

# Leap Year

This is a **pair** exercise and must be completed in your **tutorial** or **lab** with your partner.

Dates can be tricky to work with on computers for many reasons. One thing that makes dates tricky is leap years. In the lab this week, you will make a program to tell a user whether or not a given year is a leap year.

## The Gregorian Calendar

Over time, the calendar we use has changed a lot. Currently, we use a system called the [Gregorian Calendar](#) and it was introduced in 1582. Your task is to make a program called `leapYear.c` that will read in a year as an integer and display whether or not the year is a leap year.

## Leap Year Algorithm

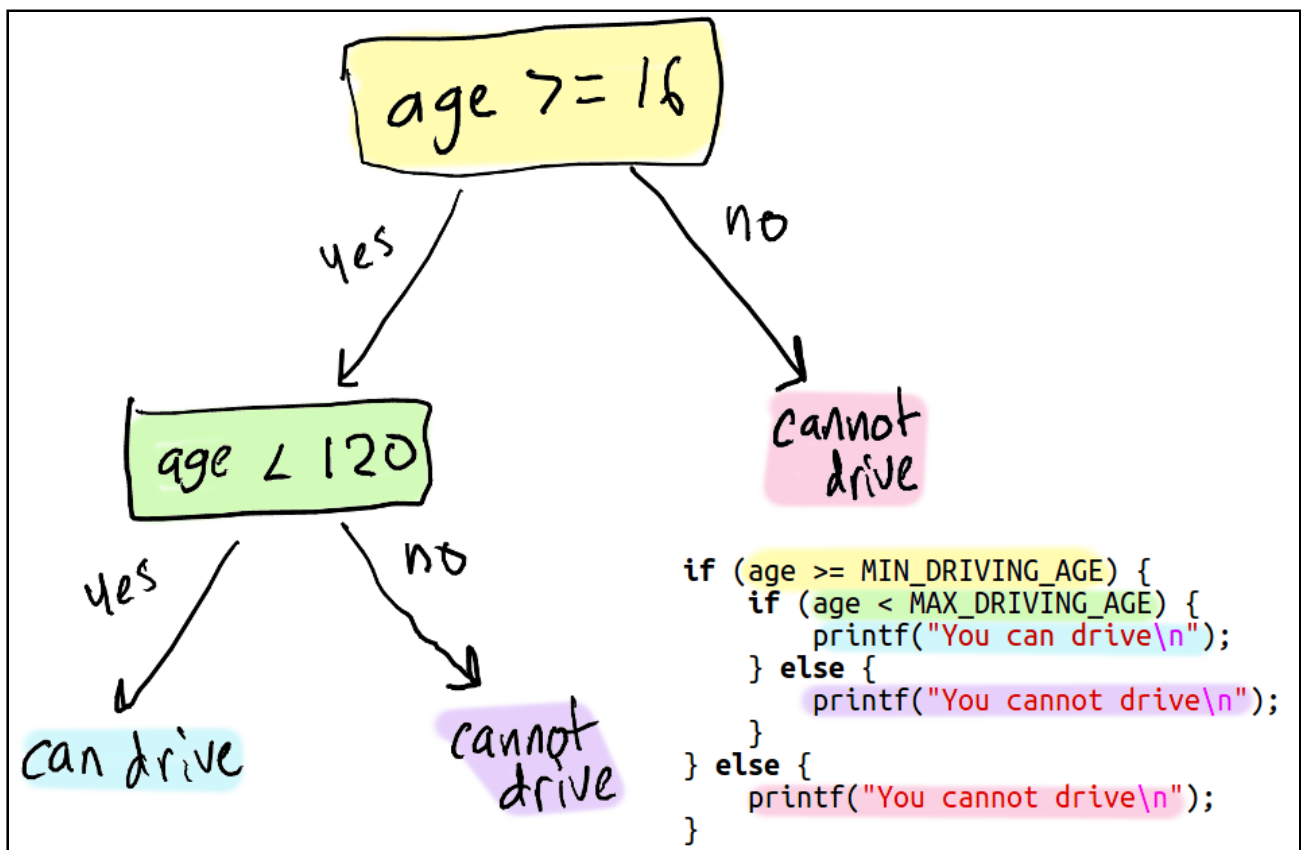
The algorithm to determine if a year is a leap year is as follows:

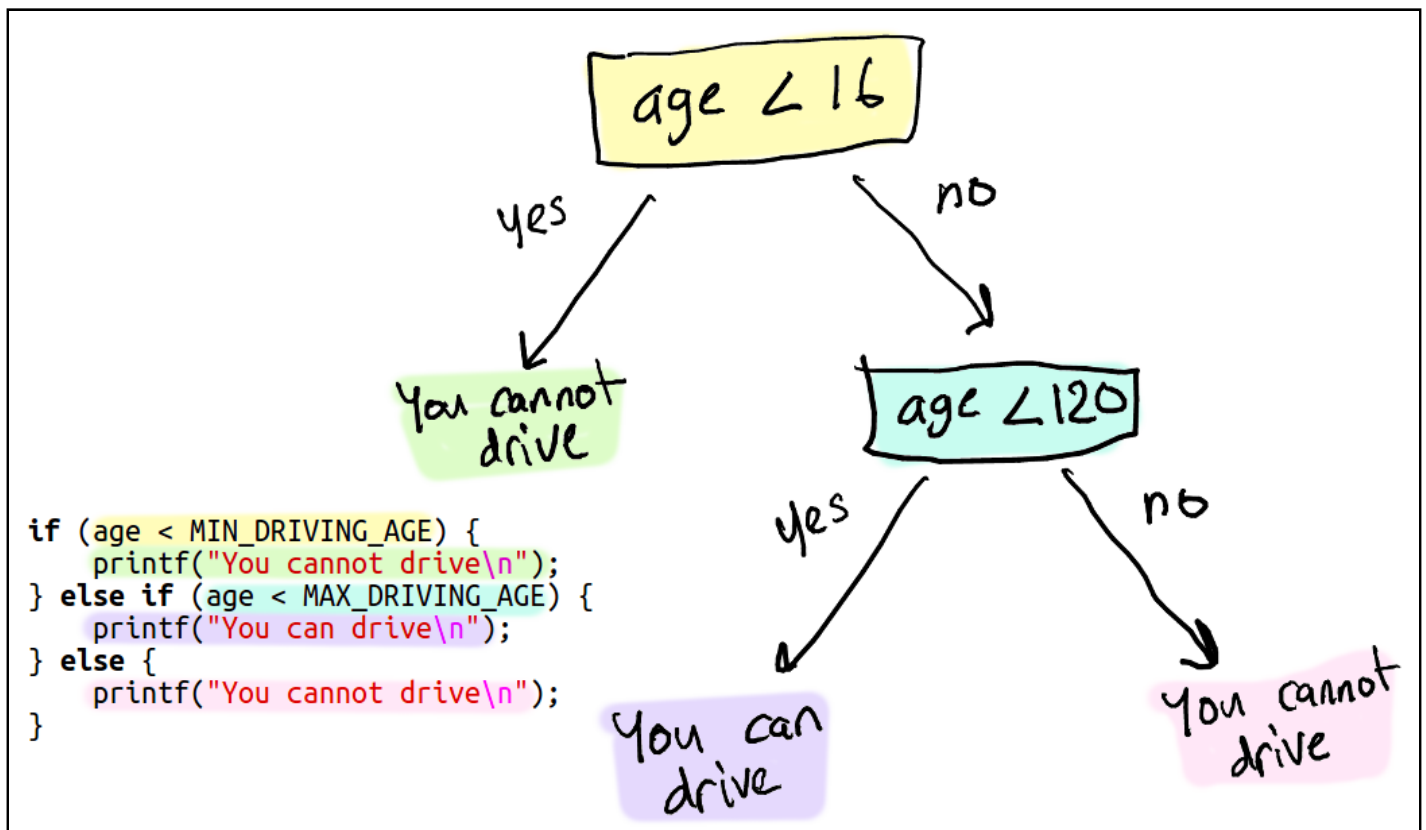
Every year that is **exactly divisible by four is a leap year**, **except** for years that are **exactly divisible by 100**, but these centurial years are leap years, if they are **exactly divisible by 400**.

For example, the years 1700, 1800, and 1900 were not leap years, but the years 1600 and 2000 were.

During your tutorial, divide into groups of 4 and discuss how you might draw the *control flow* for this algorithm. Once each group has had a chance to work out their solution, get someone from each group to write it on the whiteboard. As a class, compare each solution and see which ones are equivalent and which ones have bugs.

Here are some examples of a control flow diagrams from the lecture:





## Your Program

During the lab, create the following program.

You should ask for the user to enter a year using the message `Enter a year after 1582: .`

If the year entered is **before** 1582, display the message `n is before 1582`. If the year entered is **after** 1582, you must display the message `n is a leap year`. If the year is a leap year, or the message `n is not a leap year`. If the year is not a leap year.

## Modulus

The *modulus* operator in C (`%`) will give the *modulus* of two numbers, or the remainder when the first number is divided by the second number. For example, `5 % 2` is `1` and `10 % 5` is `0`. This also means that if we have two numbers, `a` and `b`, and `a` is divisible by `b`, then `a % b` will be 0.

## Some Examples

Enter a year after 1582: 12  
12 is before 1582.

Enter a year after 1582: 2000  
2000 is a leap year.

Enter a year after 1582: 2004  
2004 is a leap year.

Enter a year after 1582: 2001  
2001 is not a leap year.

Enter a year after 1582: 2100  
2100 is not a leap year.

To run some simple automated tests:

```
$ 1511 autotest leapYear
```

To run Styl-o-matic:

```
$ 1511 stylomatic leapYear.c  
Looks good!
```

You'll get advice if you need to make changes to your code.

Submit your work with the *give* command, like so:

```
$ give cs1511 wk02_leapYear
```

Or, if you are working from home, upload the relevant file(s) to the `wk02_leapYear` activity on [Give Online](#).