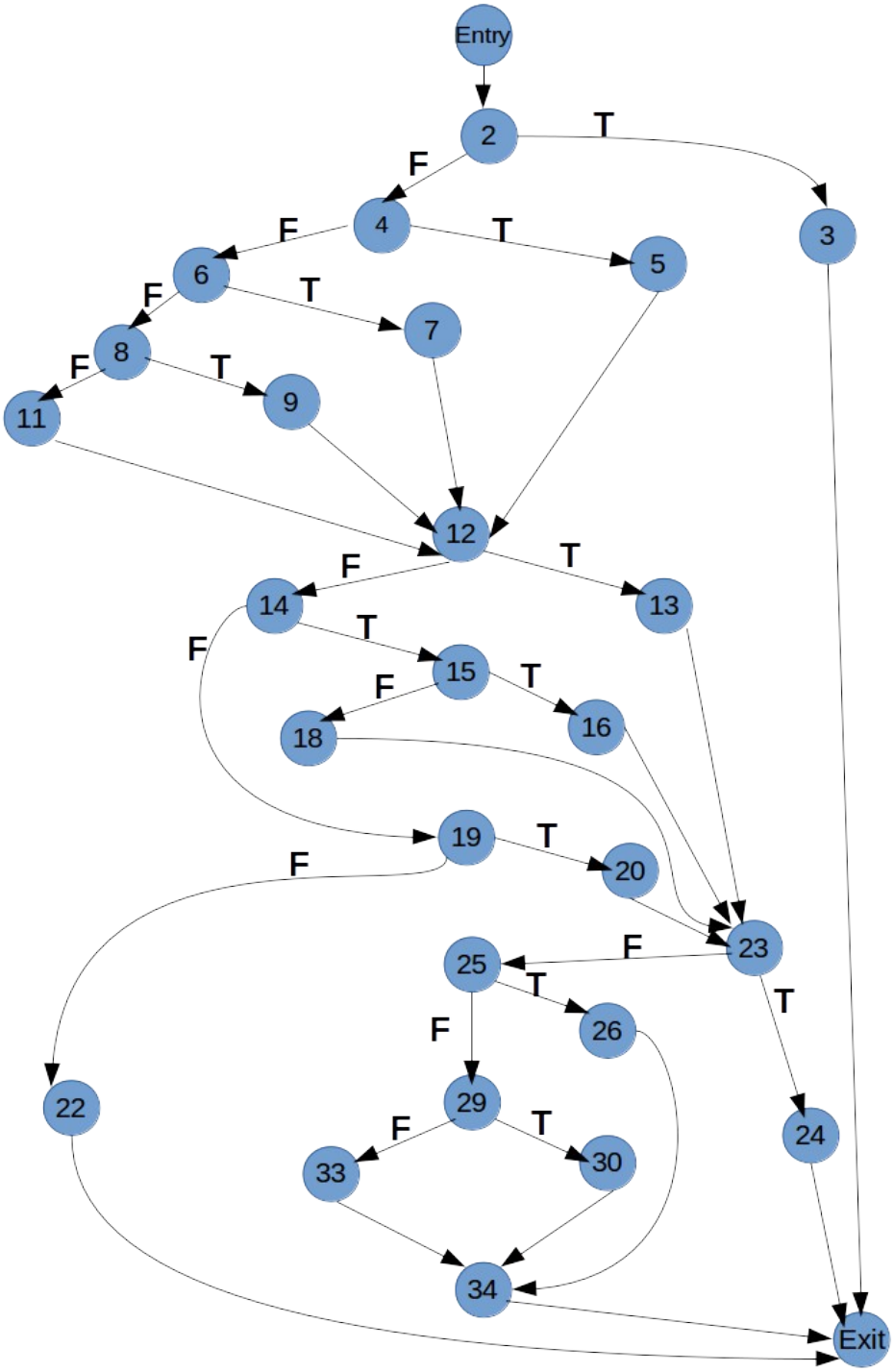
Control Flow Test Design

Question 1



Question 2

For 100% statement (instruction) coverage, our test suite must hit each instruction at least once. We need to determine a minimal number of paths so that each instruction in can be found on one path at least. When determining paths, we must make sure to keep track of the decisions conditions and be particularly careful about conditions which evaluation depend on other conditions (dependent conditions). Dependent conditions occur when a variable checked in a condition is set based on a previous decision.

For instance, in order for an execution to go through node 18, condition `leap_year` at node 15 must evaluate to false which implies that the execution of node 11 (hence the conditions at nodes 8, 6, 4, 2 false) or node 7 (hence the conditions at node 6 true and nodes 4, 2 false).

We can achieve 100% statement with the Paths:

1. Entry – 2 – 3 – Exit
2. Entry – 2 – 4 – 6 – 8 – 11 – 12 – 14 – 19 – 20 – 23 – 25 – 29 – 33 – 34 – Exit
3. Entry – 2 – 4 – 6 – 8 – 9 – 12 – 13 – 23 – 25 – 29 – 30 – 34 – Exit
4. Entry – 2 – 4 – 6 – 7 – 12 – 14 – 15 – 18 – 23 – 24 – Exit
5. Entry – 2 – 4 – 5 – 12 – 14 – 19 – 22 – Exit
6. Entry – 2 – 4 – 5 – 12 – 14 – 15 - 16 – 23 – 25 – 26 – 34 - Exit

We need to sensitize each path to find input values for their execution. This involves for each path: (1) gathering all the conditions that are needed at each decision point along the path and (2) solving the predicate obtained from combining all these conditions. As much as possible, data values should be picked on the boundaries of the conditions.

Path	**Predicates (Conditions)**	**Sample Input Data**
Entry – 2 – 3 – Exit	year < 0	year=-1, month=DC, day=DC

Entry – 2 – 4 – 6 – 8 – 11 – 12 – 14 – 19 – 20 – 23 – 25 – 29 – 33 – 34 – Exit	`year >= 0 &&`
	`year % 400 != 0 &&`
	`year % 100 != 0 &&`
	`year % 4 != 0 &&`
	`month not in (1, 3, 5, 7, 8, 10, 12)`
	`month != 2`
	month in `4, 6,9, 11) &&`
	day >= 1 && day <= 30 &&
	day >= 30 &&
	month != 12 &&
		year=1989, month=4, day=30

Entry – 2 – 4 – 6 – 8 – 9 – 12 – 13 – 23 – 25 – 29 – 30 – 34 – Exit	`year >= 0 &&`
	`year % 400 != 0 &&`
	`year % 100 != 0 &&`
	`year % 4 = 0 &&`
	`month` **`in`** `(1, 3, 5, 7, 8, 10, 12) &&`
	`day >= 1 && day <= 31 &&`
	`day >= 31 &&`
	`month = 12`
		year=4, month=12, day = 31

Entry – 2 – 4 – 6 – 7 – 12 – 14 – 15 – 18 – 23 – 24 – Exit	`year >= 0 &&`
	`year % 400 != 0 &&`
	`year % 100 = 0 &&`
	`month not in (1, 3, 5, 7, 8, 10, 12)`
	`month = 2 &&`
	`day < 1 && day > 28`
		year=1900, month=2, day=29

Path	Predicates (Conditions)	Sample Input Data

Entry – 2 – 4 – 5 – 12 – 14 – 19 – 22 – Exit	`year >= 0 &&`
	`year % 400 = 0 &&`
	`month not in (1, 3, 5, 7, 8, 10, 12)`
	`month != 2`
	`month not in (4, 6,9, 11)`
		year=400, month=13, day=DC

Entry – 2 – 4 – 5 – 12 – 14 – 15 - 16 – 23 – 25 – 26 – 34 - Exit	`year >= 0 &&`
	`year % 400 = 0 &&`
	`month = 2 &&`
	`leap_year`
	`day >= 1 && day <= 29 &&`
	`day < 29`
		year=2000, month=2,day=28

Notes:

- Path 2, variable month_lenght is equal to 30 since line 20 is executed
- Path 3, variable month_lenght is equal to 31 since line 13 is executed
- Path 4, variable month_lenght is equal to 28 since line 18 is executed
- Path 6, variable month_lenght is equal to 29 since line 16 is executed

At minimum, one test cases should be created for each path. More test cases could be added to ensure further coverage of boundaries.

Test Case	Input	Expected Output	Path Covered
1	year=-1, month=6 day=15	ValueError(“Wrong value for year”)	Entry – 2 – 3 – Exit
2	year=1989, month=4, day=30	1989-5-1	Entry – 2 – 4 – 6 – 8 – 11 – 12 – 14 – 19 – 20 – 23 – 25 – 29 – 33 – 34 – Exit
3	year=4, month=12, day = 31	5-1-1	Entry – 2 – 4 – 6 – 8 – 9 – 12 – 13 – 23 – 25 – 29 – 30 – 34 – Exit
4	year=1900, month=2, day=29	ValueError("Wrong value for day")	Entry – 2 – 4 – 6 – 7 – 12 – 14 – 15 – 18 – 23 – 24 – Exit
5	year=400, month=13, day=14	ValueError("Wrong value for month")	Entry – 2 – 4 – 5 – 12 – 14 – 19 – 22 – Exit
6	year=2000, month=2,day=28	2000-2-29	Entry – 2 – 4 – 5 – 12 – 14 – 15 - 16 – 23 – 25 – 26 – 34 - Exit

Question 3

The test suite developed for question 2 satisfies 100% branch coverage.

Question 4

In order to achieve 100% branch/condition coverage, we need to ensure each condition is evaluated at least once to *true* and *false* in addition to exercising each decision.

For simplicity, let start with the test suite previously developed for question2. Each decision except the decision at line 23 in the code under test, consists in a single condition. Therefore, branch/condition coverage is satisfied for these decisions. In order to obtain a test suite that satisfies 100% branch/condition coverage, we need only to ensure all conditions at line 23 are covered. Considering the test cases which execute line 23:

- Test case 2 (year=1989, month=4, day=30) evaluates *day < 1 to* ***false*** and *day > month_lenght to* ***false***
- Test case 3 (year=4, month=12, day = 31) evaluates *day < 1 to* ***false*** and *day > month_lenght to* ***false***
- Test case 4 (year=1900, month=2, day=29) evaluates *day < 1 to* ***false*** and *day > month_lenght* to ***true***
- Test case 6 (year=2000, month=2,day=28) evaluates *day < 1 to* ***false*** and *day > month_lenght* to ***false***

We only need an additional test case that evaluates *day < 1 to* ***true.***

Test Case	Input	Expected Output	Path Covered	Conditions Covered
1	year=-1, month=6 day=15	ValueError("Wrong value for year")	Entry – 2 – 3 – Exit	`year < 0`

2	year=1989, month=4, day=30	1989-5-1	Entry – 2 – 4 – 6 – 8 – 11 – 12 – 14 – 19 – 20 – 23 – 25 – 29 – 33 – 34 – Exit	`year >= 0`
				`year % 400 != 0`
				`year % 100 != 0`
				`year % 4 != 0`
				`month not in (1, 3, 5, 7, 8, 10, 12)`
				`month != 2`
				`month in (4, 6,9, 11)`
				`day >= 1`
				`day <= month_length`
				`day >= month_length`
				`month != 12`

3	year=4, month=12, day = 31	5-1-1	Entry – 2 – 4 – 6 – 8 – 9 – 12 – 13 – 23 – 25 – 29 – 30 – 34 – Exit	`year >= 0`
				`year % 400 != 0`
				`year % 100 != 0`
				`year % 4 == 0`
				`month in (1, 3, 5, 7, 8, 10, 12)`
				`day >= 1`
				`day <= month_length`
				`day >= month_length`
				`month == 12`

4	year=1900, month=2, day=29	ValueError("Wrong value for day")	Entry – 2 – 4 – 6 – 7 – 12 – 14 – 15 – 18 – 23 – 24 – Exit	`year >= 0`
				`year % 400 != 0`
				`year % 100 == 0`
				`month not in (1, 3, 5, 7, 8, 10, 12)`
				`not leap_year`
				`day >= 1`
				`day > month_length`

5	year=400, month=13, day=14	ValueError("Wrong value for month")	Entry – 2 – 4 – 5 – 12 – 14 – 19 – 22 – Exit	`year >= 0`
				`year % 400 == 0`
				`month not in (1, 3, 5, 7, 8, 10, 12)`
				`month != 2`
				`month not in (4, 6,9, 11)`

6	year=2000, month=2,day=28	2000-2-29	Entry – 2 – 4 – 5 – 12 – 14 – 15 - 16 – 23 – 25 – 26 – 34 - Exit	`year >= 0`
				`year % 400 == 0`
				`month not in (1, 3, 5, 7, 8, 10, 12)`
				`month == 2`
				`leap_year`
				`day >= 1`
				`day <= month_length`
				`day >= month_length`

7	year=1600, month=5, day=0	ValueError("Wrong value for day")	Entry – 2 – 4 – 5 – 12 – 13 – 23 – 24 – Exit	`year >= 0`
				`year % 400 == 0`
				`month in (1, 3, 5, 7, 8, 10, 12)`
				`day < 1`
				`day <= month_length`