Bug Report

Date: 19/04/2025

Reported By: Judi Wael, Marlin Ehab

Role/Position: Testers

Test File 1 (DiscountCalculatorTest):
Test Cases:

Test Case ID:	**1**
Title:	**testIsTheSpecialWeekWhenFalse()**
Description:	**Tests the function of isTheSpecialWeek() when false.**
Status:	**Passed**

Test Case ID:	**2**
Title:	**testIsTheSpecialWeekWhenTrue()**
Description:	**Tests the function of isTheSpecialWeek() when true.**
Status:	**Passed**

Test Case ID:	**3**

Title:	**testGetDiscountPercentageWhenEven()**
Description:	**Tests the function of getDiscountPercentage() when even.**
Status:	**Passed**

Test Case ID:	**4**
Title:	**testGetDiscountPercentageWhenOdd()**
Description:	**Tests the function of getDiscountPercentage() when odd.**
Status:	**Passed**

Note: No bugs were detected in this file.

Test File 2 (DiscountManagerTest):

Test Cases:

Test Case ID:	**1**
Title:	**testCalculatePriceWhenDiscountsSeasonIsFalse()**
Description:	**Tests the function of calculatePriceAfterDiscount() when false.**
Status:	**Passed**

Test Case ID:	**2**
Title:	**testCalculatePriceWhenDiscountsSeasonIsTrueAndIsSpecialWeekIsTrue()**
Description:	**Tests the function of calculatePriceAfterDiscount() and isSpecialWeek() when true.**
Status:	**Passed**

Test Case ID:	**3**

Title:testCalculatePriceWhenDiscountsSeasonIsTrueAndIsSpecialWeekIsFalseAndGetDiscountPercentageIsEven()

Description:	Tests the function of calculatePriceAfterDiscount() where isDiscountsSeason() is true and isSpecialWeek() is false and getDiscountPercentage() is even.
Status:	Failed

Test Case ID:	4

Title:testCalculatePriceWhenDiscountsSeasonIsTrueAndIsSpecialWeekIsFalseAndGetDiscountPercentageIsOdd()

Description:	Tests the function of calculatePriceAfterDiscount() where isDiscountsSeason() is true and isSpecialWeek() is false and getDiscountPercentage() is odd.
Status:	Failed

Bug Details:

Bug ID:	1

Title:	wrong calculation of price after discount when week number is even.
Severity:	2
Priority:	2
Description:	the function of getDiscountPercentage() in DiscountCalculator class returns 7 when Even while it should return 93 therefore when calculated in DiscountManagerTest() It returns 100 * 7 = 700 which does not match the expected 100 * 0.93 = 93.

Bug ID:	2
Title:	wrong calculation of price after discount when week number is odd.
Severity:	2
Priority:	2

Description:	**the function of getDiscountPercentage() in DiscountCalculator class returns 5 when odd while it should return 95 therefore when calculated in DiscountManagerTest() It returns 100 * 5 = 500 which does not match the expected 100 * 0.95 = 95.**

Test File 3 (YearTest):

Test Cases:

Test Case ID:	1
Title:	testYearDefaultCtor()
Description:	Tests the constructor of Year().
Status:	Failed.

Test Case ID:	2
Title:	testYearRangeShouldBeInvalid()
Description:	Checks the validity of the year range.
Status:	Failed.

Test Case ID:	3

Title:	**testYearRangeGreaterThan9999**()
Description:	**Checks that the year range is greater than 9999**.
Status:	**Passed.**

Test Case ID:	**4**
Title:	**testYearRangeLessThanNegative9999**()
Description:	**Checks that the year range is less than -9999.**
Status:	**Passed.**

Test Case ID:	**5**
Title:	**testYearRangeEqualTo9999**()
Description:	**Checks that the year range is equal to 9999**.
Status:	**Passed.**

Test Case ID: 6

Title: testYearTimeCtor()

Description: Creates a new Year, based on a particular instant in time.

Status: Passed.

Test Case ID: 7

Title: testYearTimeAndTimeZoneCtor()

Description: Constructs a year, based on a particular instant in time and a time zone.

Status: Passed.

Test Case ID: 8

Title: testYearTimeAndCalendarCtor()

Description: Constructs a year, based on a particular instant in time and a time zone using the calendar.

Status:	**Passed.**

Test Case ID:	**9**
Title:	**testGetYear**()
Description:	**returns the current year.**
Status:	**Passed.**

Test Case ID:	**10**
Title:	**testGetFirstMilliSecond**()
Description:	**Returns the first millisecond of the year, evaluated using the supplied calendar.**
Status:	**Passed.**

Test Case ID:	**11**

Title:	**testGetLastMilliSecond**()
Description:	**Returns the last millisecond of the year, evaluated using the supplied calendar.**
Status:	**Passed.**

Test Case ID:	**12**
Title:	**testPreviousShouldBeInvalid**()
Description:	**Returns the year preceding this one by decrementing the year by one however it should not accept the year as it is out of range.**
Status:	**Failed.**

Test Case ID:	**13**
Title:	**testPreviousIsInvalid**()

Description:	**Returns null as the year is out of range.**
Status:	**Passed.**
Test Case ID:	**14**
Title:	**testPreviousIsValid**()
Description:	**Returns the year preceding this one by decrementing the year by one.**
Status:	**Passed.**
Test Case ID:	**15**
Title:	**testNextIsInvalid**()
Description:	**Returns null as the year is out of range.**

Status:	**Passed.**
Test Case ID:	**16**
Title:	**testNextIsValid**()
Description:	**Returns the year following this one.**
Status:	**Passed.**
Test Case ID:	**17**
Title:	**testGetSerialIndex**()
Description:	**Returns a serial index number for the year.**
Status:	**Passed.**

Test Case ID:	**18**
Title:	**testGetFirstMillisecondOfCalendar**()
Description:	**it should return the first millisecond of the year but it fails as it exceeds the timeout limit(100).**
Status:	**Failed.**

Test Case ID:	**19**
Title:	**testGetLastMillisecondOfCalendar**()
Description:	**it should return the last millisecond of the year but it fails as the actual is not equal to the expected.**
Status:	**Failed.**

Test Case ID:	20
Title:	testEqualsIsTrue ()
Description:	Tests the equality of this Year object to an arbitrary object. Returns true as the target is a Year instance representing the same year as this object.
Status:	Passed.

Test Case ID:	21
Title:	testEqualsIsFalse()
Description:	Tests the equality of this Year object to an arbitrary object. Returns false as the target is a Year instance that is not equal to the same year as this object.
Status:	Passed.

Test Case ID: 22

Title: testHashCodeIsValid()

Description: Returns a hash code for this object instance.

Status: Passed.

Test Case ID: 23

Title: testCompareToWhenYearIsBefore()

Description: Returns an integer indicating the order of this Year object relative to the specified object: negative == before.

Status: Passed.

Test Case ID: 24

Title: testCompareToWhenYearIsSame ()

Description: Returns an integer indicating the order of this Year object relative to the specified object: zero == same.

Status:	Passed.
Test Case ID:	25
Title:	testCompareToWhenYearIsAfter()
Description:	Returns an integer indicating the order of this Year object relative to the specified object: positive == after.
Status:	Passed.
Test Case ID:	26
Title:	testCompareToWhenO1OfTypeInt()
Description:	Returns 1 indicating that o1 is of type integer.
Status:	Passed.
Test Case ID:	27

Title: testCompareToWhenO1OfTypeRegularTimePeriod()

Description: Returns 0 indicating that o1 is an instance of RegularTimePeriod.

Status: Passed.

Test Case ID: 28

Title: testToString()

Description: Returns a string representing the year.

Status: Passed.

Test Case ID: 29

Title: testParseYearCannotBeParsed()

Description: Parses the string argument as a year

And returns null as the string is not parseable.

Status: Passed.

Test Case ID: 30

Title: testParseYearIsValid()

Description: Parses the string argument as a year
And returns the year accordingly.

Status: Passed.

Test Case ID: 31

Title: testParseYearShouldBeInValid()

Description: Parses the string argument as a year
And returns the year however it should
Return null as the year is out of range.

Status: Failed.

Test Case ID:	32
Title:	testParseYearIsOutOfRange()
Description:	Parses the string argument as a year And returns null.
Status:	Passed.

Bug Details:

Bug ID:	1
Title:	wrong initialization of the calendar set.month.
Severity:	2
Priority:	2
Description:	the function of getLastMillisecond(Calendar calendar) sets the month to November instead of December and so will not return the right millisecond.

Bug ID:	2
Title:	wrong specification of year range interval.
Severity:	2
Priority:	2
Description:	the constructor of year sets the range to -9999 To 9999 however the specifications declare the Range as 1900 to 9999.

Bug ID:	3
Title:	wrong return of testParseYearShouldBeInValid() function.
Severity:	2
Priority:	2
Description:	returns the parsed year however it should return Null as the year is out of range.

Bug ID:	4
Title:	wrong return of testYearDefaultCtor() function.
Severity:	2
Priority:	1
Description:	returns null as the year object is null however It should return the current year.

Bug ID:	5
Title:	wrong return of testPreviousShouldBeInvalid() function.
Severity:	2
Priority:	1
Description:	returns the year object however It should return null as the year is out of range.