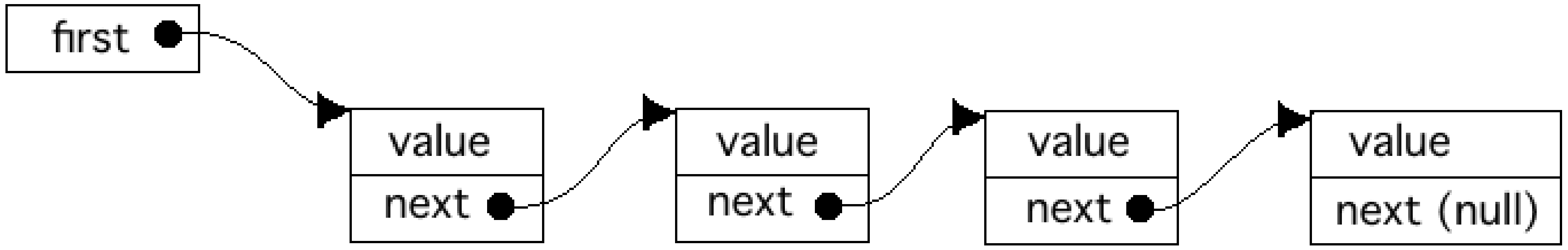
TIME TABLE

- This is a two part task
  - First Implement the class Date
  - Then Implement the class Timetable
    - With a subclass DateList

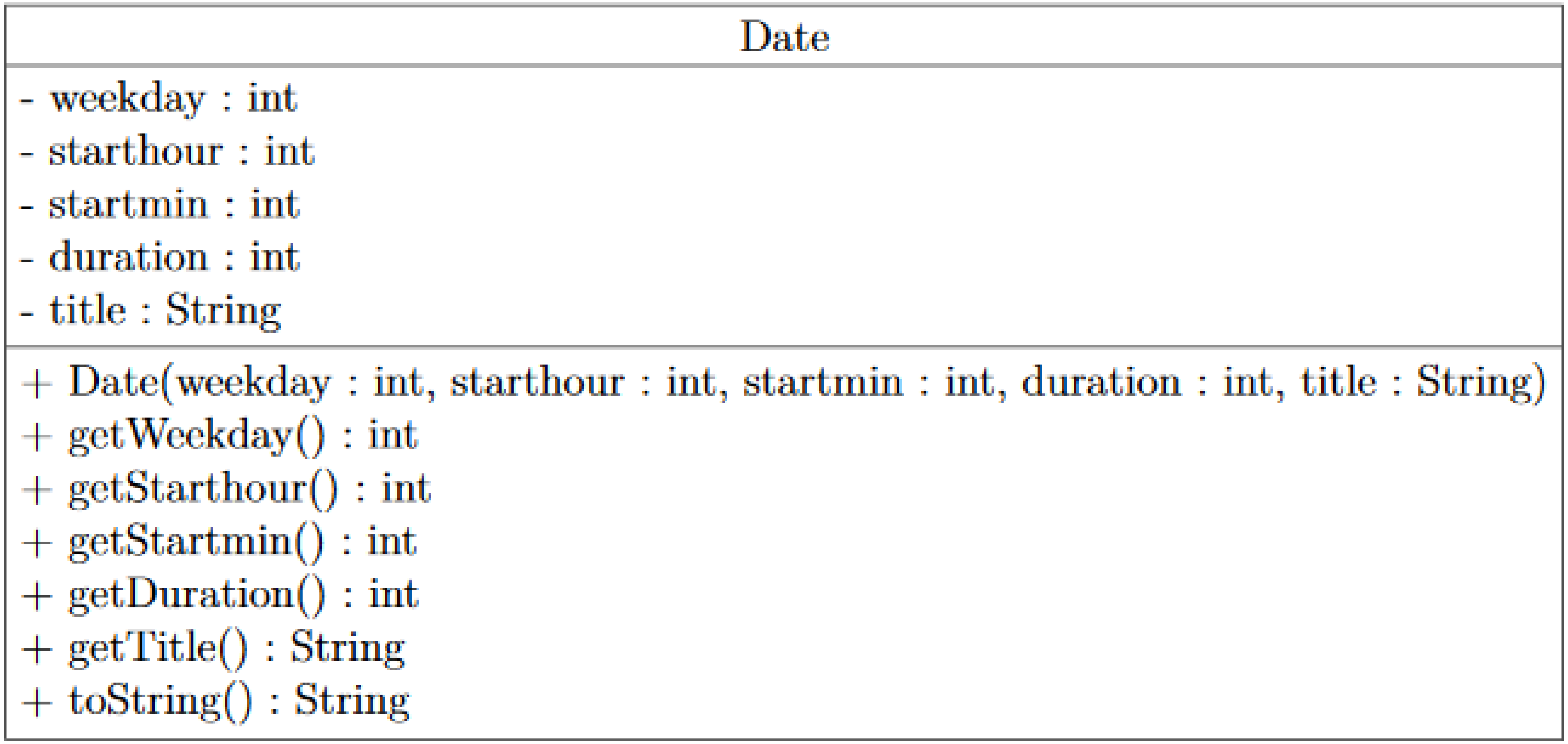
TIME TABLE – LIST REFRESHER

- What is a list?

TIME TABLE – LIST REFRESHER

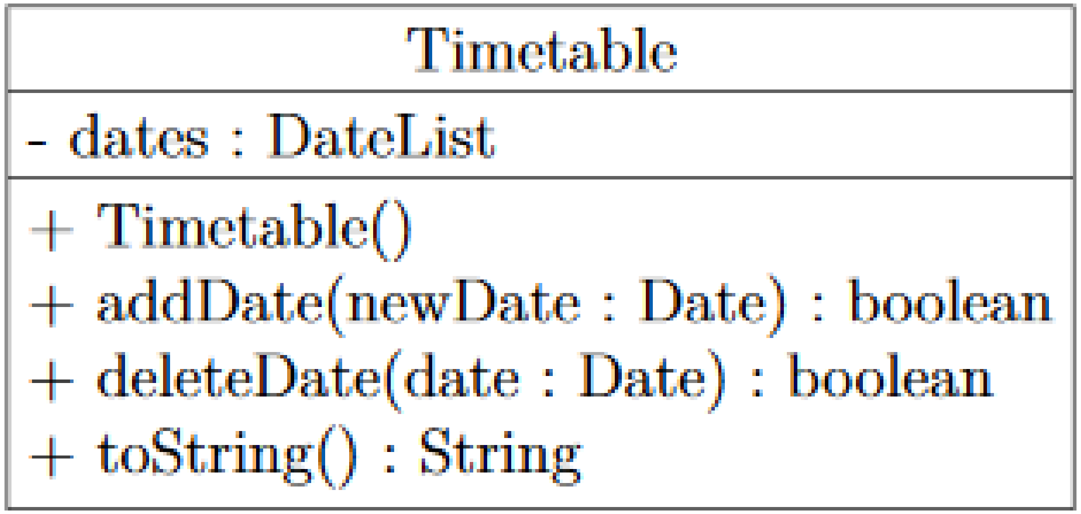


THE CLASS DATE



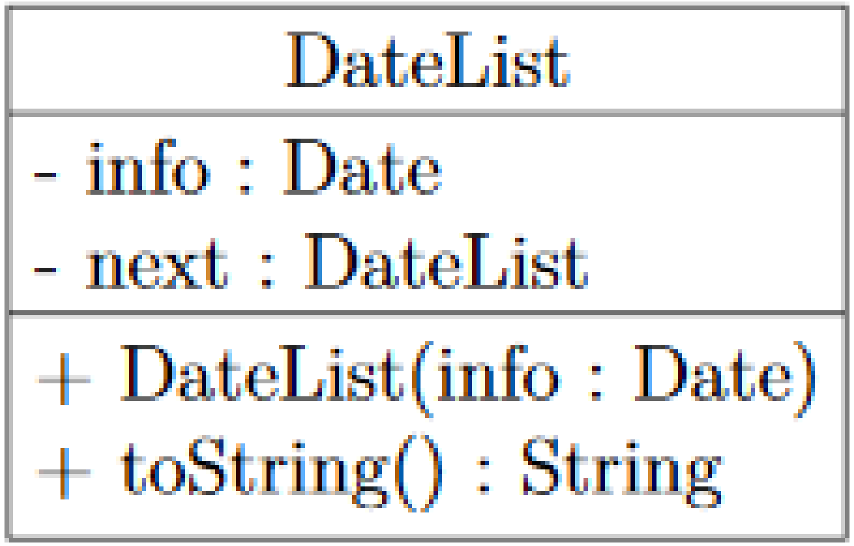
```
Date
- weekday : int
- starthour : int
- startmin : int
- duration : int
- title : String
+ Date(weekday : int, starthour : int, startmin : int, duration : int, title : String)
+ getWeekday() : int
+ getStarthour() : int
+ getStartmin() : int
+ getDuration() : int
+ getTitle() : String
+ toString() : String
```

TIME TABLE CLASS



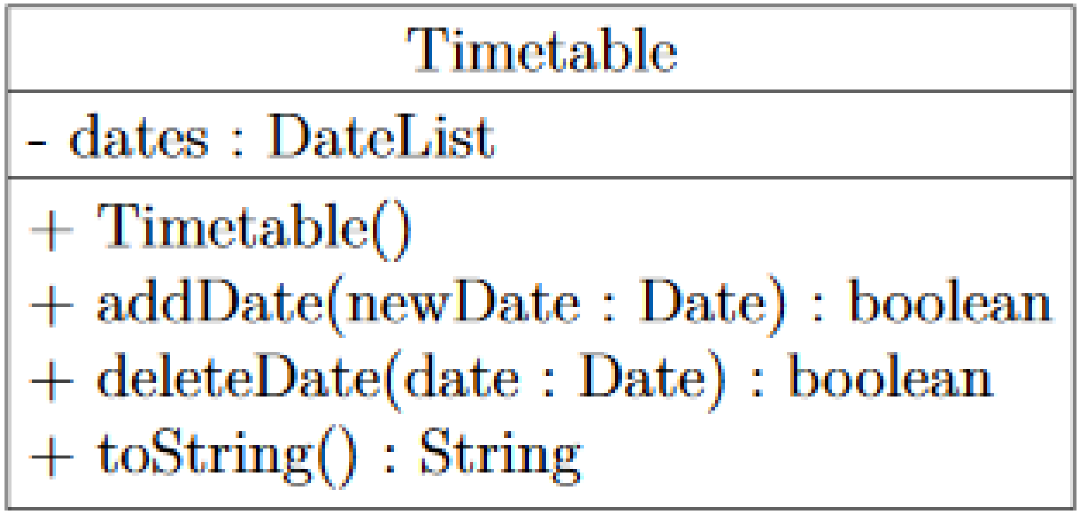
Timetable
- dates : DateList
+ Timetable() + addDate(newDate : Date) : boolean + deleteDate(date : Date) : boolean + toString() : String

DATE LIST CLASS



```
DateList
- info : Date
- next : DateList
+ DateList(info : Date)
+ toString() : String
```

TIME TABLE CLASS



Timetable

- dates : DateList

+ Timetable()
+ addDate(newDate : Date) : boolean
+ deleteDate(date : Date) : boolean
+ toString() : String