18654 Project - Test Plan

Semester	Team Name	Project Name
Fall 2023	T7	F23-T7-commons-lang

Capabilities Tested

Identify the capabilities tested and briefly describe them. Each capability is a small, independently tested requirement and can cover one or more closely-related methods/functions tested. Each capability ideally relies on a single, compact input space model.

#	Capability Name	Description
1	Date Value Comparison	Compare two date or calendar objects
2	Parsing Date	Parse a string representing a date
3	Date Addition	Add an amount to a date field
4	Setting Date	Set a field of the date
5	Converting Date	Convert a Date object to a Calendar object
6	Round/Truncate/Ceil Date	Round, truncate, ceil a date
7	Modify Date (Private, White-box)	Modify a date
8	Date Iterator	Constructs an iterator over each day in a date range defined by a focus date and range style. Tests include the inner class DateIterator.
9	Getting Date Fragment	Get a fragment of a Date object
10	Get fragment (Private, White-box)	Gets a Date fragment for any unit.
11	Compare 2 Dates with Specified Precision	Determines how two calendars compare up or equal to no more than the specified most significant field.

Attributes

List attributes for tested capabilities. Give the source of the attribute: where did it originate from and how did you identify it (e.g. parameter of function/method, state variable, pre-condition, etc.)?

Attribute Name	**Definition and Source**	**Related Capabilities**
Date	*java.util Date (or Calendar) object* *(source: parameter of function)*	1-11
Calendar	*java.util Calendar object* *(source: parameter of function)*	1, 6, 9, 10, 11
dateStr	*the date to parse, not null* *(source: parameter of function)*	2
parsePatterns	*The patterns used to parse a date str* *(source: parameter of function)*	2
amount	*the amount to modify* *(source: parameter of function)*	3, 4
calendarField/ fragment	*the calendar field to use* *(source: parameter of function)*	9, 10, 11; (used in private method of 3, 4)
rangeStyle	*the style constant to use* *(source: parameter of function)*	8
timezone	*the timezone to use* *(source: parameter of function)*	5

Spec-based Black Box Test

Input Space Models

Create one or more input space models based on the attributes. Define characteristics. Give the source of the characteristic: where did it originate from and how did you identify it (e.g. directly based on an attribute, output property, post-condition, input validity, etc.)?

Input Space Model Capability 1

Repeat for other input space models. If a characteristic is reused from a previous model, just enter its name in its table and don't repeat the other information related to the characteristic.

Q#	**Characteristic Name (Chr)**	**Definition (Def)**	**Source or Participating Attributes (Atr)**	
	Partition (Prt)		**Notes & Constraints (N&C)**	
Q1	**Chr:** Calendar Cal 1	**Def:** The value of the first input Calendar	**Atr:** Calendar	RC: 1
	Prt: Valid, Null		**N&C:** Calendar must be valid and cannot be null	
Q2	**Chr:** Calendar Cal 2	**Def:** The value of the second input Calendar	**Atr:** Calendar	RC: 1
	Prt: Valid, Null		**N&C:** Calendar must be valid and cannot be null	
Q3	**Chr:** Date date 1	**Def:** The value of the first input date	**Atr:** Date	RC: 1
	Prt: Valid, Null		**N&C:** Date must be valid and cannot be null	
Q4	**Chr:** Date date 2	**Def:** The value of the second input date	**Atr:** Date	RC: 1
	Prt: Valid, Null		**N&C:** Date must be valid and	

			cannot be null	
Q5	**Chr:** Equivalence	**Def:** Whether the two calendars or two dates are equivalent in the certain time units	**Atr:** Calendar, Date	RC: 1
	Prt: Equivalent-Day, Equivalent-Instant, Equivalent-Localtime, Inequivalent-Day, Inequivalent-Instant, Inequivalent-Localtime		**N&C:** no constraints	

Input Space Model Capability 2

Q#	**Characteristic Name (Chr)**	**Definition (Def)**	**Source or Participating Attributes (Atr)**	**Related Capabilities (RC)**
	Partition (Prt)		**Notes & Constraints (N&C)**	
Q1	**Chr:** Date Str	**Def:** The string representation to be parsed	**Atr:** dateStr	**RC**: 2
	Prt: Null, Empty, Normal		**N&C:** no constraints	
Q2	**Chr**: Parse Patterns	**Def:** *The patterns used to parse a date str*	**Atr:** parsePatterns	**RC**: 2
	Prt: Null, 0 Element, 1 Element, >1 Elements		**N&C:** no constraints	
Q3	**Chr**: Date Str Validity	**Def:** If the string represents a valid Date object both in format and in reality	**Atr:** dateStr	**RC:** 2
	Prt: Valid, Weekday_Invalid, Weekday_Mismatch Month_Invalid, Zone_Invalid, Invalid		**N&C:** Q1 must be Normal	
Q4	**Chr**: Parse Patterns Validity	**Def**: If the parse patterns are valid	**Atr:** parsePatterns	**RC:** 2
	Prt: All_Valid, Partially_Valid, Not_Valid		**N&C:** Q2 must be 1 or > 1 Elements; when Q2 = 1 Element, no Partially_Valid partition	
Q5	**Chr**: Match	**Def:** If the date string matches one of the parse patterns	**Atr:** dateStr, parsePatterns	**RC:** 2

	Prt: True, False		**N&C:** Q3 must be Valid, Q4 must be All Valid or Partially Valid;	
Q6	**Chr**: Leniency	**Def**: Whether or not date/time interpretation is to be lenient	**Atr**: None, test different methods	**RC:** 2
	Prt: None, Month_Lenient, Day_Lenient, Hour_Lenient, Min_Lenient, Sec_Lenient,		**N&C:** Q3 must be Valid, Q4 must be All Valid or Partially Valid	

Input Space Model Capability 3

Q#	**Characteristic Name (Chr)**	**Definition (Def)**	**Source or Participating Attributes (Atr)**	
	Partition (Prt)		**Notes & Constraints (N&C)**	
Q1	**Chr:** Date	**Def:** the date	Atr: Date	
	Prt: Null, Valid Date		**N&C: Date is either null or valid**	
Q2	**Chr:** Amount of Years	Def: the amount of years to add	Atr: Amount	
	Prt: negative and non-negative amount of year		**N&C: The Amount has to be a integer, may be negative**	
Q3	**Chr:** Amount of Months	Def: the amount of Months to add	Atr: Amount	
	Prt: negative and non-negative amount of month		**N&C: The Amount has to be a integer, may be negative**	
Q4	**Chr:** Amount of Weeks	Def: the amount of weeks to add	Atr: Amount	

	Prt: negative and non-negative amount of week		**N&C: The Amount has to be a integer, may be negative**	
Q5	**Chr:** Amount of Days	Def: the amount of days to add	Atr: Amount	
	Prt: negative and non-negative amount of Day		**N&C: The Amount has to be a integer, may be negative**	
Q6	**Chr:** Amount of Hours	Def: the amount of hours to add	Atr: Amount	
	Prt: negative and non-negative amount of hour		**N&C: The Amount has to be a integer, may be negative**	
Q7	**Chr:** Amount of Minutes	Def: the amount of Minutes to add	Atr: Amount	
	Prt: negative and non-negative amount of minutes		**N&C: The Amount has to be a integer, may be negative**	
Q8	**Chr:** Amount of Seconds	Def: the amount of seconds to add	Atr: Amount	
	Prt: negative and non-negative amount of seconds		**N&C: The Amount has to be a integer, may be negative**	
Q9	**Chr:** Amount of milliseconds	Def: the amount of milliseconds to add	Atr: Amount	
	Prt: negative and non-negative amount of milliseconds		**N&C: The Amount has to be a integer, may be negative**	

Input Space Model Capability 4

Q#	**Characteristic Name (Chr)**	**Definition (Def)**	**Source or Participating Attributes (Atr)**	**Related Capabilities (RC)**
	Partition (Prt)		**Notes & Constraints (N&C)**	

Q1	**Chr:** Date	**Def:** the date	Atr: Date	**RC**: 4
	Prt: Null, Valid Date		**N&C: Date is either Null or valid.**	
Q2	**Chr:** Amount of Years	Def: the amount of years to set	Atr: Amount	**RC**: 4
	Prt: {amount < 1}, {amount >= 1}		**N&C:** The amount has to be an integer	
Q5	**Chr:** Amount of Hours	Def: the amount of hours to set	Atr: Amount	**RC**: 4
	Prt: {amount < 0}, {0 <= amount <= 23}, {amount > 23},		**N&C:** The amount has to be an integer	
Q6	**Chr:** Amount of Minutes	Def: the amount of Minutes to set	Atr: Amount	**RC**: 4
	Prt: {amount < 0}, {0 <= amount <= 59}, {amount > 59}		**N&C:** The amount has to be an integer	
Q7	**Chr:** Amount of Seconds	Def: the amount of seconds to set	Atr: Amount	**RC**: 4
	Prt: {amount < 0}, {0 <= amount <= 59}, {amount > 59}		**N&C:** The amount has to be an integer	
Q8	**Chr:** Amount of milliseconds	Def: the amount of milliseconds to set	Atr: Amount	**RC**: 4
	Prt:{amount < 0}, {0 <= amount <= 999}, {amount > 999}		**N&C:** The amount has to be an integer	

Q#	**Characteristic Name (Chr)**	**Definition (Def)**	**Source or Participating Attributes (Atr)**	**Related Capabilities (RC)**
	Partition (Prt)		**Notes & Constraints (N&C)**	
Q1	**Chr:** Date	**Def:** the date	Atr: Date	**RC**: 4

	Prt: Null, Valid Date with Day = 31st, Valid Non-leap year Date with Day = 29th, Valid Leap Year Date with Day = 29th, All other valid date		**N&C: Date is either Null or valid.**	
Q3	**Chr:** Amount of Months	Def: the amount of Months to set	Atr: Amount	**RC**: 4
	Prt:{amount < 0}, { amount = 1 (Feb)}, {amount = 3, 5, 8, 10}, {amount = 0, 2, 4, 6, 7, 9, 11}, {amount > 11}		**N&C:** The amount has to be an integer	

Q#	**Characteristic Name (Chr)**	**Definition (Def)**	**Source or Participating Attributes (Atr)**	**Related Capabilities (RC)**
	Partition (Prt)		**Notes & Constraints (N&C)**	
Q1	**Chr:** Date	**Def:** the date	Atr: Date	**RC**: 4
	Prt: Null, Date of Non-leap Year in Feb, Date of Leap Year in Feb, Date in 30-day Month, Date in 31-day Month		**N&C: Date is either Null or valid.**	
Q4	**Chr:** Amount of Days	Def: the amount of days to set	Atr: Amount	**RC**: 4
	Prt: {amount < 1},{1 <= amount <= 28 },{amount = 29}, {amount = 30},{amount = 31}, {amount >31}		**N&C:** The amount has to be an integer	

Input Space Model Capability 5

Repeat for other input space models. If a characteristic is reused from a previous model, just enter its name in its table and don't repeat the other information related to the characteristic.

Q#	**Characteristic Name (Chr)**	**Definition (Def)**	**Source or Participating Attributes (Atr)**	
	Partition (Prt)		**Notes & Constraints (N&C)**	
Q1	**Chr:** Date date	The Date	**Atr:** Date	RC: 1,5

	Prt: Valid, Null		**N&C:** None.	
Q2	**Chr:** Timezone	**Def:** The timezone to use	**Atr:** timezone	**RC**: 5
	Prt: Valid, Null		**N&C:** None	

Input Space Model Capability 6

Q#	**Characteristic Name (Chr)**	**Definition (Def)**	**Source or Participating Attributes (Atr)**	**Related Capabilities (RC)**
	Partition (Prt)		**Notes & Constraints (N&C)**	
Q1	**Chr:** Date	**Def:** The Date to work with	**Atr:** date	**RC**: 6
	Prt: Date that rounds down, Date that rounds up but does not change higher field(s), Date that rounds up and changes higher field(s), Date that does not change after rounding, Null		**N&C:** Must be valid (if the type is Date or Calendar); partition based on corresponding Field	
Q2	**Chr:** Field	**Def:** The field to work with	**Atr:** field	**RC**: 6
	Prt: All fields in DateUtils.fields, Fields not in DateUtils.fields, Null		**N&C:** None	
Q3	**Chr:** Date Type	**Def:** The type of Date object	**Atr:** date	**RC**: 6
	Prt: Null, Date, Calendar, other type		**N&C:** None	
Q4	**Chr:** Field Validity	**Def:** Whether Field is valid	**Atr:** field	**RC**: 6
	Prt: Yes, No		**N&C:** None	

Input Space Model Capability 8

Q#	**Characteristic Name (Chr)**	**Definition (Def)**	**Source or Participating Attributes (Atr)**	**Related Capabilities (RC)**
	Partition (Prt)		**Notes & Constraints (N&C)**	

Q1	**Chr:** Date	**Def:** The date used to make an iterator.	**Atr:** Date	**RC**: 8
	Prt: All valid Date		**N&C: Cannot be null**	
Q2	**Chr:** Calendar	**Def:** The calendar used to make an iterator.	**Atr:** Calendar	**RC**: 8
	Prt: All valid Calendar		**N&C: Cannot be null**	
Q3	**Chr:** CalendarObj	**Def:** The calendar object used to make an iterator.	**Atr:** Calendar	**RC**: 8
	Prt: Null, ValidObj, InvalidObj		**N&C: No constraints**	
Q4	**Chr:** RangeStyle	**Def:** The style constant used to make an iterator.	**Atr:** RangeStyle	**RC**: 8
	Prt: RANGE_MONTH_SUNDAY, RANGE_MONTH_MONDAY, RANGE_WEEK_SUNDAY, RANGE_WEEK_MONDAY, RANGE_WEEK_RELATIVE, RANGE_WEEK_CENTER, Null		**N&C: No constraints**	

Input Space Model Capability 9

Q#	**Characteristic Name (Chr)**	**Definition (Def)**	**Source or Participating Attributes (Atr)**	**Related Capabilities (RC)**
	Partition (Prt)		**Notes & Constraints (N&C)**	
Q1	**Chr:** Date	**Def:** The date.	**Atr:** Date	**RC**: 9
	Prt: Null, All valid Date		**N&C: No constraints**	
Q2	**Chr:** Calendar	**Def:** The calendar.	**Atr:** Calendar	**RC**: 9
	Prt: Null, All valid Calendar		**N&C: No constraints**	
Q3	**Chr:** fragment	**Def:** the Calendar field part of date to calculate	**Atr:** calendarField/ fragment	**RC**: 9
	Prt: MILLISECOND, SECOND,MINUTE,		**N&C: No constraints**	

	HOUR_OF_DAY, DATE, DAY_OF_YEAR, MONTH, YEAR, Invalid	

Input Space Model Capability 11

Repeat for other input space models. If a characteristic is reused from a previous model, just enter its name in its table and don’t repeat the other information related to the characteristic.

Q#	**Characteristic Name (Chr)**	**Definition (Def)**	**Source or Participating Attributes (Atr)**	
	Partition (Prt)		**Notes & Constraints (N&C)**	
Q1	**Chr:** Calendar Cal 1	The calendar	**Atr:** Calendar	RC: 1, 11
	Prt: Null, All valid Calendar		**N&C: No constraints**	
Q2	**Chr:** Calendar Cal 2	The second calendar to be compared with.	**Atr:** Calendar	RC: 1, 11
	Prt: Null, All valid Calendar		**N&C: No constraints**	
Q3	**Chr:** Date date 1	The date	**Atr:** Date	RC: 1, 11
	Prt: Null, All valid Dates		**N&C: No constraints**	
Q4	**Chr:** Date date 2	The second date to be compared with	**Atr:** Date	RC: 1, 11
	Prt: Null, All valid Dates		**N&C: No constraints**	
Q5	**Chr:** Calendar Field	**Def:** The calendar field to use	**Atr:** calendarField	RC: 11
	Prt: Era, Year, Month, Date, Hour, Minute, Millisecond, HOUR_OF_DAY, DAY_OF_WEEK, DAY_OF_WEEK_IN_MONTH,		**N&C:** no constraints	

	DAY_OF_MONTH, DAY_OF_YEAR, WEEK_OF_YEAR, WEEK_OF_MONTH AM_PM, SEMI_MONTH, Fields not in DateUtils.fields			
Q6	**Chr:** Field Validity	**Def:** Whether the calendar field is valid or not	**Atr:** calendarField	RC: 11
	Prt: Valid, Invalid		**N&C:** no constraints	
Q7	**Chr:** Equivalence	**Def:** Whether the two calendars or two dates are equivalent in the after truncated in certain time units	**Atr:** Calendar, Date	RC: 11
	Prt: T,F		**N&C:** no constraints	

Test Design Strategy

Step 1A - Test Case Specs (Capability 1)

Cases for testing the method DateUtils.IsSameDay(Date, Date), IsSameInstant(Date, Date). Input space model is included in Input Space Model Capability 1. All Choices were applied.			
Case Id	**Date 1**	**Date 2**	**Equivalence**
1A1	*Valid*	*Valid*	Equivalent-Day
1A2	*Valid*	*Valid*	Inequivalent-Day
1A3	*Valid*	*Valid*	Equivalent-Instant
1A4	*Valid*	*Valid*	Inequivalent-Instant
1A5	*Valid*	*Null*	-
1A6	*Null*	*Valid*	-
1A7	*Null*	*Null*	-

Step 1B - Test Case Specs (Capability 1)

Cases for testing the method DateUtils.IsSameDay(Calendar, Calendar), IsSameInstant(Calendar, Calendar). IsSameLocalTime(Calendar, Calendar). Input space model is included in Input Space Model Capability 1. All Choices were applied.			
Case Id	**Cal 1**	**Cal 2**	**Equivalence**
1B1	*Valid*	*Valid*	Equivalent-Day
1B2	*Valid*	*Valid*	Inequivalent-Day
1B3	*Valid*	*Valid*	Equivalent-Instant
1B4	*Valid*	*Valid*	Inequivalent-Instant
1B5	*Valid*	*Valid*	Equivalent-Localtime
1B6	*Valid*	*Valid*	Inequivalent-Localtime
1B7	*Null*	*Valid*	-
1B8	*Null*	*Valid*	-
1B9	*Null*	*Null*	-

Step 2A - Test Case Specs

Base Choice combinations of Date Str, Parse Patterns, Date Str Validity, Parse Patterns Validity, Match, and Leniency. Two Base Choices were applied.							
Case Id	**Date Str**	**Parse Patterns**	**Date Str Validity**	**Parse Patterns Validity**	**Match**	**Leniency**	**Notes**
2A1	*Normal*	*1 Element*	*Valid*	*All_Valid*	*T*	*None*	Base Choice 1
2A2	*Normal*	*>1 Elements*	*Valid*	*All_Valid*	*T*	*None*	Base Choice 2
2A3	*Null*	*1 Element*	**-**	*All_Valid*	**-**	**-**	

2A4	*Empty*	*1 Element*	-	*All_Valid*	-	-	
2A5	*Null*	*>1 Elements*	-	*All_Valid*	-	-	
2A6	*Empty*	*>1 Elements*	-	*All_Valid*	-	-	
2A7	*Normal*	*Null*	*Valid*	-	-	-	
2A8	*Normal*	*0 Element*	*Valid*	-	-	-	
2A9	*Normal*	*1 Element*	*Weekday_ Invalid*	*All_Valid*	-	-	
2A10	*Normal*	*>1 Elements*	*Weekday_ Invalid*	*All_Valid*	-	-	
2A11	*Normal*	*1 Element*	Weekday_ Mismatch	*All_Valid*	-	-	
2A12	*Normal*	*>1 Elements*	Weekday_ Mismatch	*All_Valid*	-	-	
2A13	*Normal*	*1 Element*	*Month_Inv alid*	*All_Valid*	-	-	
2A14	*Normal*	*>1 Elements*	*Month_Inv alid*	*All_Valid*	-	-	
2A15	*Normal*	*1 Elements*	Zone_Inv alid	*All_Valid*	-	-	
2A16	*Normal*	*>1 Elements*	Zone_Inv alid	*All_Valid*	-	-	
2A17	*Normal*	*1 Elements*	Invalid	*All_Valid*	-	-	
2A18	*Normal*	*>1 Elements*	Invalid	*All_Valid*	-	-	
2A19	*Normal*	*>1 Elements*	*Valid*	*Partially_V alid*	*T*	*None*	
2A20	*Normal*	*1 Element*	*Valid*	*Not_Valid*	-	-	
2A21	*Normal*	*>1 Elements*	*Valid*	*Not_Valid*	-	-	
2A22	*Normal*	*1 Element*	*Valid*	*All_Valid*	*F*	*None*	
2A23	*Normal*	*>1 Elements*	*Valid*	*All_Valid*	*F*	*None*	
2A24	*Normal*	*1 Element*	*Valid*	*All_Valid*	*T*	*Month_Li*	

						nent	
2A25	*Normal*	*>1 Elements*	*Valid*	*All_Valid*	*T*	*Month_Linent*	
2A26	*Normal*	*1 Element*	*Valid*	*All_Valid*	*T*	*Day_Linent*	
2A27	*Normal*	*>1 Elements*	*Valid*	*All_Valid*	*T*	*Day_Linent*	
2A28	*Normal*	*1 Element*	*Valid*	*All_Valid*	*T*	*Hour_Linent*	
2A29	*Normal*	*>1 Elements*	*Valid*	*All_Valid*	*T*	*Hour_Linent*	
2A30	*Normal*	*1 Element*	*Valid*	*All_Valid*	*T*	*Min_Linent*	
2A31	*Normal*	*>1 Elements*	*Valid*	*All_Valid*	*T*	*Min_Linent*	
2A32	*Normal*	*1 Element*	*Valid*	*All_Valid*	*T*	*Sec_Linent*	
2A33	*Normal*	*>1 Elements*	*Valid*	*All_Valid*	*T*	*Sec_Linent*	

Step 3A - Test Case Specs (Capability 3)

Cases for testing the method DateUtils.addYears(). *Input space model is included in Input Space Model Capability 3.*		
Case Id	**Date**	**Amount of Years**
3A1	*Valid Date*	*Non-negative*
3A2	*Valid Date*	*Negative*
3A3	*Null*	*Non-negative or Negative*

Step 3B - Test Case Specs (Capability 3)

Cases for testing the method DateUtils.addMonths(). *Input space model is included in Input Space Model Capability 3.*

Case Id	**Date**	**Amount of Months**
3B1	*Valid Date*	*Non-negative*
3B2	*Valid Date*	*Negative*
3B3	*Null*	*Non-negative or Negative*

Step 3C - Test Case Specs (Capability 3)

<table>
<tr><td colspan="3"><i>Cases for testing the method DateUtils.addWeeks().</i>
<i>Input space model is included in Input Space Model Capability 3.</i></td></tr>
<tr><th>Case Id</th><th>Date</th><th>Amount of Weeks</th></tr>
<tr><td>3C1</td><td><i>Valid Date</i></td><td><i>Non-negative</i></td></tr>
<tr><td>3C2</td><td><i>Valid Date</i></td><td><i>Negative</i></td></tr>
<tr><td>3C3</td><td><i>Null</i></td><td><i>Non-negative or Negative</i></td></tr>
</table>

Step 3D - Test Case Specs (Capability 3)

<table>
<tr><td colspan="3"><i>Cases for testing the method DateUtils.addDays().</i>
<i>Input space model is included in Input Space Model Capability 3.</i></td></tr>
<tr><th>Case Id</th><th>Date</th><th>Amount of Days</th></tr>
<tr><td>3D1</td><td><i>Valid Date</i></td><td><i>Non-negative</i></td></tr>
<tr><td>3D2</td><td><i>Valid Date</i></td><td><i>Negative</i></td></tr>
<tr><td>3D3</td><td><i>Null</i></td><td><i>Non-negative or Negative</i></td></tr>
</table>

Step 3E - Test Case Specs (Capability 3)

<table>
<tr><td colspan="3"><i>Cases for testing the method DateUtils.addHours().</i>
<i>Input space model is included in Input Space Model Capability 3.</i></td></tr>
<tr><th>Case Id</th><th>Date</th><th>Amount of Hours</th></tr>
<tr><td>3E1</td><td><i>Valid Date</i></td><td><i>Non-negative</i></td></tr>
<tr><td>3E2</td><td><i>Valid Date</i></td><td><i>Negative</i></td></tr>
<tr><td>3E3</td><td><i>Null</i></td><td><i>Non-negative or Negative</i></td></tr>
</table>

Step 3F - Test Case Specs (Capability 3)

Cases for testing the method DateUtils.addMinutes(). *Input space model is included in Input Space Model Capability 3.*		
Case Id	**Date**	**Amount of Minutes**
3F1	*Valid Date*	*Non-negative*
3F2	*Valid Date*	*Negative*
3F3	*Null*	*Non-negative or Negative*

Step 3G - Test Case Specs (Capability 3)

Cases for testing the method DateUtils.addSeconds(). *Input space model is included in Input Space Model Capability 3.*		
Case Id	**Date**	**Amount of Seconds**
3G1	*Valid Date*	*Non-negative*
3G2	*Valid Date*	*Negative*
3G3	*Null*	*Non-negative or Negative*

Step 3H - Test Case Specs (Capability 3)

Cases for testing the method DateUtils.addMilliseconds(). *Input space model is included in Input Space Model Capability 3.*		
Case Id	**Date**	**Amount of Milliseconds**
3H1	*Valid Date*	*Non-negative*
3H2	*Valid Date*	*Negative*
3H3	*Null*	*Non-negative or Negative*

Step 4A - Test Case Specs (Capability 4)

Cases for testing the method DateUtils.setYears(). *Input space model is included in Input Space Model Capability 4.*		
Case Id	**Month of Date**	**Amount of Years**
4A1	*Valid Date*	amount >= 1

4A2	*Valid Date*	amount < 1
4A3	*Null*	amount >= 1 or amount < 1

Step 4B - Test Case Specs (Capability 4)

Cases for testing the method DateUtils.setMonths().
Input space model is included in Input Space Model Capability 4.

Case Id	**Date**	**Amount of Months**
4B1	*Valid Date*	0 <= amount <=11
4B2	*Valid Date*	amount < 0
4B3	*Valid Date*	amount > 11
4B4	*Null*	0 <= amount <=11 or amount < 0 or amount > 11
4B5	*Date with Day larger than or equal to 29th*	amount = 1 (Feb)
4B6	*Date with Day = 31st*	amount = 3, 5, 8, 10 (Apr, Jun, Sep, Nov)

Step 4C - Test Case Specs (Capability 4)

Cases for testing the method DateUtils.setDays().
Input space model is included in Input Space Model Capability 4.

Case Id	**Date**	**Amount of Days**
4C1	*Valid Date*	1 <= amount <= 31
4C2	*Valid Date*	amount < 1
4C3	*Valid Date*	amount > 31
4C4	*Null*	1 <= amount <= 31 or amount < 1 or amount > 31
4C5	*Date with Non-leap Year at Feb.28*	Amount > 28
4C6	*Date with leap Year at Feb.29*	Amount > 29

4C7	*Date with 30-day Month*	Amount > 30

Step 4D - Test Case Specs (Capability 4)

Cases for testing the method DateUtils.setHours(). Input space model is included in Input Space Model Capability 4.		
Case Id	**Date**	**Amount of Hours**
4D1	*Valid Date*	0 <= amount <=23
4D2	*Valid Date*	amount < 0
4D2	*Valid Date*	amount > 23
4D3	*Null*	0 <= amount <=23 or amount < 0 or amount > 23

Step 4E - Test Case Specs (Capability 4)

Cases for testing the method DateUtils.setMinutes(). Input space model is included in Input Space Model Capability 4.		
Case Id	**Date**	**Amount of Minutes**
4E1	*Valid Date*	{0 <= amount <=59}
4E2	*Valid Date*	amount < 0
4E3	*Valid Date*	amount > 59
4E4	*Null*	{0 <= amount <=59} or amount < 0 or amount > 59

Step 4F - Test Case Specs (Capability 4)

Cases for testing the method DateUtils.setSeconds(). Input space model is included in Input Space Model Capability 4.		
Case Id	**Date**	**Amount of Seconds**
4F1	*Valid Date*	{0 <= amount <=59}
4F2	*Valid Date*	amount < 0
4F3	*Valid Date*	amount > 59
4F4	*Null*	{0 <= amount <=59} or amount < 0 or amount > 59

Step 4G - Test Case Specs (Capability 4)

Cases for testing the method DateUtils.setMilliseconds().
Input space model is included in Input Space Model Capability 4.

Case Id	Date	Amount of MilliSeconds
4G1	*Valid Date*	0 <= amount <=999
4G2	*Valid Date*	amount < 0
4G3	*Valid Date*	amount > 999
4G4	*Null*	0 <= amount <=999 or amount < 0 or amount > 999

Step 5A - Test Case Specs (Capability 5)

Cases for testing the method DateUtils.toCalendar(Date), DateUtils.toCalendar(Date, Timezone)
Input space model is included in Input Space Model Capability 5. All Choices were applied.

Case Id	Date	Timezone
5A1	*Valid*	-
5A2	*Invalid*	-
5A3	*Valid*	*Valid*
5A4	*Invalid*	*Valid*
5A5	*Valid*	*Null*
5A6	*Invalid*	*Null*

Step 6A - Test Case Specs (Capability 6)

Cases for testing the method DateUtils.round().
Input space model is included in Input Space Model Capability 6.

Case Id	Date	Field	Date Type	Field Validity
6A1	Date that rounds down	Calendar.MINUTE	Calendar	Yes

6A2	Date that rounds up but does not change higher field(s)	Calendar.MINUTE	Calendar	Yes
6A3	Date that rounds up and changes higher field(s)	Calendar.MINUTE	Calendar	Yes
6A4	Date that does not change after rounding	Calendar.MINUTE	Calendar	Yes
6A5	Date that rounds down	Calendar.MILLISECOND	Calendar	Yes
6A6	Date that rounds down	Calendar.SECOND	Calendar	Yes
6A7	Date that rounds down	Calendar.HOUR_OF_DAY	Calendar	Yes
6A8	Date that rounds down	Calendar.HOUR	Calendar	Yes
6A9	Date that rounds down	Calendar.DATE	Calendar	Yes
6A10	Date that rounds down	Calendar.DAY_OF_MONTH	Calendar	Yes
6A11	Date that rounds down	Calendar.AM_PM	Calendar	Yes
6A12	Date that rounds down	Calendar.MONTH	Calendar	Yes
6A13	Date that rounds down	DateUtils.SEMI_MONTH	Calendar	Yes
6A14	Date that rounds down	Calendar.YEAR	Calendar	Yes
6A15	Date that rounds down	Calendar.ERA	Calendar	Yes
6A16	Date that rounds down	Invalid Field	Calendar	No
6A17	Date that rounds down	Calendar.MINUTE	Date	Yes
6A	Null	Calendar.MINUTE	Null	Yes

18				
6A19	Date that rounds down	Calendar.MINUTE	Other type	Yes

Step 6B - Test Case Specs (Capability 6)

Cases for testing the method DateUtils.truncate().
Input space model is included in Input Space Model Capability 6.

Case Id	**Date**	**Field**	**Date Type**	**Field Validity**
6B1	Date that rounds down	Calendar.MINUTE	Calendar	Yes
6B2	Date that does not change after rounding	Calendar.MINUTE	Calendar	Yes
6B3	Date that rounds down	Calendar.MILLISECOND	Calendar	Yes
6B4	Date that rounds down	Calendar.SECOND	Calendar	Yes
6B5	Date that rounds down	Calendar.HOUR_OF_DAY	Calendar	Yes
6B6	Date that rounds down	Calendar.HOUR	Calendar	Yes
6B7	Date that rounds down	Calendar.DATE	Calendar	Yes
6B8	Date that rounds down	Calendar.DAY_OF_MONTH	Calendar	Yes
6B9	Date that rounds down	Calendar.AM_PM	Calendar	Yes
6B10	Date that rounds down	Calendar.MONTH	Calendar	Yes
6B11	Date that rounds down	DateUtils.SEMI_MONTH	Calendar	Yes
6B	Date that rounds down	Calendar.YEAR	Calendar	Yes

12				
6B 13	Date that rounds down	Calendar.ERA	Calendar	Yes
6B 14	Date that rounds down	Invalid Field	Calendar	No
6B 15	Date that rounds down	Calendar.MINUTE	Date	Yes
6B 16	Null	Calendar.MINUTE	Null	Yes
6B 17	Date that rounds down	Calendar.MINUTE	Other type	Yes

Step 6C - Test Case Specs (Capability 6)

Cases for testing the method DateUtils.ceiling(). *Input space model is included in Input Space Model Capability 6.*				
Case Id	**Date**	**Field**	**Date Type**	**Field Validity**
6C 1	Date that rounds up but does not change higher field(s)	Calendar.MINUTE	Calendar	Yes
6C 2	Date that rounds up and changes higher field(s)	Calendar.MINUTE	Calendar	Yes
6C 3	Date that rounds up but does not change higher field(s)	Calendar.MILLISECOND	Calendar	Yes
6C 4	Date that rounds up but does not change higher field(s)	Calendar.SECOND	Calendar	Yes
6C 5	Date that rounds up but does not change higher field(s)	Calendar.HOUR_OF_DAY	Calendar	Yes
6C 6	Date that rounds up but does not change higher field(s)	Calendar.HOUR	Calendar	Yes

6C7	Date that rounds up but does not change higher field(s)	Calendar.DATE	Calendar	Yes
6C8	Date that rounds up but does not change higher field(s)	Calendar.DAY_OF_MONTH	Calendar	Yes
6C9	Date that rounds up but does not change higher field(s)	Calendar.AM_PM	Calendar	Yes
6C10	Date that rounds up but does not change higher field(s)	Calendar.MONTH	Calendar	Yes
6C11	Date that rounds up but does not change higher field(s)	DateUtils.SEMI_MONTH	Calendar	Yes
6C12	Date that rounds up but does not change higher field(s)	Calendar.YEAR	Calendar	Yes
6C13	Date that rounds up but does not change higher field(s)	Calendar.ERA	Calendar	Yes
6C14	Date that rounds up but does not change higher field(s)	Invalid Field	Calendar	No
6C15	Date that rounds up but does not change higher field(s)	Calendar.MINUTE	Date	Yes
6C16	Null	Calendar.MINUTE	Null	Yes
6C17	Date that rounds up but does not change higher field(s)	Calendar.MINUTE	Other type	Yes

Step 8A - Test Case Specs (Capability 8)

Cases for testing the method DateUtils.iterator(final Date focus, final int rangeStyle). Input space model is included in Input Space Model Capability 8.		
Case Id	**CalendarObj**	**RangeStyle**
8A1	Null	RANGE_MONTH_SUNDAY
8A2	Null	RANGE_MONTH_MONDAY
8A3	Null	RANGE_WEEK_SUNDAY
8A4	Null	RANGE_WEEK_MONDAY
8A5	Null	RANGE_WEEK_RELATIVE
8A6	Null	RANGE_WEEK_CENTER
8A7	Null	Invalid
8A8	ValidObj	RANGE_MONTH_SUNDAY
8A9	ValidObj	RANGE_MONTH_MONDAY
8A10	ValidObj	RANGE_WEEK_SUNDAY
8A11	ValidObj	RANGE_WEEK_MONDAY
8A12	ValidObj	RANGE_WEEK_RELATIVE
8A13	ValidObj	RANGE_WEEK_CENTER
8A14	ValidObj	Invalid
8A15	InvalidObj	RANGE_MONTH_SUNDAY
8A16	InvalidObj	RANGE_MONTH_MONDAY
8A17	InvalidObj	RANGE_WEEK_SUNDAY
8A18	InvalidObj	RANGE_WEEK_MONDAY
8A19	InvalidObj	RANGE_WEEK_RELATIVE
8A20	InvalidObj	RANGE_WEEK_CENTER
8A21	InvalidObj	Null

Step 8B - Test Case Specs (Capability 8)

<table>
<tr><td colspan="3"><i>Cases for testing the method DateUtils.iterator(final Object calendar, final int rangeStyle).
Input space model is included in Input Space Model Capability 8.</i></td></tr>
<tr><th>Case Id</th><th>Calendar</th><th>RangeStyle</th></tr>
<tr><td>8B1</td><td>Valid Calendar</td><td>RANGE_MONTH_SUNDAY</td></tr>
<tr><td>8B2</td><td>Valid Calendar</td><td>RANGE_MONTH_MONDAY</td></tr>
<tr><td>8B3</td><td>Valid Calendar</td><td>RANGE_WEEK_SUNDAY</td></tr>
<tr><td>8B4</td><td>Valid Calendar</td><td>RANGE_WEEK_MONDAY</td></tr>
<tr><td>8B5</td><td>Valid Calendar</td><td>RANGE_WEEK_RELATIVE</td></tr>
<tr><td>8B6</td><td>Valid Calendar</td><td>RANGE_WEEK_CENTER</td></tr>
<tr><td>8B7</td><td>Valid Calendar</td><td>Null</td></tr>
</table>

Step 8C - Test Case Specs (Capability 8)

<table>
<tr><td colspan="3"><i>Cases for testing the method DateUtils.iterator(final Object calendar, final int rangeStyle).
Input space model is included in Input Space Model Capability 8.</i></td></tr>
<tr><th>Case Id</th><th>Date</th><th>RangeStyle</th></tr>
<tr><td>8C1</td><td>Valid Date</td><td>RANGE_MONTH_SUNDAY</td></tr>
<tr><td>8C2</td><td>Valid Date</td><td>RANGE_MONTH_MONDAY</td></tr>
<tr><td>8C3</td><td>Valid Date</td><td>RANGE_WEEK_SUNDAY</td></tr>
<tr><td>8C4</td><td>Valid Date</td><td>RANGE_WEEK_MONDAY</td></tr>
<tr><td>8C5</td><td>Valid Date</td><td>RANGE_WEEK_RELATIVE</td></tr>
<tr><td>8C6</td><td>Valid Date</td><td>RANGE_WEEK_CENTER</td></tr>
<tr><td>8C7</td><td>Valid Date</td><td>Null</td></tr>
</table>

Step 9A - Test Case Specs (Capability 9)

Cases for testing the method DateUtils.getFragmentInMilliseconds(final Date date, final int fragment). *Input space model is included in Input Space Model Capability 9.*		
Case Id	**Date**	**fragment**
9A1	Valid Date	MILLISECOND
9A2	Valid Date	SECOND
9A3	Valid Date	MINUTE
9A4	Valid Date	HOUR_OF_DAY
9A5	Valid Date	DATE
9A6	Valid Date	DAY_OF_YEAR
9A7	Valid Date	MONTH
9A8	Valid Date	YEAR
9A9	Valid Date	Invalid
9A10	Null	MILLISECOND

Step 9B - Test Case Specs (Capability 9)

Cases for testing the method DateUtils.getFragmentInSeconds(final Date date, final int fragment). *Input space model is included in Input Space Model Capability 9.*		
Case Id	**Date**	**fragment**
9B1	Valid Date	MILLISECOND
9B2	Valid Date	SECOND
9B3	Valid Date	MINUTE
9B4	Valid Date	HOUR_OF_DAY
9B5	Valid Date	DATE

9B6	Valid Date	DAY_OF_YEAR
9B7	Valid Date	MONTH
9B8	Valid Date	YEAR
9B9	Valid Date	Invalid
9B10	Null	MILLISECOND

Step 9C - Test Case Specs (Capability 9)

Cases for testing the method DateUtils.getFragmentInMinutes(final Date date, final int fragment).
Input space model is included in Input Space Model Capability 9.

Case Id	**Date**	**fragment**
9C1	Valid Date	MILLISECOND
9C2	Valid Date	SECOND
9C3	Valid Date	MINUTE
9C4	Valid Date	HOUR_OF_DAY
9C5	Valid Date	DATE
9C6	Valid Date	DAY_OF_YEAR
9C7	Valid Date	MONTH
9C8	Valid Date	YEAR
9C9	Valid Date	Invalid
9C10	Null	MILLISECOND

Step 9D - Test Case Specs (Capability 9)

Cases for testing the method DateUtils.getFragmentInHours(final Date date, final int fragment).
Input space model is included in Input Space Model Capability 9.

Case Id	Date	fragment
9D1	Valid Date	MILLISECOND
9D2	Valid Date	SECOND
9D3	Valid Date	MINUTE
9D4	Valid Date	HOUR_OF_DAY
9D5	Valid Date	DATE
9D6	Valid Date	DAY_OF_YEAR
9D7	Valid Date	MONTH
9D8	Valid Date	YEAR
9D9	Valid Date	Invalid
9D10	Null	MILLISECOND

Step 9E - Test Case Specs (Capability 9)

Cases for testing the method DateUtils.getFragmentInDays(final Date date, final int fragment). *Input space model is included in Input Space Model Capability 9.*		
Case Id	**Date**	**fragment**
9E1	Valid Date	MILLISECOND
9E2	Valid Date	SECOND
9E3	Valid Date	MINUTE
9E4	Valid Date	HOUR_OF_DAY
9E5	Valid Date	DATE
9E6	Valid Date	DAY_OF_YEAR
9E7	Valid Date	MONTH
9E8	Valid Date	YEAR
9E9	Valid Date	Invalid

9E10	Null	MILLISECOND

Step 9F - Test Case Specs (Capability 9)

<table>
<tr><td colspan="3"><i>Cases for testing the method DateUtils.getFragmentInMilliseconds(final Calendar calendar, final int fragment).
Input space model is included in Input Space Model Capability 9.</i></td></tr>
<tr><td><b>Case Id</b></td><td><b>Calendar</b></td><td><b>fragment</b></td></tr>
<tr><td>9F1</td><td>Valid Calendar</td><td>MILLISECOND</td></tr>
<tr><td>9F2</td><td>Valid Calendar</td><td>SECOND</td></tr>
<tr><td>9F3</td><td>Valid Calendar</td><td>MINUTE</td></tr>
<tr><td>9F4</td><td>Valid Calendar</td><td>HOUR_OF_DAY</td></tr>
<tr><td>9F5</td><td>Valid Calendar</td><td>DATE</td></tr>
<tr><td>9F6</td><td>Valid Calendar</td><td>DAY_OF_YEAR</td></tr>
<tr><td>9F7</td><td>Valid Calendar</td><td>MONTH</td></tr>
<tr><td>9F8</td><td>Valid Calendar</td><td>YEAR</td></tr>
<tr><td>9F9</td><td>Valid Calendar</td><td>Invalid</td></tr>
<tr><td>9F11</td><td>Null</td><td>MILLISECOND</td></tr>
</table>

Step 9G - Test Case Specs (Capability 9)

<table>
<tr><td colspan="3"><i>Cases for testing the method DateUtils.getFragmentInSeconds(final Calendar calendar, final int fragment).
Input space model is included in Input Space Model Capability 9.</i></td></tr>
<tr><td><b>Case Id</b></td><td><b>Calendar</b></td><td><b>fragment</b></td></tr>
<tr><td>9G1</td><td>Valid Calendar</td><td>MILLISECOND</td></tr>
<tr><td>9G2</td><td>Valid Calendar</td><td>SECOND</td></tr>
<tr><td>9G3</td><td>Valid Calendar</td><td>MINUTE</td></tr>
<tr><td>9G4</td><td>Valid Calendar</td><td>HOUR_OF_DAY</td></tr>
</table>

9G5	Valid Calendar	DATE
9G6	Valid Calendar	DAY_OF_YEAR
9G7	Valid Calendar	MONTH
9G8	Valid Calendar	YEAR
9G9	Valid Calendar	Invalid
9G11	Null	MILLISECOND

Step 9H - Test Case Specs (Capability 9)

Cases for testing the method DateUtils.getFragmentInMinutes(final Calendar calendar, final int fragment).
Input space model is included in Input Space Model Capability 9.

Case Id	**Calendar**	**fragment**
9H1	Valid Calendar	MILLISECOND
9H2	Valid Calendar	SECOND
9H3	Valid Calendar	MINUTE
9H4	Valid Calendar	HOUR_OF_DAY
9H5	Valid Calendar	DATE
9H6	Valid Calendar	DAY_OF_YEAR
9H7	Valid Calendar	MONTH
9H8	Valid Calendar	YEAR
9H9	Valid Calendar	Invalid
9H10	Null	MILLISECOND

Step 9I - Test Case Specs (Capability 9)

Cases for testing the method DateUtils.getFragmentInHours(final Calendar calendar, final int fragment).
Input space model is included in Input Space Model Capability 9.

Case Id	Calendar	fragment
9I1	Valid Calendar	MILLISECOND
9I2	Valid Calendar	SECOND
9I3	Valid Calendar	MINUTE
9I4	Valid Calendar	HOUR_OF_DAY
9I5	Valid Calendar	DATE
9I6	Valid Calendar	DAY_OF_YEAR
9I7	Valid Calendar	MONTH
9I8	Valid Calendar	YEAR
9I9	Valid Calendar	Invalid
9I10	Null	MILLISECOND

Step 9J - Test Case Specs (Capability 9)

<table>
<tr><td colspan="3"><i>Cases for testing the method DateUtils.getFragmentInDays(final Calendar calendar, final int fragment).
Input space model is included in Input Space Model Capability 9.</i></td></tr>
<tr><th>Case Id</th><th>Calendar</th><th>fragment</th></tr>
<tr><td>9J1</td><td>Valid Calendar</td><td>MILLISECOND</td></tr>
<tr><td>9J2</td><td>Valid Calendar</td><td>SECOND</td></tr>
<tr><td>9J3</td><td>Valid Calendar</td><td>MINUTE</td></tr>
<tr><td>9J4</td><td>Valid Calendar</td><td>HOUR_OF_DAY</td></tr>
<tr><td>9J5</td><td>Valid Calendar</td><td>DATE</td></tr>
<tr><td>9J6</td><td>Valid Calendar</td><td>DAY_OF_YEAR</td></tr>
<tr><td>9J7</td><td>Valid Calendar</td><td>MONTH</td></tr>
<tr><td>9J8</td><td>Valid Calendar</td><td>YEAR</td></tr>
<tr><td>9J9</td><td>Valid Calendar</td><td>Invalid</td></tr>
</table>

9J10	Null	MILLISECOND

Step 11A - Test Case Specs (Capability 11)

Base Choice combinations of Calendar 1, Calendar 2, Calendar Field, Field Validity, and Equivalence. Two Base Choices were applied.

Case Id	**Calendar 1**	**Calendar 2**	**Calendar Field**	**Field Validity**	**Equivalence**	**Notes**
11A1	*Valid*	*Valid*	Calendar.MINUTE	*Valid*	*T*	Base Choice 1
11A2	*Valid*	*Valid*	Calendar.MINUTE	*Valid*	*F*	Base Choice 2
11A3	*Null*	*Valid*	Calendar.MINUTE	*Valid*	-	
11A4	*Valid*	*Null*	Calendar.MINUTE	*Valid*	-	
11A5	*Valid*	*Valid*	Calendar.ERA	*Valid*	*T*	
11A6	*Valid*	*Valid*	Calendar.YEAR	*Valid*	*T*	
11A7	*Valid*	*Valid*	Calendar.MONTH	*Valid*	*T*	
11A10	*Valid*	*Valid*	Calendar.DATE	*Valid*	*T*	
11A11	*Valid*	*Valid*	Calendar.DAY_OF_MONTH	*Valid*	*T*	
11A15	*Valid*	*Valid*	Calendar.AM_PM	*Valid*	*T*	
11A16	*Valid*	*Valid*	Calendar.HOUR	*Valid*	*T*	
11A17	*Valid*	*Valid*	Calendar.HOUR_OF_DAY	*Valid*	*T*	
11A18	*Valid*	*Valid*	Calendar.SECOND	*Valid*	*T*	
11A19	*Valid*	*Valid*	Calendar.MILLISECOND	*Valid*	*T*	
11A20	*Valid*	*Valid*	Value out of Calendar Fields	*Invalid*	-	
11A22	*Valid*	*Valid*	Calendar.ERA	*Valid*	*F*	

11A23	Valid	Valid	Calendar.YEAR	Valid	F	
11A24	Valid	Valid	Calendar.MONTH	Valid	F	
11A27	Valid	Valid	Calendar.DATE	Valid	F	
11A28	Valid	Valid	Calendar.DAY_OF_MONTH	Valid	F	
11A32	Valid	Valid	Calendar.AM_PM	Valid	F	
11A33	Valid	Valid	Calendar.HOUR	Valid	F	
11A34	Valid	Valid	Calendar.HOUR_OF_DAY	Valid	F	
11A35	Valid	Valid	Calendar.SECOND	Valid	F	
11A36	Valid	Valid	Calendar.MILLISECOND	Valid	F	

Step 11B - Test Case Specs

Base Choice combinations of Date 1, Date 2, Calendar Field, Field Validity, and Equivalence. Two Base Choices were applied.

Case Id	Date 1	Date 2	Calendar Field	Field Validity	Equivalence	Notes
11B1	Valid	Valid	Calendar.MINUTE	Valid	T	Base Choice 1
11B2	Valid	Valid	Calendar.MINUTE	Valid	F	Base Choice 2
11B3	Null	Valid	Calendar.MINUTE	Valid	-	
11B4	Valid	Null	Calendar.MINUTE	Valid	-	
11B5	Valid	Valid	Calendar.ERA	Valid	T	
11B6	Valid	Valid	Calendar.YEAR	Valid	T	
11B7	Valid	Valid	Calendar.MONTH	Valid	T	
11B10	Valid	Valid	Calendar.DATE	Valid	T	

11B11	*Valid*	*Valid*	Calendar.DAY_OF_MONTH	*Valid*	*T*	
11B15	*Valid*	*Valid*	Calendar.AM_PM	*Valid*	*T*	
11B16	*Valid*	*Valid*	Calendar.HOUR	*Valid*	*T*	
11B17	*Valid*	*Valid*	Calendar.HOUR_OF_DAY	*Valid*	*T*	
11B18	*Valid*	*Valid*	Calendar.MINUTE	*Valid*	*T*	
11B19	*Valid*	*Valid*	Calendar.SECOND	*Valid*	*T*	
11B20	*Valid*	*Valid*	Calendar.MILLISECOND	*Valid*	*T*	
11B21	*Valid*	*Valid*	Value out of Calendar Fields	*Invalid*	-	
11B23	*Valid*	*Valid*	Calendar.ERA	*Valid*	*F*	
11B24	*Valid*	*Valid*	Calendar.YEAR	*Valid*	*F*	
11B25	*Valid*	*Valid*	Calendar.MONTH	*Valid*	*F*	
11B28	*Valid*	*Valid*	Calendar.DATE	*Valid*	*F*	
11B29	*Valid*	*Valid*	Calendar.DAY_OF_MONTH	*Valid*	*F*	
11B33	*Valid*	*Valid*	Calendar.AM_PM	*Valid*	*F*	
11B34	*Valid*	*Valid*	Calendar.HOUR	*Valid*	*F*	
11B35	*Valid*	*Valid*	Calendar.HOUR_OF_DAY	*Valid*	*F*	
11B36	*Valid*	*Valid*	Calendar.MINUTE	*Valid*	*F*	
11B37	*Valid*	*Valid*	Calendar.SECOND	*Valid*	*F*	
11B38	*Valid*	*Valid*	Calendar.MILLISECOND	*Valid*	*F*	

Test Cases

For each capability, aggregate create test case specs. Choose representative values for each participating attribute and specify any expected observable effects based on the chosen values (this is effectively the oracle). Effects may correspond to return values of functions/methods tested or observed changes in program state when those functions/methods are executed.

Capability 1 IsSame

Case #	Source Case Id	Attributes			Expected Effects	
		Date 1	Date 2	Equivalence	Return Boolean	Throw Exception
1.1	1A1	Valid	Valid	Equivalent-Day	*True*	-
1.2	1A2	Valid	Valid	Inequivalent-Day	*False*	-
1.3	1A3	Valid	Valid	Equivalent-Instant	*True*	-
1.4	1A4	Valid	Valid	Inequivalent-Instant	*False*	-
1.5	1A5	Valid	Invalid	-	-	*NullPointerException*
1.6	1A6	Invalid	Valid	-	-	*NullPointerException*
1.7	1A7	Invalid	Invalid	-	-	*NullPointerException*

Capability 1 IsSame (continued)

Case #	Source Case Id	Attributes			Expected Effects	
		Calendar 1	Calendar 2	Equivalence	Return Boolean	Throw Exception
1.8	1B1	Valid	Valid	Equivalent-Day	*True*	-
1.9	1B2	Valid	Valid	Inequivalent-Day	*False*	-
1.1	1B3	Valid	Valid	Equivalent-	*True*	-

0				Instant		
1.11	1B4	Valid	Valid	Inequivalent-Instant	*False*	-
1.12	1B5	Valid	Valid	Equivalent-LocalTime	*True*	
1.13	1B6	Valid	Valid	Inequivalent-LocalTime	*False*	
1.14	1B7	Valid	Invalid	-	-	*NullPointerException*
1.15	1B8	Invalid	Valid	-	-	*NullPointerException*
1.16	1B9	Invalid	Invalid	-	-	*NullPointerException*

Capability 2 Parse Date

Case #	Source Case Id	Attributes		Expected Effects	
		dateStr	parsePatterns	parseDate	parseDateStrictly
2.1	2A1	"Wed, 09 Apr 2008 23:55:38 GMT"	"EEE, dd MMM yyyy HH:mm:ss zzz"	*Returns a correct Date object*	*Returns a correct Date object*
2.2	2A2	"Wed, 09 Apr 2008 23:55:38 GMT"	["EEE, dd MMM yyyy HH:mm:ss zzz", "yyyy'-'MM'-'dd"]	*Returns a correct Date object*	*Returns a correct Date object*
2.3	2A3	null	"EEE, dd MMM yyyy HH:mm:ss zzz"	NullPointerException	NullPointerException
2.4	2A4	“”	"EEE, dd MMM yyyy HH:mm:ss zzz"	ParseException	ParseException

2.5	2A5	null	["EEE, dd MMM yyyy HH:mm:ss zzz", "yyyy'-'MM'-'dd"]	ParseException	ParseException
2.6	2A6	“”	["EEE, dd MMM yyyy HH:mm:ss zzz", "yyyy'-'MM'-'dd"]	ParseException	ParseException
2.7	2A7	"Wed, 09 Apr 2008 23:55:38 GMT"	null	NullPointerException	NullPointerException
2.8	2A8	"Wed, 09 Apr 2008 23:55:38 GMT"	[]	ParseException	ParseException
2.9	2A9	"Abc, 09 Apr 2008 23:55:38 GMT"	"EEE, dd MMM yyyy HH:mm:ss zzz"	ParseException	ParseException
2.10	2A10	"Abc, 09 Apr 2008 23:55:38 GMT"	["EEE, dd MMM yyyy HH:mm:ss zzz", "yyyy'-'MM'-'dd"]	ParseException	ParseException
2.11	2A11	"Thr, 09 Apr 2008 23:55:38 GMT"	"EEE, dd MMM yyyy HH:mm:ss zzz"	ParseException	ParseException
2.12	2A12	"Thr, 09 Apr 2008 23:55:38 GMT"	["EEE, dd MMM yyyy HH:mm:ss zzz", "yyyy'-'MM'-'dd"]	ParseException	ParseException
2.13	2A13	"Wed, 09 Abc 2008 23:55:38 GMT"	"EEE, dd MMM yyyy HH:mm:ss zzz"	ParseException	ParseException
2.14	2A14	"Wed, 09 Abc 2008 23:55:38 GMT"	["EEE, dd MMM yyyy HH:mm:ss zzz", "yyyy'-'MM'-'dd"]	ParseException	ParseException
2.15	2A15	"Wed, 09 Apr 2008 23:55:38 ABC"	"EEE, dd MMM yyyy HH:mm:ss zzz"	ParseException	ParseException

2.16	2A16	"Wed, 09 Apr 2008 23:55:38 ABC"	["EEE, dd MMM yyyy HH:mm:ss zzz", "yyyy'-'MM'-'dd"]	ParseException	ParseException
2.17	2A17	"Not valid string"	"EEE, dd MMM yyyy HH:mm:ss zzz"	ParseException	ParseException
2.18	2A18	"Not valid string"	["EEE, dd MMM yyyy HH:mm:ss zzz", "yyyy'-'MM'-'dd"]	ParseException	ParseException
2.19	2A19	"Wed, 09 Apr 2008 23:55:38 GMT"	["EEE, dd MMM yyyy HH:mm:ss zzz", "Not a valid pattern"]	*Returns a correct Date object*	*Returns a correct Date object*
2.20	2A20	"Wed, 09 Apr 2008 23:55:38 GMT"	"Not a valid pattern"	IllegalArgumentException	IllegalArgumentException
2.21	2A21	"Wed, 09 Apr 2008 23:55:38 GMT"	["Not a valid pattern", "Another not valid pattern"]	IllegalArgumentException	IllegalArgumentException
2.22	2A22	"Wed, 09 Apr 2008 23:55:38 GMT"	"yyyy'-'MM'-'dd"	ParseException	ParseException
2.23	2A23	"Wed, 09 Apr 2008 23:55:38 GMT"	["yyyy HH:mm", "yyyy'-'MM'-'dd"]	ParseException	ParseException
2.24	2A24	"Wed, 13/09 2008 23:55:38 GMT"	"EEE, MM/dd yyyy HH:mm:ss zzz"	*Returns a correct Date object*	ParseException
2.25	2A25	"Wed, 00/09 2008 23:55:38 GMT"	["EEE, MM/dd yyyy HH:mm:ss zzz", "yyyy'-'MM'-'dd"]	*Returns a correct Date object*	ParseException
2.26	2A26	"Wed, 32 Apr 2008 23:55:38 GMT"	"EEE, dd MMM yyyy HH:mm:ss zzz"	*Returns a correct Date object*	ParseException

2.27	2A27	"Wed, 00 Apr 2008 23:55:38 GMT"	["EEE, dd MMM yyyy HH:mm:ss zzz", "yyyy'-'MM'-'dd"]	*Returns a correct Date object*	ParseException
2.28	2A28	"Wed, 09 Apr 2008 24:55:38 GMT"	"EEE, dd MMM yyyy HH:mm:ss zzz"	*Returns a correct Date object*	ParseException
2.29	2A29	"Wed, 09 Apr 2008 25:55:38 GMT"	["EEE, dd MMM yyyy HH:mm:ss zzz", "yyyy'-'MM'-'dd"]	*Returns a correct Date object*	ParseException
2.30	2A30	"Wed, 09 Apr 2008 23:60:38 GMT"	"EEE, dd MMM yyyy HH:mm:ss zzz"	*Returns a correct Date object*	ParseException
2.31	2A31	"Wed, 09 Apr 2008 23:70:38 GMT"	["EEE, dd MMM yyyy HH:mm:ss zzz", "yyyy'-'MM'-'dd"]	*Returns a correct Date object*	ParseException
2.32	2A32	"Wed, 09 Apr 2008 23:55:60 GMT"	"EEE, dd MMM yyyy HH:mm:ss zzz"	*Returns a correct Date object*	ParseException
2.33	2A33	"Wed, 09 Apr 2008 23:55:99 GMT"	["EEE, dd MMM yyyy HH:mm:ss zzz", "yyyy'-'MM'-'dd"]	*Returns a correct Date object*	ParseException

Capability 3 Date Addition

Case #	Source Case Id	Attributes					Expected Effects	
		Date(yyyy-mm-dd hh:mm:ss.SSS)	Amount					
			Year	Month	Week	Day	Return Date(yyyy-mm-dd hh:mm:ss.SSS)	Throws Exception
3.1	3A1	*2023-10-29 22:33:44.555*	*1*	-	-	-	*2024-10-29 22:33:44.555*	-

3.2	3A2	*2023-10-29 22:33:44.555*	-1	-	-	-	*2022-10-29 22:33:44.555*	-
3.3	3B1	*2023-10-29 22:33:44.555*	-	1	-	-	*2023-10-30 22:33:44.555*	-
3.4	3B2	*2023-10-29 22:33:44.555*	-	-1	-	-	*2023-10-28 22:33:44.555*	-
3.5	3C1	*2023-10-29 22:33:44.555*	-	-	1	-	*2023-11-05 22:33:44.555*	-
3.6	3C2	*2023-10-29 22:33:44.555*	-	-	-1	-	*2023-10-22 22:33:44.555*	-
3.7	3D1	*2023-10-29 22:33:44.555*	-	-	-	*1*	*2023-10-30 22:33:44.555*	-
3.8	3D2	*2023-10-29 22:33:44.555*	-	-	-	-1	*2023-10-28 22:33:44.555*	-
3.17	3A3	*null*	*1*	-	-	-	-	*NullPointerException*
3.18	3B3	*null*	-	*1*	-	-	-	*NullPointerException*
3.19	3C3	*null*	-	-	*1*	-	-	*NullPointerException*
3.20	3D3	*null*	-	-	-	*1*	-	*NullPointerException*

Capability 3 Date Addition (continued)

Case #	Source Case Id	Attributes					Expected Effects	
		Date(yyyy-mm-dd hh:mm:ss.SSS)	Amount					
			Hour	Minute	Second	Millisecond	Return Date(yyyy-mm-dd hh:mm:ss.SSS)	Throws Exception
3.9	3E1	*2023-10-29 22:33:44.555*	1	-	-	-	*2023-10-29 23:33:44.555*	-
3.10	3E2	*2023-10-29 22:33:44.555*	-1	-	-	-	*2023-10-29 21:33:44.555*	-
3.11	3F1	*2023-10-29 22:33:44.555*	-	1	-	-	*2023-10-29 22:34:44.555*	-

3.12	3F2	*2023-10-29 22:33:44.555*	-	-1	-	-	*2023-10-29 22:32:44.555*	-
3.13	3G1	*2023-10-29 22:33:44.555*	-	-	1	-	*2023-10-29 22:33:45.555*	-
3.14	3G2	*2023-10-29 22:33:44.555*	-	-	-1	-	*2023-10-29 22:33:43.555*	-
3.15	3H1	*2023-10-29 22:33:44.555*	-	-		1	*2023-10-29 22:33:44.556*	-
3.16	3H2	*2023-10-29 22:33:44.555*	-	-	-	-1	*2023-10-29 22:33:44.554*	-
3.21	3E3	*null*	1	-	-	-	-	*NullPointerException*
3.22	3F3	*null*	-	1	-	-	-	*NullPointerException*
3.23	3G3	*null*	-	-	1	-	-	*NullPointerException*
3.24	3H3	*null*	-	-	-	1	-	*NullPointerException*

Capability 4 Setting Date

Case #	Source Case Id	Attributes				Expected Effects	
		Date(yyyy-mm-dd hh:mm:ss.SSS)	Amount			Return Date(yyyy-mm-dd hh:mm:ss.SSS)	Throw Exception
			Year	Month	Day		
4.1	4A1	*2023-10-29 22:33:44.555*	*2001*	-	-	*2001-10-29 22:33:44.555*	-
4.2	4A2	*2023-10-29 22:33:44.555*	-1	-	-	-	*java.lang.IllegalArgumentException*
4.3	4B1	*2023-10-29 22:33:44.555*	-	3	-	*2023-4-30 22:33:44.555*	-
4.28	4B5	*2023-10-29 22:33:44.555*	-	1		-	*java.lang.IllegalArgumentException*
4.4	4B2	*2023-10-29 22:33:44.555*	-	-1	-	-	*java.lang.IllegalArgumentException*
4.5	4B3	*2023-10-29 22:33:44.555*	-	12	-	-	*java.lang.IllegalArgumentException*
4.6	4C1	*2023-10-29 22:33:44.555*	-	-	1	*2023-10-01 22:33:44.555*	
4.7	4C2	*2023-10-29 22:33:44.555*	-	-	*0*	-	*java.lang.IllegalArgumentException*
4.8	4C3	*2023-10-29 22:33:44.555*	-	-	32	-	*java.lang.IllegalArgumentException*
4.29	4C5	*2023-2-28 22:33:44.555*			29		
4.30	4C6	*2023-10-29 22:33:44.555*			31		
4.21	4A3	*null*	2001	-	-	-	*NullPointerException*
4.22	4B4	*null*	-	3	-	-	*NullPointerException*
4.23	4C4	*null*	-		1	-	*NullPointerException*
4.29	4B6	*2023-3-31 22:33:44.555*	-	3	-	-	*java.lang.IllegalArgumentException*
4.30	4C6	*2000-02-28*	-	-	30		*java.lang.IllegalArgu*

Case #	Source Case Id	Attributes				Expected Effects	
		Date(yyyy-mm-dd hh:mm:ss.SSS)	Amount			Return Date(yyyy-mm-dd hh:mm:ss.SSS)	Throw Exception
			Year	Month	Day		
4.1	4A1	*2023-10-29 22:33:44.555*	*2001*	-	-	*2001-10-29 22:33:44.555*	-
4.2	4A2	*2023-10-29 22:33:44.555*	-1	-	-	-	*java.lang.IllegalArgumentException*
		22:33:44.555					*mentException*
4.31	4C5	*2023-2-28 22:33:44.555*	-	-	29	-	*java.lang.IllegalArgumentException*
4.32	4C7	*2023-4-28 22:33:44.555*	-	-	31	-	*java.lang.IllegalArgumentException*

Capability 4 Setting Date (continued)

Case #	Source Case Id	Attributes			Expected Effects	
		Date(yyyy-mm-dd hh:mm:ss.SSS)	Hour Amount	Minute Amount	Return Value Date(yyyy-mm-dd hh:mm:ss.SSS)	Throw Exception
4.9	4D1	*2023-10-29 22:33:44.555*	0	-	*2023-10-29 00:33:44.555*	-
4.10	4D2	*2023-10-29 22:33:44.555*	-1	-	-	*java.lang.IllegalArgumentException*
4.11	4D3	*2023-10-29 22:33:44.555*	24	-	-	*java.lang.IllegalArgumentException*
4.12	4E1	*2023-10-29 22:33:44.555*	-	0	*2023-10-29 22:00:44.555*	-
4.13	4E2	*2023-10-29 22:33:44.555*	-	-1	-	*java.lang.IllegalArgumentException*
4.14	4E3	*2023-10-29 22:33:44.555*	-	60	-	*java.lang.IllegalArgumentException*
4.24	4D4	*null*	0	-	-	*NullPointerException*
4.25	4E4	*null*	-	0	-	*NullPointerException*

Capability 4 Setting Date (continued)

Case #	Source Case Id	Attributes			Expected Effects	
		Date(yyyy-mm-dd hh:mm:ss.SSS)	Second Amount	Millisecond Amount	Return Date(yyyy-mm-dd hh:mm:ss.SSS)	Throw Exception
4.15	4F1	*2023-10-29 22:33:44.555*	0	-	*2023-10-29 23:33:00.555*	-
4.16	4F2	*2023-10-29 22:33:44.555*	-1	-	-	*java.lang.IllegalArgumentException*
4.17	4F3	*2023-10-29 22:33:44.555*	60	-	-	*java.lang.IllegalArgumentException*
4.18	4G1	*2023-10-29 22:33:44.555*	-	0	*2023-10-29 22:32:44.000*	-
4.19	4G2	*2023-10-29 22:33:44.555*	-	-1	-	*java.lang.IllegalArgumentException*
4.20	4G3	*2023-10-29 22:33:44.555*	-	1000	-	*java.lang.IllegalArgumentException*
4.26	4F4	*null*	0	-	-	*NullPointerException*
4.27	4G4	*nill*	-	0	-	*NullPointerException*

Capability 5 Converting to Date

Case #	Source Case Id	Attributes		Expected Effects	
		Date	Timezone	Return Calendar	Throw Exception
5.1	5A1	Sun Nov 18 01:23:11 PST 2001	-	*Calendar Object*	-
5.2	5A2	Null	-	-	*NullPointerException*
5.3	5A3	Sun Nov 18 01:23:11 PST 2001	DEFAULT_ZONE	*Calendar Object*	-
5.4	5A4	Null	DEFAULT_	-	*NullPointerException*

			ZONE		
5.5	5A5	Sun Nov 18 01:23:11 PST 2001	Null	-	*NullPointerException*
5.6	5A6	Null	Null	-	*NullPointerException*

Capability 6 Round a Date

Case #	Source Case Id	Attributes		Expected Effects	
		Date	Field	Result	Throw Exception
6.1	6A1	2008-04-09 23:55:27:025	Calendar.MINUTE	2008-04-09 23:55:00:000	-
6.2	6A2	2008-12-31 23:59:20:025	Calendar.MINUTE	2008-12-31 23:59:00:000	-
6.3	6A3	2008-12-31 23:59:45:025	Calendar.MINUTE	2009-01-01 00:00:00:000	-
6.4	6A4	2008-04-09 23:55:00:000	Calendar.MINUTE	2008-04-09 23:55:00:000	-
6.5	6A5	2008-04-09 23:55:27:025	Calendar.MILLISECOND	2008-04-09 23:55:27:025	-
6.6	6A6	2008-04-09 23:55:27:025	Calendar.SECOND	2008-04-09 23:55:27:000	-
6.7	6A7	2008-04-09 23:25:27:025	Calendar.HOUR_OF_DAY	2008-04-09 23:00:00:000	-
6.8	6A8	2008-04-09 23:25:27:025	Calendar.HOUR	2008-04-09 23:00:00:000	-
6.9	6A9	2008-04-09 11:25:27:025	Calendar.DATE	2008-04-09 00:00:00:000	-
6.10	6A10	2008-04-09 11:25:27:025	Calendar.DAY_OF_MONTH	2008-04-09 00:00:00:000	

6.11	6A11	2008-04-09 03:25:27:025	Calendar.AM_PM	2008-04-09 00:00:00:000	-
6.12	6A12	2008-04-09 03:25:27:025	Calendar.MONTH	2008-04-01 00:00:00:000	-
6.13	6A13	2008-04-18 03:25:27:025	DateUtils.SEMI_MONTH	2008-04-16 00:00:00:000	-
6.14	6A14	2008-04-18 03:25:27:025	Calendar.YEAR	2008-01-01 00:00:00:000	-
6.15	6A15	2008-04-18 03:25:27:025	Calendar.ERA	0001-01-01 00:00:00:000	-
6.16	6A16	2008-04-09 23:55:27:025	-1234	-	IllegalArgumentException
6.17	6A17	2008-04-09 23:55:27:025	Calendar.MINUTE	Date(2008-04-09 23:55:00:000)	-
6.18	6A18	null	Calendar.MINUTE	-	NullPointerException
6.19	6A19	"Wed, 09 Apr 2008 23:55:38 GMT"	Calendar.MINUTE	-	ClassCastException

Capability 6 Truncate a Date

Case #	Source Case Id	Attributes		Expected Effects	
		Date	Field	Result	Throw Exception
6.20	6B1	2008-04-09 23:55:27:025	Calendar.MINUTE	2008-04-09 23:55:00:000	-
6.21	6B2	2008-04-09 23:55:00:000	Calendar.MINUTE	2008-04-09 23:55:00:000	-

6.22	6B3	2008-04-09 23:55:27:025	Calendar.MILLISECOND	2008-04-09 23:55:27:025	-
6.23	6B4	2008-04-09 23:55:27:025	Calendar.SECOND	2008-04-09 23:55:27:000	-
6.24	6B5	2008-04-09 23:25:27:025	Calendar.HOUR_OF_DAY	2008-04-09 23:00:00:000	-
6.25	6B6	2008-04-09 23:25:27:025	Calendar.HOUR	2008-04-09 23:00:00:000	-
6.26	6B7	2008-04-09 11:25:27:025	Calendar.DATE	2008-04-09 00:00:00:000	-
6.27	6B8	2008-04-09 11:25:27:025	Calendar.DAY_OF_MONTH	2008-04-09 00:00:00:000	
6.28	6B9	2008-04-09 03:25:27:025	Calendar.AM_PM	2008-04-09 00:00:00:000	-
6.29	6B10	2008-04-09 03:25:27:025	Calendar.MONTH	2008-04-01 00:00:00:000	-
6.30	6B11	2008-04-18 03:25:27:025	DateUtils.SEMI_MONTH	2008-04-16 00:00:00:000	-
6.31	6B12	2008-04-18 03:25:27:025	Calendar.YEAR	2008-01-01 00:00:00:000	-
6.32	6B13	2008-04-18 03:25:27:025	Calendar.ERA	0001-01-01 00:00:00:000	-
6.33	6B14	2008-04-09 23:55:27:025	-1234	-	IllegalArgumentException
6.34	6B15	2008-04-09 23:55:27:025	Calendar.MINUTE	Date(2008-04-09 23:55:00:000)	-
6.35	6B16	null	Calendar.MINUTE	-	NullPointerException
6.36	6B17	"Wed, 09 Apr 2008 23:55:38 GMT"	Calendar.MINUTE	-	ClassCastException

Capability 6 Ceiling a Date

Case #	Source Case Id	Attributes		Expected Effects	
		Date	Field	Result	Throw Exception
6.37	6C1	2008-04-09 23:55:38:025	Calendar.MINUTE	2008-04-09 23:56:00:000	-
6.38	6C2	2008-12-31 23:59:45:025	Calendar.MINUTE	2009-01-01 00:00:00:000	-
6.39	6C3	2008-04-09 23:55:38:025	Calendar.MILLISECOND	2008-04-09 23:55:38:025	-
6.40	6C4	2008-04-09 23:55:38:025	Calendar.SECOND	2008-04-09 23:55:39:000	-
6.41	6C5	2008-04-09 22:25:27:025	Calendar.HOUR_OF_DAY	2008-04-09 23:00:00:000	-
6.42	6C6	2008-04-09 23:25:27:025	Calendar.HOUR	2008-04-09 23:00:00:000	-
6.43	6C7	2008-04-09 11:25:27:025	Calendar.DATE	2008-04-10 00:00:00:000	-
6.44	6C8	2008-04-09 11:25:27:025	Calendar.DAY_OF_MONTH	2008-04-10 00:00:00:000	
6.45	6C9	2008-04-09 18:25:27:025	Calendar.AM_PM	2008-04-10 00:00:00:000	-
6.46	6C10	2008-04-09 03:25:27:025	Calendar.MONTH	2008-05-01 00:00:00:000	-
6.47	6C11	2008-04-28 03:25:27:025	DateUtils.SEMI_MONTH	2008-05-01 00:00:00:000	-
6.48	6C12	2008-04-18 03:25:27:025	Calendar.YEAR	2009-01-01 00:00:00:000	-
6.49	6C13	2008-04-18 03:25:27:025	Calendar.ERA	0001-01-01 00:00:00:000	-
6.50	6C14	2008-04-09	-1234	-	IllegalArgumentE

		23:55:27:025			xception
6.51	6C15	2008-04-09 23:55:38:025	Calendar.MINUTE	Date(2008-04-09 23:56:00:000)	-
6.52	6C16	null	Calendar.MINUTE	-	NullPointerException
6.53	6C17	"Wed, 09 Apr 2008 23:55:38 GMT"	Calendar.MINUTE	-	ClassCastException

Capability 8 Date Iterator

Case #	Source Case Id	Attributes		Expected Effects
		CalendarObj	RangeStyle	Throw Exception
8.1	8A1	null	RANGE_MONTH_SUNDAY	NullPointerException
8.2	8A2	null	RANGE_MONTH_MONDAY	NullPointerException
8.3	8A3	null	RANGE_WEEK_SUNDAY	NullPointerException
8.4	8A4	null	RANGE_WEEK_MONDAY	NullPointerException
8.5	8A5	null	RANGE_WEEK_RELATIVE	NullPointerException
8.6	8A6	null	RANGE_WEEK_CENTER	NullPointerException
8.7	8A7	null	Invalid	NullPointerException
8.8	8A15	InvalidObj	RANGE_MONTH_SUNDAY	ClassCastException
8.9	8A16	InvalidObj	RANGE_MONTH_MONDAY	ClassCastException
8.10	8A17	InvalidObj	RANGE_WEEK_SUNDAY	ClassCastException
8.11	8A18	InvalidObj	RANGE_WEEK_MONDAY	ClassCastException
8.12	8A19	InvalidObj	RANGE_WEEK_RELATIVE	ClassCastException

8.13	8A20	InvalidObj	RANGE_WEEK_CENTER	ClassCastException
8.14	8A21	InvalidObj	Invalid	ClassCastException

(8A8 - 8A14 is covered by following specs)

Capability 8 Date Iterator (continue)

Case #	Source Case Id	Attributes		Expected Effects		
		Calendar	RangeStyle	Throw Exception	hasNext()	next()
8.15	8B1	Current date	RANGE_MONTH_SUNDAY	-	true	Next date of current date
8.16	8B2	Current date	RANGE_MONTH_MONDAY	-	true	Next date of current date
8.17	8B3	Current date	RANGE_WEEK_SUNDAY	-	true	Next date of current date
8.18	8B4	Current date	RANGE_WEEK_MONDAY	-	true	Next date of current date
8.19	8B5	Current date	RANGE_WEEK_RELATIVE	-	true	Next date of current date
8.20	8B6	Current date	RANGE_WEEK_CENTER	-	true	Next date of current date
8.21	8B7	Current date	null	NullPointerException	-	-

Capability 8 Date Iterator (continue)

Case #	Source Case Id	Attributes		Expected Effects		
		Date	RangeStyle	Throw Exception	hasNext()	next()
8.22	8C1	Current date	RANGE_MONTH_SUNDAY	-	true	Next date of current date
8.23	8C2	Current date	RANGE_MONTH_MONDAY	-	true	Next date of current date
8.24	8C3	Current date	RANGE_WEEK_SUNDAY	-	true	Next date of current date
8.25	8C4	Current date	RANGE_WEEK_MONDAY	-	true	Next date of current date
8.26	8C5	Current date	RANGE_WEEK_RELATIVE	-	true	Next date of current date
8.27	8C6	Current date	RANGE_WEEK_CENTER	-	true	Next date of current date
8.28	8C7	Current date	null	NullPointerException	-	-

Capability 9 Getting Date Fragment

Case #	Source Case Id	Attributes		Expected Effects	
		Date	fragment	number of milliseconds	Throw Exception
1.1	9A1	January 1, 2008 7:15:10.538	MILLISECOND	0	-
1.2	9A2	January 1, 2008 7:15:10.538	SECOND	538	-
1.3	9A3	January 1, 2008	MINUTE	10538	-

		7:15:10.538			
1.4	9A4	January 1, 2008 7:15:10.538	HOUR_OF_DAY	910538	-
1.5	9A5	January 1, 2008 7:15:10.538	DATE	26110538	-
1.6	9A6	January 1, 2008 7:15:10.538	DAY_OF_YEAR	26110538	-
1.7	9A7	January 1, 2008 7:15:10.538	MONTH	26110538	-
1.8	9A8	January 1, 2008 7:15:10.538	YEAR	2704510538	-
1.9	9A9	January 1, 2008 7:15:10.538	0	-	IllegalArgumentException
1.10	9A10	null	MILLISECOND	-	NullPointerException

Capability 9 Getting Date Fragment (continue)

Case #	Source Case Id	Attributes		Expected Effects	
		Date	fragment	number of seconds	Throw Exception
1.11	9B1	January 1, 2008 7:15:10.538	MILLISECOND	0	-
1.12	9B2	January 1, 2008 7:15:10.538	SECOND	0	-
1.13	9B3	January 1, 2008 7:15:10.538	MINUTE	10	-

1.14	9B4	January 1, 2008 7:15:10.538	HOUR_OF_DAY	910	-
1.15	9B5	January 1, 2008 7:15:10.538	DATE	26110	-
1.16	9B6	January 1, 2008 7:15:10.538	DAY_OF_YEAR	26110	-
1.17	9B7	January 1, 2008 7:15:10.538	MONTH	26110	-
1.18	9B8	January 1, 2008 7:15:10.538	YEAR	2704510	-
1.19	9B9	January 1, 2008 7:15:10.538	0	-	IllegalArgumentException
1.20	9B10	null	MILLISECOND	-	NullPointerException

Capability 9 Getting Date Fragment (continue)

Case #	Source Case Id	Attributes		Expected Effects	
		Date	fragment	number of minutes	Throw Exception
1.21	9C1	January 1, 2008 7:15:10.538	MILLISECOND	0	-
1.22	9C2	January 1, 2008 7:15:10.538	SECOND	0	-
1.23	9C3	January 1, 2008 7:15:10.538	MINUTE	0	-
1.24	9C4	January 1,	HOUR_OF_DAY	15	-

		2008 7:15:10.538			
1.25	9C5	January 1, 2008 7:15:10.538	DATE	435	-
1.26	9C6	January 1, 2008 7:15:10.538	DAY_OF_YEAR	435	-
1.27	9C7	January 1, 2008 7:15:10.538	MONTH	435	-
1.28	9C8	January 1, 2008 7:15:10.538	YEAR	45075	-
1.29	9C9	January 1, 2008 7:15:10.538	0	-	IllegalArgumentException
1.30	9C10	null	MILLISECOND	-	NullPointerException

Capability 9 Getting Date Fragment (continue)

Case #	Source Case Id	Attributes		Expected Effects	
		Date	fragment	number of hours	Throw Exception
1.31	9D1	January 1, 2008 7:15:10.538	MILLISECOND	0	-
1.32	9D2	January 1, 2008 7:15:10.538	SECOND	0	-
1.33	9D3	January 1, 2008 7:15:10.538	MINUTE	0	-
1.34	9D4	January 1, 2008	HOUR_OF_DAY	0	-

		7:15:10.538			
1.35	9D5	January 1, 2008 7:15:10.538	DATE	7	-
1.36	9D6	January 1, 2008 7:15:10.538	DAY_OF_YEAR	7	-
1.37	9D7	January 1, 2008 7:15:10.538	MONTH	7	-
1.38	9D8	January 1, 2008 7:15:10.538	YEAR	751	-
1.39	9D9	January 1, 2008 7:15:10.538	0	-	IllegalArgumentException
1.40	9D10	null	MILLISECOND	-	NullPointerException

Capability 9 Getting Date Fragment (continue)

Case #	Source Case Id	Attributes		Expected Effects	
		Date	fragment	number of days	Throw Exception
1.41	9E1	January 1, 2008 7:15:10.538	MILLISECOND	0	-
1.42	9E2	January 1, 2008 7:15:10.538	SECOND	0	-
1.43	9E3	January 1, 2008 7:15:10.538	MINUTE	0	-
1.44	9E4	January 1, 2008 7:15:10.538	HOUR_OF_DAY	0	-

1.45	9E5	January 1, 2008 7:15:10.538	DATE	0	-
1.46	9E6	January 1, 2008 7:15:10.538	DAY_OF_YEAR	0	-
1.47	9E7	January 1, 2008 7:15:10.538	MONTH	1	-
1.48	9E8	January 1, 2008 7:15:10.538	YEAR	32	-
1.49	9E9	January 1, 2008 7:15:10.538	0	-	IllegalArgumentException
1.50	9E10	null	MILLISECOND	-	NullPointerException

Capability 9 Getting Date Fragment (continue)

Case #	Source Case Id	Attributes		Expected Effects	
		Calendar	fragment	number of milliseconds	Throw Exception
1.51	9F1	January 1, 2008 7:15:10.538	MILLISECOND	0	-
1.52	9F2	January 1, 2008 7:15:10.538	SECOND	538	-
1.53	9F3	January 1, 2008 7:15:10.538	MINUTE	10538	-
1.54	9F4	January 1, 2008 7:15:10.538	HOUR_OF_DAY	910538	-
1.55	9F5	January 1,	DATE	26110538	-

		2008 7:15:10.538			
1.56	9F6	January 1, 2008 7:15:10.538	DAY_OF_YEAR	26110538	-
1.57	9F7	January 1, 2008 7:15:10.538	MONTH	26110538	-
1.58	9F8	January 1, 2008 7:15:10.538	YEAR	2704510538	-
1.59	9F9	January 1, 2008 7:15:10.538	0	-	IllegalArgumentException
1.60	9F10	null	MILLISECOND	-	NullPointerException

Capability 9 Getting Date Fragment (continue)

Case #	Source Case Id	Attributes		Expected Effects	
		Calendar	fragment	number of seconds	Throw Exception
1.61	9G1	January 1, 2008 7:15:10.538	MILLISECOND	0	-
1.62	9G2	January 1, 2008 7:15:10.538	SECOND	0	-
1.63	9G3	January 1, 2008 7:15:10.538	MINUTE	10	-
1.64	9G4	January 1, 2008 7:15:10.538	HOUR_OF_DAY	910	-
1.65	9G5	January 1, 2008	DATE	26110	-

		7:15:10.538			
1.66	9G6	January 1, 2008 7:15:10.538	DAY_OF_YEAR	26110	-
1.67	9G7	January 1, 2008 7:15:10.538	MONTH	26110	-
1.68	9G8	January 1, 2008 7:15:10.538	YEAR	2704510	-
1.69	9G9	January 1, 2008 7:15:10.538	0	-	IllegalArgumentException
1.70	9G10	null	MILLISECOND	-	NullPointerException

Capability 9 Getting Date Fragment (continue)

Case #	Source Case Id	Attributes		Expected Effects	
		Calendar	fragment	number of minutes	Throw Exception
1.71	9H1	January 1, 2008 7:15:10.538	MILLISECOND	0	-
1.72	9H2	January 1, 2008 7:15:10.538	SECOND	0	-
1.73	9H3	January 1, 2008 7:15:10.538	MINUTE	0	-
1.74	9H4	January 1, 2008 7:15:10.538	HOUR_OF_DAY	15	-
1.75	9H5	January 1, 2008 7:15:10.538	DATE	435	-

1.76	9H6	January 1, 2008 7:15:10.538	DAY_OF_YEAR	435	-
1.77	9H7	January 1, 2008 7:15:10.538	MONTH	435	-
1.78	9H8	January 1, 2008 7:15:10.538	YEAR	45075	-
1.79	9H9	January 1, 2008 7:15:10.538	0	-	IllegalArgumentException
1.80	9H10	null	MILLISECOND	-	NullPointerException

Capability 9 Getting Date Fragment (continue)

Case #	Source Case Id	Attributes		Expected Effects	
		Calendar	fragment	number of hours	Throw Exception
1.81	9I1	January 1, 2008 7:15:10.538	MILLISECOND	0	-
1.82	9I2	January 1, 2008 7:15:10.538	SECOND	0	-
1.83	9I3	January 1, 2008 7:15:10.538	MINUTE	0	-
1.84	9I4	January 1, 2008 7:15:10.538	HOUR_OF_DAY	0	-
1.85	9I5	January 1, 2008	DATE	7	-

		7:15:10.538			
1.86	9I6	January 1, 2008 7:15:10.538	DAY_OF_YEAR	7	-
1.87	9I7	January 1, 2008 7:15:10.538	MONTH	7	-
1.88	9I8	January 1, 2008 7:15:10.538	YEAR	751	-
1.89	9I9	January 1, 2008 7:15:10.538	0	-	IllegalArgumentException
1.90	9I10	null	MILLISECOND	-	NullPointerException

Capability 9 Getting Date Fragment (continue)

Case #	Source Case Id	Attributes		Expected Effects	
		Calendar	fragment	number of days	Throw Exception
1.91	9J1	January 1, 2008 7:15:10.538	MILLISECOND	0	-
1.92	9J2	January 1, 2008 7:15:10.538	SECOND	0	-
1.93	9J3	January 1, 2008 7:15:10.538	MINUTE	0	-
1.94	9J4	January 1, 2008 7:15:10.538	HOUR_OF_DAY	0	-
1.95	9J5	January 1,	DATE	0	-

		2008 7:15:10.538			
1.96	9J6	January 1, 2008 7:15:10.538	DAY_OF_YEAR	0	-
1.97	9J7	January 1, 2008 7:15:10.538	MONTH	1	-
1.98	9J8	January 1, 2008 7:15:10.538	YEAR	32	-
1.99	9J9	January 1, 2008 7:15:10.538	0	-	IllegalArgumentException
1.100	9J10	null	MILLISECOND	-	NullPointerException

Capability 11 Compare 2 Dates with Specified Precision

Case #	Source Case Id	Attributes					Expected Effects	
		Calendar 1	Calendar 2	Field	Field Validity	Equivalence	Return Boolean	Throw Exception
11.1	11A1	Sun Nov 18 01:23:11 PST 2001	Sun Nov 18 01:23:11 PST 2001	Calendar.MINUTE	T	T	*True*	-
11.2	11A2	Sun Nov 18 01:23:11 PST 2001	Sun Nov 18 01:23:11 PST 2002	Calendar.MINUTE	T	F	*False*	-
11.3	11A3	Null	Sun Nov 18 01:23:11	Calendar.MINUTE	-	-	-	*NullPointerException*

			PST 2001					
11.4	11A4	Sun Nov 18 01:23:11 PST 2001	Null	Calendar.MINUTE	-	-	-	*NullPointerException*
11.5	11A5	Sun Nov 18 01:23:11 PST 2001	Sun Nov 18 01:23:11 PST 2001	Calendar.ERA	T	T	*True*	-
11.6	11A6	Sun Nov 18 01:23:11 PST 2001	Sun Nov 18 01:23:11 PST 2001	Calendar.YEAR	T	T	*True*	-
11.7	11A7	Sun Nov 18 01:23:11 PST 2001	Sun Nov 18 01:23:11 PST 2001	Calendar.MONTH	T	T	*True*	-
11.8	11A8	Sun Nov 18 01:23:11 PST 2001	Sun Nov 18 01:23:11 PST 2001	Calendar.DATE	T	T	*True*	-
11.9	11A9	Sun Nov 18 01:23:11 PST 2001	Sun Nov 18 01:23:11 PST 2001	Calendar.DAY_OF_MONTH	T	T	*True*	-
11.10	11A10	Sun Nov 18 01:23:11 PST 2001	Sun Nov 18 01:23:11 PST 2001	Calendar.AM_PM	T	T	*True*	-
11.	11A	Sun	Sun Nov	Calendar.HOUR	T	T	*True*	-

11	11	Nov 18 01:23:11 PST 2001	18 01:23:11 PST 2001					
11.12	11A12	Sun Nov 18 01:23:11 PST 2001	Sun Nov 18 01:23:11 PST 2001	Calendar.HOUR_OF_DAY	T	T	*True*	-
11.13	11A13	Sun Nov 18 01:23:11 PST 2001	Sun Nov 18 01:23:11 PST 2001	Calendar.SECOND	T	T	*True*	-
11.14	11A14	Sun Nov 18 01:23:11 PST 2001	Sun Nov 18 01:23:11 PST 2001	Calendar.MILLISECOND	T	T	*True*	-
11.15	11A15	Sun Nov 18 01:23:11 PST 2001	Sun Nov 18 01:23:11 PST 2001	-1234	F	-	-	IllegalArgumentException
11.16	11A16	Sun Nov 18 01:23:11 PST 2001, AD	Sun Nov 18 01:23:11 PST 2002, BC	Calendar.ERA	T	F	*False*	-
11.17	11A17	Sun Nov 18 01:23:11 PST 2001	Sat Oct 12 15:29:53 PST 2002	Calendar.YEAR	T	F	*False*	-
11.18	11A18	Sun Nov 18 01:23:1	Sat Oct 12 15:29:53	Calendar.MONTH	T	F	*False*	-

		1 PST 2001	PST 2002					
11.19	11A19	Sun Nov 18 01:23:11 PST 2001	Sat Oct 12 15:29:53 PST 2002	Calendar.DATE	T	F	*False*	-
11.20	11A20	Sun Nov 18 01:23:11 PST 2001	Sat Oct 12 15:29:53 PST 2002	Calendar.DAY_OF_MONTH	T	F	*False*	-
11.21	11A21	Sun Nov 18 01:23:11 PST 2001	Sat Oct 12 15:29:53 PST 2002	Calendar.AM_PM	T	F	*False*	-
11 22	11A22	Sun Nov 18 01:23:11 PST 2001	Sat Oct 12 15:29:53 PST 2002	Calendar.HOUR	T	F	*False*	-
11.23	11A23	Sun Nov 18 01:23:11 PST 2001	Sat Oct 12 15:29:53 PST 2002	Calendar.HOUR_OF_DAY	T	F	*False*	-
11.24	11A24	Sun Nov 18 01:23:11 PST 2001	Sat Oct 12 15:29:53 PST 2002	Calendar.SECOND	T	F	*False*	-
11.25	11A25	Sun Nov 18 01:23:11 PST 2001	Sat Oct 12 15:29:53 PST 2002	Calendar.MILLISECOND	T	F	*False*	-

Capability 11 Compare 2 Dates with Specified Precision（Continued)

Case #	Source Case Id	Attributes					Expected Effects	
		Date 1	Date 2	Field	Field Validity	Equivalence	Return Boolean	Throw Exception
11.26	11B1	Sun Nov 18 01:23:11 PST 2001	Sun Nov 18 01:23:11 PST 2001	Calendar.MINUTE	T	T	*True*	-
11.27	11B2	Sun Nov 18 01:23:11 PST 2001	Sun Nov 18 01:23:11 PST 2002	Calendar.MINUTE	T	F	*False*	-
11.28	11B3	Null	Sun Nov 18 01:23:11 PST 2001	Calendar.MINUTE	-	-	-	*NullPointerException*
11.29	11B4	Sun Nov 18 01:23:11 PST 2001	Null	Calendar.MINUTE	-	-	-	*NullPointerException*
11.30	11B5	Sun Nov 18 01:23:11 PST 2001	Sun Nov 18 01:23:11 PST 2001	Calendar.ERA	T	T	*True*	-

11.31	11B6	Sun Nov 18 01:23:11 PST 2001	Sun Nov 18 01:23:11 PST 2001	Calendar.YEAR	T	T	*True*	-
11.32	11B7	Sun Nov 18 01:23:11 PST 2001	Sun Nov 18 01:23:11 PST 2001	Calendar.MONTH	T	T	*True*	-
11.33	11B8	Sun Nov 18 01:23:11 PST 2001	Sun Nov 18 01:23:11 PST 2001	Calendar.DATE	T	T	*True*	-
11.34	11B9	Sun Nov 18 01:23:11 PST 2001	Sun Nov 18 01:23:11 PST 2001	Calendar.DAY_OF_MONTH	T	T	*True*	-
11.35	11B10	Sun Nov 18 01:23:11 PST 2001	Sun Nov 18 01:23:11 PST 2001	Calendar.AM_PM	T	T	*True*	-
11.36	11B11	Sun Nov 18 01:23:11 PST 2001	Sun Nov 18 01:23:11 PST 2001	Calendar.HOUR	T	T	*True*	-

11.37	11B12	Sun Nov 18 01:23:11 PST 2001	Sun Nov 18 01:23:11 PST 2001	Calendar.HOUR_OF_DAY	T	T	*True*	-
11.38	11B13	Sun Nov 18 01:23:11 PST 2001	Sun Nov 18 01:23:11 PST 2001	Calendar.SECOND	T	T	*True*	-
11.39	11B14	Sun Nov 18 01:23:11 PST 2001	Sun Nov 18 01:23:11 PST 2001	Calendar.MILLISECOND	T	T	*True*	-
11.40	11B15	Sun Nov 18 01:23:11 PST 2001	Sun Nov 18 01:23:11 PST 2001	-1234	F	-	-	IllegalArgumentException
11.41	11B16	Sun Nov 18 01:23:11 PST 200, AD	Sun Nov 18 01:23:11 PST 2002, BC	Calendar.ERA	T	F	*False*	-
11.42	11B17	Sun Nov 18 01:23	Sat Oct 12 15:29:53 PST	Calendar.YEAR	T	F	*False*	-

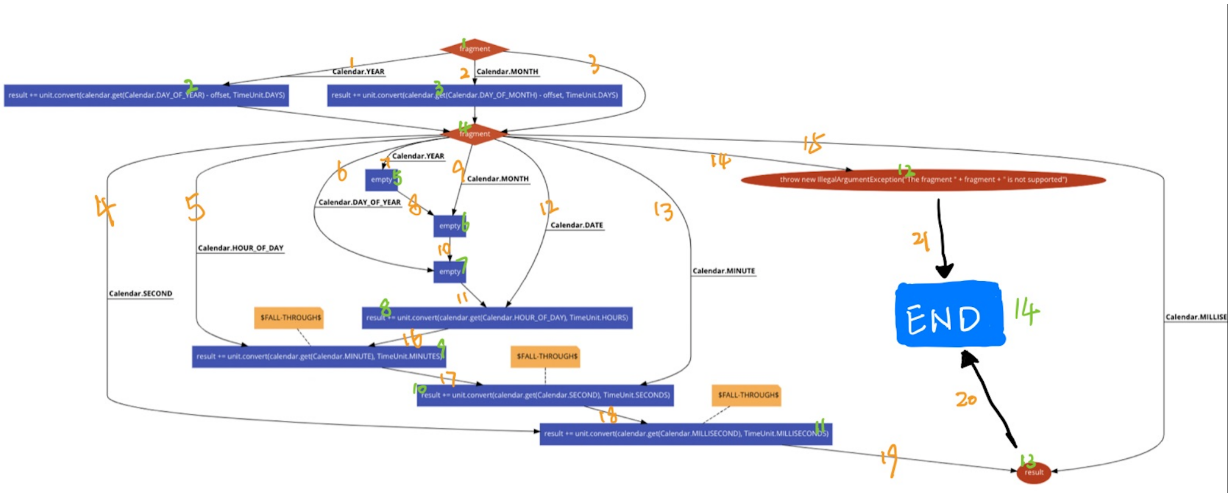
		:11 PST 2001	2002					
11.43	11B18	Sun Nov 18 01:23:11 PST 2001	Sat Oct 12 15:29:53 PST 2002	Calendar.MONTH	T	T	*True*	-
11.44	11B19	Sun Nov 18 01:23:11 PST 2001	Sat Oct 12 15:29:53 PST 2002	Calendar.DATE	T	F	*False*	-
11.45	11B20	Sun Nov 18 01:23:11 PST 2001	Sat Oct 12 15:29:53 PST 2002	Calendar.DAY_OF_MONTH	T	F	*False*	-
11.46	11B21	Sun Nov 18 01:23:11 PST 2001	Sat Oct 12 15:29:53 PST 2002	Calendar.AM_PM	T	F	*False*	-
11.47	11B22	Sun Nov 18 01:23:11 PST 2001	Sat Oct 12 15:29:53 PST 2002	Calendar.HOUR	T	F	*False*	-
11.4	11B23	Sun Nov	Sat Oct 12	Calendar.HOUR_OF_DAY	T	F	*False*	-

8		18 01:23:11 PST 2001	15:29:53 PST 2002					
11.49	11B24	Sun Nov 18 01:23:11 PST 2001	Sat Oct 12 15:29:53 PST 2002	Calendar.SECOND	T	F	*False*	-
11.50	11B25	Sun Nov 18 01:23:11 PST 2001	Sat Oct 12 15:29:53 PST 2002	Calendar.MILLISECOND	T	F	*False*	-

Structural White Box Test

getFragment

Control Flow Graph


fragment
Calendar.YEAR
Calendar.MONTH
result += unit.convert(calendar.get(Calendar.DAY_OF_YEAR) - offset, TimeUnit.DAYS)
result += unit.convert(calendar.get(Calendar.DAY_OF_MONTH) - offset, TimeUnit.DAYS)
fragment
Calendar.YEAR
empty
Calendar.MONTH
Calendar.DAY_OF_YEAR
empty
Calendar.DATE
empty
Calendar.HOUR_OF_DAY
Calendar.SECOND
Calendar.MINUTE
throw new IllegalArgumentException("The fragment " + fragment + " is not supported")
END
Calendar.MILLISE
$FALL-THROUGH$
result += unit.convert(calendar.get(Calendar.HOUR_OF_DAY), TimeUnit.HOURS)
result += unit.convert(calendar.get(Calendar.MINUTE), TimeUnit.MINUTES)
$FALL-THROUGH$
result += unit.convert(calendar.get(Calendar.SECOND), TimeUnit.SECONDS)
$FALL-THROUGH$
result += unit.convert(calendar.get(Calendar.MILLISECOND), TimeUnit.MILLISECONDS)
result
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21

$e - n + 2 = 21 - 14 + 2 = 9$

Complexity = Number of Edges - Number of Nodes + 2 = 21 - 14 + 2 = 9

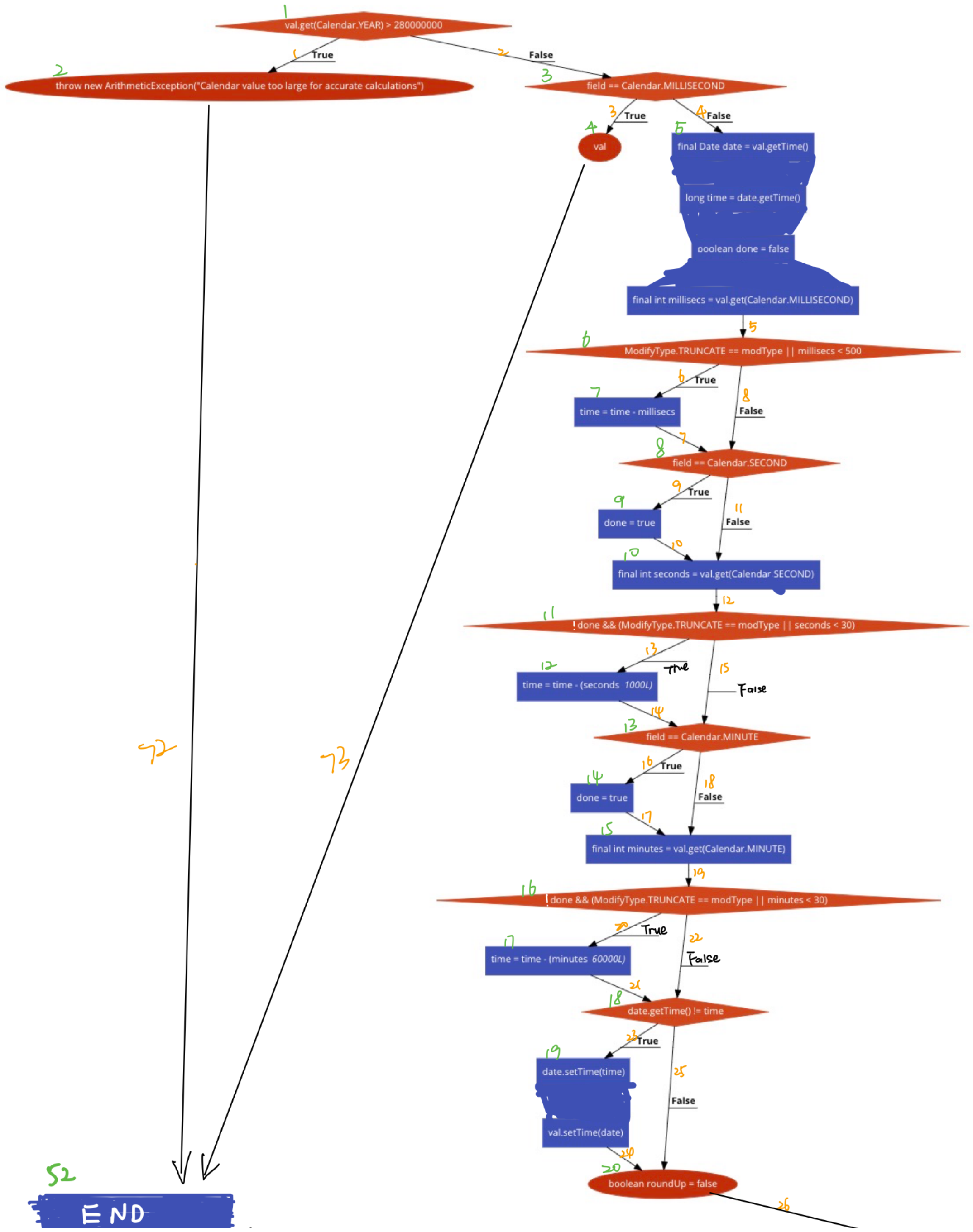
Test Cases (white-box) - Capability 10 Get Fragment

Case #	Attributes			Expected Effects	
	calendar	fragment	unit	number of days	Throw Exception
1	January 1, 2008 7:15:10.538	Calendar.YEAR	TimeUnit.MILLISECONDS	2704510538	-
2	January 1, 2008 7:15:10.538	Calendar.MONTH	TimeUnit.MILLISECONDS	26110538	-

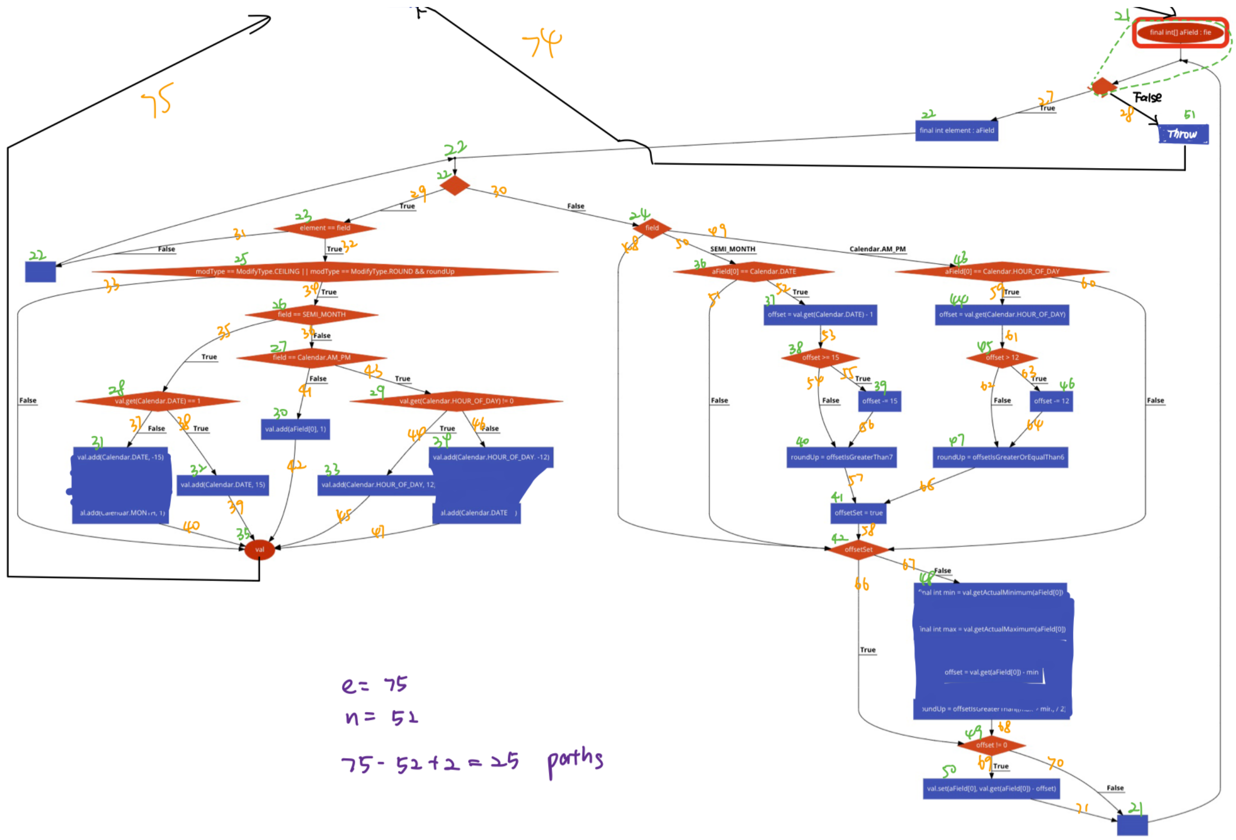
3	January 1, 2008 7:15:10.538	Calendar.SECOND	TimeUnit.MILLISECONDS	538	-
4	January 1, 2008 7:15:10.538	Calendar.HOUR_OF_DAY	TimeUnit.MILLISECONDS	910538	-
5	January 1, 2008 7:15:10.538	Calendar.DAY_OF_YEAR	TimeUnit.MILLISECONDS	26110538	-
6	January 1, 2008 7:15:10.538	Calendar.DATE	TimeUnit.MILLISECONDS	26110538	-
7	January 1, 2008 7:15:10.538	Calendar.MINUTE	TimeUnit.MILLISECONDS	10538	-
8	January 1, 2008 7:15:10.538	-9999	TimeUnit.MILLISECONDS	-	IllegalArgumentException
9	January 1, 2008 7:15:10.538	Calendar.MILLISECOND	TimeUnit.MILLISECONDS	0	-

modify

Control Flow Graph



val.get(Calendar.YEAR) > 280000000
True
False
throw new ArithmeticException("Calendar value too large for accurate calculations")
field == Calendar.MILLISECOND
True
False
val
final Date date = val.getTime()
long time = date.getTime()
boolean done = false
final int millisecs = val.get(Calendar.MILLISECOND)
ModifyType.TRUNCATE == modType || millisecs < 500
True
False
time = time - millisecs
field == Calendar.SECOND
True
False
done = true
final int seconds = val.get(Calendar.SECOND)
!done && (ModifyType.TRUNCATE == modType || seconds < 30)
True
False
time = time - (seconds 1000L)
field == Calendar.MINUTE
True
False
done = true
final int minutes = val.get(Calendar.MINUTE)
!done && (ModifyType.TRUNCATE == modType || minutes < 30)
True
False
time = time - (minutes 60000L)
date.getTime() != time
True
False
date.setTime(time)
val.setTime(date)
boolean roundUp = false
END
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
52
72
73



74
75
final int[] aField : fie
False
Throw
True
final int element : aField
False
True
element == field
False
True
modType == ModifyType.CEILING || modType == ModifyType.ROUND && roundUp
True
field == SEMI_MONTH
True
False
field == Calendar.AM_PM
False
True
val.get(Calendar.DATE) == 1
False
True
val.add(aField[0], 1)
val.get(Calendar.HOUR_OF_DAY) != 0
True
False
val.add(Calendar.DATE, -15)
al.add(Calendar.MONTH, 1)
val.add(Calendar.DATE, 15)
val.add(Calendar.HOUR_OF_DAY, 12)
val.add(Calendar.HOUR_OF_DAY. -12)
al.add(Calendar.DATE)
val
field
SEMI_MONTH
Calendar.AM_PM
aField[0] == Calendar.DATE
True
aField[0] == Calendar.HOUR_OF_DAY
True
offset = val.get(Calendar.DATE) - 1
offset = val.get(Calendar.HOUR_OF_DAY)
offset >= 15
True
False
offset -= 15
offset > 12
True
False
offset -= 12
False
False
roundUp = offsetIsGreaterThan7
roundUp = offsetIsGreaterOrEqualThan6
offsetSet = true
offsetSet
False
True
inal int min = val.getActualMinimum(aField[0])
inal int max = val.getActualMaximum(aField[0])
offset = val.get(aField[0]) - min
oundUp = offsetIsGreaterThan(max - min) / 2)
offset != 0
True
False
val.set(aField[0], val.get(aField[0]) - offset)

e= 75

n= 52

75- 52+2 = 25 paths

Cyclomatic Complexity = Number of Edges - Number of Nodes + 2 = 75 - 52 + 2 = 25

Test Cases (white-box) - Capability 7 Modify Date

The following tests are designed and implemented to improve branch coverage (with branch testing in mind).

Case #	Attributes			Expected Effects
	Calendar	Type	Field	modify()
7.1	November 1, 2002, 01:43:41.560	CEILING	DateUtils.SEMI_MONTH	November 16, 2002, 00:00:00.000

7.2	November 1, 2002, 01:43:41.560	ROUND	DateUtils.SEMI_MONTH	November 1, 2002, 00:00:00.000
7.3	November 1, 2002, 01:43:41.560	TRUNCATE	DateUtils.SEMI_MONTH	November 1, 2002, 00:00:00.000
7.4	February 3, 2002, 11:10:00.000	ROUND	Calendar.AM_PM	February 3, 2002, 12:00:00.000
7.5	February 3, 2002, 23:10:00.000	ROUND	Calendar.AM_PM	February 4, 2002, 00:00:00.000
7.6	February 3, 2002, 01:10:00.000	CEILING	Calendar.AM_PM	February 3, 2002, 12:00:00.000
7.7	February 3, 2002, 12:10:00.000	CEILING	Calendar.AM_PM	February 4, 2002, 00:00:00.000
7.8	February 3, 300000000, 13:10:00.000	ROUND	Calendar.MINUTE	Throw ArithmeticException
7.9	February 3, 300000000, 13:10:00.000	TRUNCATE	Calendar.MINUTE	Throw ArithmeticException
7.10	February 3, 300000000, 13:10:00.000	CEILING	Calendar.MINUTE	Throw ArithmeticException
7.11	February 3, 2008, 13:10:00.750	ROUND	Calendar.SECOND	February 3, 2008, 13:10:01.000
7.12	February 3, 2008, 13:10:00.750	CEILING	Calendar.SECOND	February 3, 2008, 13:10:01.000

Results and Interpretation

Report and interpret results here after converting test cases to executable tests and running them on your project. Comment on how effective your strategy was, based on native faults and mutations found, and based on coverage analysis. Was it as effective as you had hoped? If not, why do you think it was not?

No, it was not as effective as we had hoped.

In terms of native faults, our team's tests did not identify any native faults. This is primarily because Apache Commons Lang is a library that receives regular patching and maintenance.

Regarding the mutations found, our team's tests detected 4 out of 10 mutations, whereas the native tests found 2 out of 10 mutations. While our team's tests outperformed the native tests in this regard, they still fell short of our expectations. One critical reason was the low testability of the target class. For instance, the rival team created the #6 mutation by removing one of the checks for a NullPointerException in the method 'add(final Date date, final int calendarField, final int amount)'. However, our tests didn't detect this mutation because the NullPointerException was still checked and an exception was thrown in another part of the same method. The presence of duplication and trivial conditions significantly contributed to the low testability of the target codebase, making it impossible to detect certain mutations.

In terms of the number of test cases, our team executed a total of 286 tests, whereas the native tests only consisted of 63.

Our more comprehensive test cases resulted in better coverage of the target codebase. Compared to the native tests, our team's tests increased line coverage from 84% to 95% and branch coverage from 80% to 85%. This enhanced coverage was achieved by developing a comprehensive black-box test plan and applying white-box strategies like Cyclomatic Testing and Branch Testing.