# First programming exercise: Admission

## The exercise

For this assignment, you're going to implement a small user interface which asks a user to input some personal data and answer a few questions as verification.

A user will first enter his/her birthyear as a number (integer), then the month (as integer). Then he/she enters the day (as integer). The program will calculate the users age, both in years and months. Both are shown to the user.
Age in years and age in months are calculated separately (when born on the 31st of some month, you only become one month older once an entire month has passed, not when it starts). You may define a reference point beforehand for the current date (specify that the current date is September 24th 2010 was X years, Y months and Z days ago for example), but you may also use ctime to get the current date. The program should be able to handle any reference date from present to 2100.

Potential students who are less than 10 years old (10th birthday has not passed yet) or alumni who are more than 100 years old (celebrated 101st birthday) are refused by the system. Most of the time the birthyear will be sufficient to tell whether users should be refused, but note that sometimes you will only be able to check this using the month or day!

Now, the user must enter his actual birthday (Sunday, Monday, etc) as a verification. If it's wrong, the user is refused and the program stops. The answer must be entered using a single letter (char). Think of a solution for the Tuesday/Thursday, Saturday/Sunday problem! You are not allowed to use ctime to calculate the correct day, the program should be able to determine the correct day of the week itself. Again, use a reference date: For example that January 1st 1901 was a tuesday. You are not allowed to use the Doomsday algorithm (http://en.wikipedia.org/wiki/Doomsday_(weekday)). From 1901 to 2099, a year is a leap year if it's divisible by 4.

Admission consists of three questions:
First, the user must prove he/she is human by entering the product of his/her birthday and month of birth. If the inputted answer is wrong, the program exits.
Then, a multiple choice question (A/B/C/D) must be answered. If the answer is wrong, the program exits as well. Users that are 18 years or younger must answer a different question than older users. You may think of your own questions, but make them AI-related! Be sure to handle both capital and non-capital input.
Lastly, the user must enter the product of two random numbers. These numbers can have up to 3 non-decimals and up to 3 decimals. You can generate 'random' numbers between 0 and 999 using, for example, *rand() % 1000.* Be sure to handle conversion from integer to double correctly. The user may be wrong by a prespecified margin MAXERROR. On some systems,

you will need to add *#include <cstdlib>* in the top of your program. You can specify the seed for the randomgenerator using (for example) *srand( 123 )*.

## Notes

If the user enters a non-existing month, for example -8, or a year such as 4242, the program exits with the note that the current input is impossible/illegal. Same goes for nonexistent days of course. We assume that the user does not try to break the system, so he/she does not enter characters or extremely large numbers. The questions shown afterwards should be clear and should show the user how to input the answer.
Every program should print at the start of a run who the authors of the system are, their student number, and their current education (e.g. BSc AI year 2). It should also show which assignment this is, what is expected of the user, and the date of implementation. Try to format this info block so that it is easy to read, and so that it can be used in the next assignments as well. In the source code, authors should also specify their names, student number, and additional info regarding the system and compiler used in comments in the top of the program.

IMPORTANT: Be sure to use comments, whitelines and indentation correctly. Keep your code easy to read, and your comments useful. Use sensible variable names.

**Bad (bad variable name, useless comment):**
*int x = 3;  // here, I assign value 3 to variable x*
**Better (good variable names, comment not that useful):**
*int leap_years = years % 4;  // every 4th year is a leap year*
**Good (good variable names, useful comment):**
*int non_decimals = rand() % 1000;  // generate a number between 0 and 999*

For this assignment, you do not have to use functions, arrays, while or for loops (though of course you may do this if you want to). Use only headers iostream, ctime, string and/or cstdlib. Your program should be approximately 200 - 300 lines. Shorter does not necessarily mean better, but if you're passing line 300, this should at least alarm you somewhat.