C++ Programming Methods

Assignment 1, Date Problems

Bas Terwijn <b.terwijn@uva.nl>

In this assignment we ask the user for information about his/her day of birth and process this. Here we practice using basic data types, control flow statements and functions.

Age

Ask the user for his/her birth year (range 1900-2100), birth month (range 1-12), and birth day (range 1-31, or depending on month and year) and check if it is in the valid range. Design and implement an algorithm for the functions:

```
  int yearsOld(int currentYear,int currentMonth,int currentDay,
               int birthYear,int birthMonth,int birthDay);

  int monthsOld(int currentYear,int currentMonth,int currentDay,
                int birthYear,int birthMonth,int birthDay);
```

to calculate how many years old and separately how many months old the user is and report the result to the user. Thereby use the following example code to get the current date:

```
#include <time.h>
#include <iostream>
using namespace std;

int main()
{
  time_t currentTime;
  time(&currentTime);
  tm* timePtr = localtime(&currentTime);
  cout<<" year:" << timePtr->tm_year+1900
      <<" month:"<< timePtr->tm_mon+1
      <<" day:"  << timePtr->tm_mday <<endl;
}
```

A user is defined to be 1 month older in the next month only when the day number is larger than the birth day (for example a person born on January 5 is a month old on February 6 through March 5) or when the next month has passed (a person born on January 31 is a month old on March 1 through March 31). Similarly a user is defined to be 1 year older in the next year when the month number is larger than the birth month, or when the month is equal to the birth month and the day number is larger than the birth day, or when the next year has passed.

Day of the Week

Then ask the user on what day of the week he/she was born by having the user select a number from a list:

```
1: Monday
2: Tuesday
.: ......
7: Sunday
```

Design and implement your own algorithm (without using existing date libraries) for the function:

```
int dayOfTheWeek(int birthYear,int birthMonth,int birthDay);
```

which checks if the day of the week the user selected is consistent with the information provided earlier and report this to the user. Take into account that different months have different length and a year is a leap year from 1901 to 2099 when it is divisible by 4. For your implementation select a single reference day for which you know what day of the week it is/was (for example 2000-01-03 was a Monday).

Testing

Use the rand() function to generate random birth years, months and days and a random day-of-the-weeks and use your dayOfTheWeek() function to see if it is correct. Compute the ratio of correct random guesses over 1000 tries. Thereby use the srand() function to seed the random generator so that different executions of your program do not produce the same result. This ratio should tell you something about the correctness of your algorithm.

In addition use the source code in “testCode.cpp” to test your implementation for correctness. Your implementation has to pass all tests. You will probably write some tests yourself when implementing your algorithms. Make a habit of writting your own tests in such a way that they can be easily run again later.

Algorithm Design

Creating a new algorithm is not easy, what can help is to write a first draft in pseudo code (some high-level informal language you make up yourself that fits the problem). Then you incrementally improve the draft by fixing problems you identify. This should give you a good idea what the difficulties are and how the problem can best be decomposed in functions. Start from scratch when you get stuck and avoid adding unnecessary complexity. Only when you have have a good understanding of what your algorithm looks like should you want to deal with how it can be implemented in C++. Add any pseudo code you create to the source code as documentation.

Warnings

If you compile your source code using the “g++” compiler you can add “-W” flags to let the compiler warn you about things in your source code that could be the cause of incorrect results. This can save you a lot of time. For example use this in the shell to compile the “main.cpp” file and receive warnings if the compiler finds suspicious things in the file:

```
g++ -Wall -Wextra main.cpp
```

If instead your IDE compiles your source code for you, I expect you can turn on warnings in its menus. It’s worth your time to google or read the manual to turn these on if they aren’t already.

Submission

Submit your solution before the deadline to blackboard. Add to each solution:

- your name, student number and the name of the assignment

- references to the source of any algorithm or code that you did not create yourself

- operating system and compiler that was used to test the code