Q1

William HU ZIHAO

2024-09-24

Q1

1) Leap Year

```
is_leap_year <- function(year) {
  if (!is.numeric(year) || year != as.integer(year)) {
    message("Invalid input: The year must be an integer or a numeric in integer format.")
  }
  else{
    ((year %% 4 == 0 && year %% 100 != 0) || (year %% 400 == 0))
  }

}
```

```
is_leap_year(2012)
```

```
## [1] TRUE
```

```
is_leap_year(2023)
```

```
## [1] FALSE
```

```
is_leap_year(2024.000)
```

```
## [1] TRUE
```

```
is_leap_year(2024.2910)
```

```
## Invalid input: The year must be an integer or a numeric in integer format.
```

```
is_leap_year("2024")
```

```
## Invalid input: The year must be an integer or a numeric in integer format.
```

2) Weekday of 28th

```
weekdays_28th <- function(year) {
    if (!is.numeric(year) || year != as.integer(year)) {
    message("Invalid input: The year must be an integer or a numeric in integer format.")
  }
  else{
      months <- c(1:12)
      date_strs <- as.Date(paste0(year, "-", months, "-28"))
      weekdays <- sapply(date_strs, (\(x) {format(x, "%a")}))

      for (month in months)
```

```
      {
       cat(paste0(year,"-",month,"-28 is ", weekdays[month]),"\n")
      }
  }
}
```

```
weekdays_28th(2024)
```

```
## 2024-1-28 is Sun
## 2024-2-28 is Wed
## 2024-3-28 is Thu
## 2024-4-28 is Sun
## 2024-5-28 is Tue
## 2024-6-28 is Fri
## 2024-7-28 is Sun
## 2024-8-28 is Wed
## 2024-9-28 is Sat
## 2024-10-28 is Mon
## 2024-11-28 is Thu
## 2024-12-28 is Sat
```

```
cat("\n")
```

```
weekdays_28th(2031.9009021)
```

```
## Invalid input: The year must be an integer or a numeric in integer format.
```

```
cat("\n")
```

```
weekdays_28th("2024")
```

```
## Invalid input: The year must be an integer or a numeric in integer format.
```

```
cat("\n")
```

```
weekdays_28th(1923)
```

```
## 1923-1-28 is Sun
## 1923-2-28 is Wed
## 1923-3-28 is Wed
## 1923-4-28 is Sat
## 1923-5-28 is Mon
## 1923-6-28 is Thu
## 1923-7-28 is Sat
## 1923-8-28 is Tue
## 1923-9-28 is Fri
## 1923-10-28 is Sun
## 1923-11-28 is Wed
## 1923-12-28 is Fri
```

```
cat("\n")
```

```
weekdays_28th(2923)
```

```
## 2923-1-28 is Thu
## 2923-2-28 is Sun
## 2923-3-28 is Sun
## 2923-4-28 is Wed
```

```
## 2923-5-28 is Fri
## 2923-6-28 is Mon
## 2923-7-28 is Wed
## 2923-8-28 is Sat
## 2923-9-28 is Tue
## 2923-10-28 is Thu
## 2923-11-28 is Sun
## 2923-12-28 is Tue
```

3) Working Days

```
library(bizdays)
```

```
##
## Attaching package: 'bizdays'
## The following object is masked from 'package:stats':
##
##     offset
```

```
# Final function for counting working days
count_working_days <- function(year) {
  if (!is.numeric(year) || year != as.integer(year)) {
    message("Invalid input: The year must be an integer or a numeric in integer format.")
    return()
  } else {
    start_date <- as.Date(paste0(year, "-01-01"))
    end_date <- as.Date(paste0(year, "-12-31"))

    working_days <- 0
    current_date <- start_date

    while (current_date <= end_date) {
      weekday <- format(current_date, "%a")
      if (weekday != "Sat" && weekday != "Sun") {
        working_days <- working_days + 1
      }
      current_date <- current_date + 1
    }

    # Count working days using bizdays
    create.calendar("MyCalendar", holidays = NULL, weekdays = c("saturday", "sunday"), start.date = sta

    working_days_bizdays <- bizdays(start_date, bizdays::adjust.next(end_date, "MyCalendar"), "MyCalend

    # If end_date is a weekday, add 1
    if (format(end_date, "%a") != "Sat" && format(end_date, "%a") != "Sun") {
      working_days_bizdays <- working_days_bizdays + 1
    }

    # Return both manual and bizdays results
    return(list(manual_count = working_days, bizdays_count = working_days_bizdays))
  }
}
```

```r
# TEST FUNCTION GENERATED BY ChatGPT

# Testing loop for a range of valid inputs (years)
test_years <- 1900:2900  # Range of years to test
discrepancies <- list()  # To store any discrepancies between manual and bizdays counts

for (year in test_years) {
  result <- count_working_days(year)

  if (result$manual_count != result$bizdays_count) {
    # If there is a discrepancy, store the year and the respective counts
    discrepancies[[as.character(year)]] <- result
    cat("Discrepancy found for year:", year, "\n")
  }
}

# Summary of results
if (length(discrepancies) == 0) {
  cat("All tests passed! No discrepancies found.\n")
} else {
  cat("Discrepancies found in the following years:\n")
  print(discrepancies)
}
```

```
## All tests passed! No discrepancies found.
```