```
/**
 * Prints the calendars of all the years in the 20th century.
 */
public class Calendar {
   // Starting the calendar on 1/1/1900
   static int dayOfMonth = 1;
   static int month = 1;
   static int year = 1900;
   static int dayOfWeek = 2;     // 1.1.1900 was a Monday
   static int nDaysInMonth = 31; // Number of days in January


   /**
    * Prints the calendars of all the years in the 20th century. Also prints the
    * number of Sundays that occurred on the first day of the month during this period.
    */
   public static void main(String args[]) {

      // Prints each date dd/mm/yyyy in a separate line. If the day is a Sunday, prints "Sunday".
      // getting the year of the calendar.

      int chosenYear = Integer.parseInt(args[0]);
      // Write the necessary initialization code, and replace the condition
      // of the while loop with the necessary condition
      while (year <= chosenYear) {

         advance(chosenYear);

         }
         }
```

```
// Advances the date (day, month, year) and the day-of-the-week.
// If the month changes, sets the number of days in this month.
// Side effects: changes the static variables dayOfMonth, month, year, dayOfWeek, nDaysInMonth.
public static void advance(int chosenYear) {
  // cheking if the year is the chosen year and printing it
  if (year == chosenYear){
  printDate();
  }

  dayOfWeek = (dayOfWeek + 1) % 7; // Move to the next day of the week

  // Check if the month needs to be advanced
    dayOfWeek = (dayOfWeek + 1) % 7; // Move to the next day of the week
    dayOfMonth++; // Move to the next day of the month

    // Check if the month needs to be advanced
    if (dayOfMonth > nDaysInMonth) {
      dayOfMonth = 1; // Reset the day of the month
      month++; // Move to the next month
    }
      // Check if the year needs to be advanced
      if (month > 12) {
        month = 1; // Reset the month
        year++; // Move to the next year
        nDaysInMonth = nDaysInMonth(month, year);// Set the number of days in the new
month
      }
      nDaysInMonth = nDaysInMonth(month, year);
```

```
        }



    // Prints the current date in the format dd/mm/yyyy and the day of the week if it is a Sunday
    // Prints the current date in the format dd/mm/yyyy and the day of the week if it is a Sunday
    public static void printDate(){
        System.out.print(dayOfMonth + "/" + month + "/" + year);
        if (dayOfWeek == 0) {
            System.out.print(" Sunday");
        }
        System.out.println();
    }



    // Returns true if the given year is a leap year, false otherwise.
    // Returns true if the given year is a leap year, false otherwise.
public static boolean isLeapYear(int year) {
    return ((year % 400) == 0) || (((year % 4) == 0) && ((year % 100) != 0));
}

    // Returns the number of days in the given month and year.
    // April, June, September, and November have 30 days each.
    // February has 28 days in a common year, and 29 days in a leap year.
    // All the other months have 31 days.
    public static int nDaysInMonth(int month, int year) {
        if ((month == 1) || (month == 3) || (month == 5) || (month == 7) || (month == 8) ||
                (month == 10) || (month == 12)) {
            return 31;
        } else if (month == 2) {
            if (isLeapYear(year)) {
                return 29;
            } else {
                return 28;
            }
        } else if ((month == 4) || (month == 6) || (month == 9) || (month == 11)) {
            return 30;
        } else {
            return 31;
        }
    }
}
```