|  |  |  |  |
| --- | --- | --- | --- |
| **Date.isValid():** | | | |
| **Test Case #** | **Requirement** | **Test Description and Input Data** | **Expected output** |
| 1 | The year should not be before 1900 | * Instance of Date where the month and day are valid, but the date is < 1900 * “5/22/1883” | False |
| 2 | The month should be between 1 and 12 | * Instance of Date where the day < 28, the year is valid, but the month is > 12 * “16/3/1984” | False |
| 3 | The month should be between 1 and 12 | * Instance of Date where day < 28, the year is > 1900, but the month is < 1 * “0/15/2002” | False |
| 4 | If the month is Feb, and it is not a leap year, the day should be between 1 and 28 | * Instance of Date where month = 2, day > 28, and year is not a leap year * “2/29/1993” | False |
| 5 | If the month is Feb, and it is not a leap year, the day should be between 1 and 28 | * Instance of Date where month = 2, day < 1, and year is not a leap year * “2/-1/1993” | False |
| 6 | If the month is Feb, and it is a leap year, the day of month should be between 1 and 29 | * Instance of Date where month = 2, day > 29, and year is a leap year * “2/31/1984” | False |
| 7 | If the month is April, June, September, or November, the day of month should be between 1 and 30 | * Instance of Date where month = 4, 6, 9, or 11, day > 30, and year is valid * “6/31/2000” | False |
| 8 | If the month is April, June, September, or November, the day of month should be between 1 and 30 | * Instance of Date where month = 4, 6, 9, or 11, day < 1, and year is valid * “6/0/2000” | False |
| 9 | If the month is January, March, May, July, August, October, or December, the day on month should be between 1 and 31 | * Instance of Date where month = 1, 3, 5, 7, 8, 10, 12, day > 31, and year is valid * “12/500/1994” | False |
| 10 | If the month is January, March, May, July, August, October, or December, the day on month should be between 1 and 31 | * Instance of Date where month = 1, 3, 5, 7, 8, 10, 12, day < 1, and year is valid * “12/0/1994” | False |
| 11 | If all conditions above are met, the date is valid | * Instance of Date where month, day, and year are valid * “3/3/2019” | True |

|  |  |  |  |
| --- | --- | --- | --- |
| **Timeslot.compateTo():** | | | |
| **Test Case #** | **Requirement** | **Test Description and Input Data** | **Expected output** |
| 1 | If the year of the first Timeslot comes before the second one, the code will return -1 | * Two instances of Timeslot where the times, days, and months are the same, but the year of the first one comes before the second one * “2/20/1984 4:20” & “2/20/1985 4:20” | -1 |
| 2 | If the year of the first Timeslot comes after the second one, the code will return 1 | * Two instances of Timeslot where the times, days, and months are the same, but the year of the first one comes after the second one * “2/20/1983 4:20” & “2/20/1984 4:20” | 1 |
| 3 | If the years are the same and the month of the first Timeslot comes before the second one, the code will return -1 | * Two instances of Timeslot where the times, days, and years are the same, but the month of the first one comes before the second one * “2/20/1984 4:20” & “3/20/1984 4:20” | -1 |
| 4 | If the years are the same and the month of the first Timeslot comes after the second one, the code will return 1 | * Two instances of Timeslot where the times, days, and years are the same, but the month of the first one comes after the second one * “3/20/1984 4:20” & “2/20/1985 4:20” | 1 |
| 5 | If the years and months are the same and the day of the first Timeslot comes before the second one, the code will return -1 | * Two instances of Timeslot where the times, months, and years are the same, but the day of the first one comes before the second one * “2/19/1984 4:20” & “2/20/1984 4:20” | -1 |
| 6 | If the years and months are the same and the day of the first Timeslot comes after the second one, the code will return 1 | * Two instances of Timeslot where the times, months, and years are the same, but the day of the first one comes after the second one * “2/20/1984 4:20” & -“2/19/1984 4:20” | 1 |
| 7 | If the dates are the same and the hour of the first Timeslot comes before the second one, the code will return -1 | * Two instances of Timeslot where the dates and minutes are the same, but the hour of the first one comes before the second one * “2/20/1984 3:20” & “2/20/1984 4:20” | -1 |
| 8 | If the dates are the same and the hour of the first Timeslot comes after the second one, the code will return 1 | * Two instances of Timeslot where the dates and minutes are the same, but the hour of the first one comes after the second one * “2/20/1984 4:20” & “2/20/1984 3:20” | 1 |
| 9 | If the dates and hours are the same and the minute of the first Timeslot comes before the second one, the code will return -1 | * Two instances of Timeslot where the dates and hours are the same, but the minute of the first one comes before the second one * “2/20/1984 4:19” & “2/20/1984 4:20” | -1 |
| 10 | If the dates and hours are the same and the minute of the first Timeslot comes after the second one, the code will return 1 | * Two instances of Timeslot where the dates and hours are the same, but the minute of the first one comes after the second one * “2/20/1984 4:20” & “2/20/1984 4:19” | 1 |
| 11 | If the dates and times are the same, the code will return 0 | * Two instances of Timeslot with the same dates and times * “2/20/1984 4:20” & “2/20/1984 4:20” | 0 |