# Opdracht 1: Uitwerking van de voorbeeldspecificatie ‘dagen tot mijn verjaardag.’

NB: In dit voorbeeld is geen rekening gehouden met schrikkeljaren. De specificatie is dus niet compleet! We accepteren dat voor deze keer.

/\*\*

\* @desc Calculates the amount of days until the next birthday of dateOfBirth.

\*

\* @subcontract: not born yet {

\* @requires dateOfBirth > LocalDate.now();

\* @signals (IllegalArgumentException) dateOfBirth > LocalDate.now();

\* }

\*

\* @subcontract: null dateOfBirth {

\* @requires dateOfBirth == null;

\* @signals (NullPointerException) dateOfBirth == null;

\* }

\*

\* @subcontract: birthday is yet to come this year {

\* @requires LocalDate.now().getMonthValue() < dateOfBirth.getMonthValue() ||

\* LocalDate.now().getMonthValue() == dateOfBirth.getMonthValue() &&

\* LocalDate.now().getDayOfMonth() < dateOfBirth.getDayOfMonth();

\* @ensures **\result** = #days until (not including) next birthday && 1 <= **\result** < 365;

\* }

\*

\* @subcontract: today is my birthday {

\* @requires dateOfBirth.getMonthValue() == LocalDate.now().getMonthValue() &&

\* dateOfBirth.getDayOfMonth() == LocalDate.now().getDayOfMonth();

\* @ensures **\result** = 365;

\* }

\*

\* @subcontract: birthday has already passed this year {

\* @requires dateOfBirth.getMonthValue() < LocalDate.now().getMonthValue() ||

\* dateOfBirth.getMonthValue() == LocalDate.now().getMonthValue() &&

\* dateOfBirth.getDayOfMonth() < LocalDate.now().getDayOfMonth();

\* @ensures **\result** = #days until (not including) next birthday && 1 <= **\result** < 365;

\* }

\*

public static int daysUntilNextBirthday**(**/\*@ non null @\*/ LocalDate dateOfBirth**);**

# Sjabloon opstellen tests opdracht 1

|  |  |
| --- | --- |
| **Testcase** | Not born yet |
| **Preconditie** | dateOfBirth = LocalDate.now().plusDays(10) |
| **Orakel** | IllegalArgumentException |
| **Postconditie of Exception** | IllegalArgumentException |

|  |  |
| --- | --- |
| **Testcase** | Null dateOfBirth |
| **Preconditie** | dateOfBirth = null |
| **Orakel** | NullReferenceException |
| **Postconditie of Exception** | Is niet te testen cli |

|  |  |
| --- | --- |
| **Testcase** | birthday is yet to come this year |
| **Preconditie** | dateOfBirth = LocalDate.now().minusYears(10).plusDays(10) |
| **Orakel** | 10 |
| **Postconditie of Exception** |  |

|  |  |
| --- | --- |
| **Testcase** | today is my birthday |
| **Preconditie** | dateOfBirth = LocalDate.now().minusYears(10) |
| **Orakel** |  |
| **Postconditie of Exception** |  |

|  |  |
| --- | --- |
| **Testcase** | birthday has already passed this year |
| **Preconditie** |  |
| **Orakel** |  |
| **Postconditie of Exception** |  |

|  |  |
| --- | --- |
| **Testcase** |  |
| **Preconditie** |  |
| **Orakel** |  |
| **Postconditie of Exception** |  |

# Opdracht 2: ‘Annual Bonus’

/\*\*

\* @desc A company orders an application that needs to calculate the annual bonus

\* of its employees. This bonus is a percentage of their monthly salary, and

\* depends on how long they have worked for the company.

\*

\* @subcontract: negative years in service {

\* @requires yearsInService < 0;

\* @signals (IllegalArgumentException) yearsInService < 0;

\* }

\*

\* @subcontract: less or equal than three years at the company yields a bonus of 0% {

\* @requires 0 <= yearsInService <= 3;

\* @ensures **\result** = 0;

\* }

\*

\* @subcontract: more than three years at the company yields a bonus of 50% {

\* @requires 3 < yearsInService <= 5;

\* @ensures **\result** = 50;

\* }

\*

\* @subcontract: more than five years yields a bonus of 75% {

\* @requires 5 < yearsInService <= 8;

\* @ensures **\result** = 75;

\* }

\*

\* @subcontract: more than eight years yields a bonus of 100% {

\* @requires 8 < yearsInService;

\* @ensures **\result** = 100;

\* }

\*/

public static int getAnnualBonusPercentage**(**int yearsInService**);**

|  |  |
| --- | --- |
| **Testcase** | Test negative years in service |
| **Preconditie** | yearsInService (-1) < 0 |
| **Orakel** | IllegalArgumentException |
| **Postconditie of Exception** | IllegalArgumentException (yearsInService must be positive) |

|  |  |
| --- | --- |
| **Testcase** | less or equal than three years at the company yields a bonus of 0 |
| **Preconditie** | 0 <= yearsInService <= 3; |
| **Orakel** |  |
| **Postconditie of Exception** |  |

|  |  |
| --- | --- |
| **Testcase** |  |
| **Preconditie** |  |
| **Orakel** |  |
| **Postconditie of Exception** |  |

|  |  |
| --- | --- |
| **Testcase** | more than eight years yields a bonus of 100% |
| **Preconditie** | 8 < yearsInService |
| **Orakel** |  |
| **Postconditie of Exception** |  |

|  |  |
| --- | --- |
| **Testcase** |  |
| **Preconditie** |  |
| **Orakel** |  |
| **Postconditie of Exception** |  |

|  |  |
| --- | --- |
| **Testcase** |  |
| **Preconditie** |  |
| **Orakel** |  |
| **Postconditie of Exception** |  |

# Opdracht 3: ‘Hardware store Discount’

A hardware store sells hammers **(**5 euros**)** and screwdrivers **(**10 euros**).** Over time however**,** their discount system has grown a bit complex**.**

They have asked the nephew of the boss **(**who is studying computer science**)** to develop a little application that can calculate the price a customer needs to pay when buying these products**.** They have the following discount rules**:**

• If the total is more than 200 euros**,** then the client obtains a discount of 5**%** over the total**.**

• If the total is more than 1000 euros**,** then the client obtains a discount of 20**%** over the total**.**

• If the client buys more than 30 screwdrivers**,** then there is an additional discount of 10**%.**

/\*\*

\* @desc A hardware store sells hammers (5 euros) and screwdrivers (10 euros).

\* This method can calculate the price a customer needs to pay when buying

\* these products.

\*

\* @subcontract: invalid input negative totalDue {

\* @requires totalDue < 0.0;

\* @signals (IllegalArgumentException) totalDue < 0.0;

\* }

\*

\* @subcontract: invalid input negative amount of screwdrivers {

\* @requires amountOfScrewdrivers < 0;

\* @signals (IllegalArgumentException) amountOfScrewdrivers < 0;

\* }

\*

\* @subcontract: If the total is less than 200 euros, no discount is applicable {

\* @requires 0.0 < totalDue <= 200.0;

\* @ensures **\result** = totalDue;

\* }

\*

\* @subcontract: If the total is more than 200 euros and buys no more than 30 screwdrivers,

\* then the client obtains a discount of 5% over the total {

\* @requires 200.0 < totalDue <= 1000.0 && 0 <= amountOfScrewdrivers <=30;

\* @ensures **\result** = totalDue \* 0.95;

\* }

\*

\* @subcontract: If the total is more than 200 euros and buys more than 30 screwdrivers,

\* then the client obtains a discount of 15% over the total {

\* @requires 200.0 < totalDue <= 1000.0 && 30 < amountOfScrewdrivers;

\* @ensures **\result** = totalDue \* 0.85;

\* }

\*

\* @subcontract: If the total is more than 1000 euros and buys no more than 30 screwdrivers,

\* then the client obtains a discount of 20% over the total {

\* @requires 1000.0 < totalDue && 0 <= amountOfScrewdrivers <=30;

\* @ensures **\result** = totalDue \* 0.80;

\* }

\*

\* @subcontract: If the total is more than 1000 euros and buys more than 30 screwdrivers,

\* then the client obtains a discount of 30% over the total {

\* @requires 1000.0 < totalDue && 30 < amountOfScrewdrivers;

\* @ensures **\result** = totalDue \* 0.70;

\* }

\*

\*/

public static double getDiscountedPrice**(**double totalDue**,** int amountOfScrewdrivers**);**

|  |  |
| --- | --- |
| **Testcase** |  |
| **Preconditie** |  |
| **Orakel** |  |
| **Postconditie of Exception** |  |

|  |  |
| --- | --- |
| **Testcase** |  |
| **Preconditie** |  |
| **Orakel** |  |
| **Postconditie of Exception** |  |

|  |  |
| --- | --- |
| **Testcase** |  |
| **Preconditie** |  |
| **Orakel** |  |
| **Postconditie of Exception** |  |

|  |  |
| --- | --- |
| **Testcase** |  |
| **Preconditie** |  |
| **Orakel** |  |
| **Postconditie of Exception** |  |

|  |  |
| --- | --- |
| **Testcase** |  |
| **Preconditie** |  |
| **Orakel** |  |
| **Postconditie of Exception** |  |

|  |  |
| --- | --- |
| **Testcase** |  |
| **Preconditie** |  |
| **Orakel** |  |
| **Postconditie of Exception** |  |

|  |  |
| --- | --- |
| **Testcase** |  |
| **Preconditie** |  |
| **Orakel** |  |
| **Postconditie of Exception** |  |

|  |  |
| --- | --- |
| **Testcase** |  |
| **Preconditie** |  |
| **Orakel** |  |
| **Postconditie of Exception** |  |