Dirk Praetorius,
Michele Ruggeri

Wintersemester 2015/16
16.-17.12.2016

Übungen zur Vorlesung
Einführung in das Programmieren für TM

Serie 10

Aufgabe 10.1. The command `cin` reads a text input only until the first space. Write a function `myFullName`, which reads your given and your surname from the keyboard and stores them both to strings. Then, put these two strings together in one string and print that string on screen. Save your source code as `MyFullName.cpp` into the directory `serie10`. Test your code on a suitable example. Do you know any other ways to read a longer keyboard input?

Aufgabe 10.2. Write a class `Name` which contains two members, `firstName` and `surname` of type `string`. Implement the set-method `setName` that has one string variable as input parameter, and splits the input in first name and surname automatically. Note that the input can contain multiple first names. Furthermore, write a method `printName` which prints out the whole name on the monitor. In case of multiple first names, the output should be shortened as follows: The name `Max Maxi Mustermann` should be printed out as `Max M. Mustermann`. Save your source code as `name.{hpp,cpp}` into the directory `serie10`.

Aufgabe 10.3. Extend the class `Fraction` from the lecture by the `public` method `void reduce()` that determines the reduced form of the fraction `numerator/denominator`. Use the *euclidean division algorithm*. Moreover, implement the method `setValue(string value)` that converts an arbitrary number, given as a string, into a fraction. For the implementation you can proceed as follows: First, find the decimal-point in the string and count the number of positions after the decimal-point. Then, erase the decimal-point from the string. The string now represents a natural number and can be converted into an `int` variable by use of the function `atoi`. This number is used for the numerator. Then, the denominator is set to 10^p, where $p \in \mathbb{N}$ is the number of positions after the decimal-point. Then, call the method `reduce()`. Finally, overload the method `setValue` in an appropiate way, so that `setValue(n)` for n of type `int` makes sense. Save your source code as `fraction.{hpp,cpp}` into the directory `serie10`. Test your code on a suitable example.
Hint: The method `find` of the class `string` allows you to find a specific character in the string, e.g., `int pos = value.find('.')` returns the position of the decimal-point in the string `value`. The call `value.erase(pos,k)`, erases `k` characters after the position `pos` in the string `value`. The function `atoi` from the standard library `cstdlib` converts a given string (in C-style) to an `int` variable. To get the string as `char *`, you can use the method `c_str()` of class `string`.

Aufgabe 10.4. Write a class `Stopwatch` that simulates a stopwatch. The stopwatch consists of two buttons: If the first button is pressed, then the time measurement starts. If the button is pressed again, then the time measurement stops. The second button is used to reset the time to zero. To realize this situation, implement the methods `pushButtonStartStop` (first button) and `pushButtonReset`. Implement another method that prints out the time formatted in the style `hh:mm:ss.xx`, e.g., if the measured time is two minutes, then the output should be `00:02:00.00`. Save your source code as `Stopwatch.{hpp,cpp}` into the directory `serie10`.
Hint: Use the data-type `clock_t` and the function `clock()` from the library `time.h`. It makes sense to use a variable `isRunning` of type `bool`. If the first button is pressed, then this variable is either set to `true` or `false`.

Aufgabe 10.5. Write a `Makefile` for the exercises of this sheet. It should contain:

- The compilation of all solved exercises.
- The generation of a library and an example of its usage.

Aufgabe 10.6. For the HR-department of the University it can be tedious to add and delete students one by one in their data. Therefore, overload the methods `graduate` and `newStudent` from the

class `University` from Exercise 9.7, so that the number of graduating and beginning students can be a parameter of the methods. Moreover, write Constructors which initialize your object with meaningful data. If the object is not initialized directly, then set `numStudents = 0`, `city = noWhere`, `name = noName`. Write a `plot`-routine to print the data of your object on screen. Save your source code as `University.{hpp,cpp}` into the directory `serie10`.

Aufgabe 10.7. Write a class `Hangman` that contains the methods `guessChar`, `solve`, `newString`. The class should store a string of length n which has to be guessed. The method `guessChar` allows the user to guess a single character in the string. In case that the string contains the character, the method `guessChar` should return the index resp. the indices of the the character in the string. In case that the string does not contain the character, an appropriate message should be printed out. The user loses, if he is not able to find the correct string after 8 tries. Write all necessary `set`- and `get`-methods and constructors, a method `newString` to start the game with a new word, and a method `solve` which allows to solve it. Moreover, write a main program to check if your implementation is correct. Save your source code as `hangman.{hpp,cpp}` into the directory `serie10`.

Aufgabe 10.8. According to the lecture Members of the class can only be accessed indirectly via `set`- and `get`-methods. What is the output of the following C++ program? Why is this possible? Explain why this is a bad programming style.

```
#include <iostream>
using std::cout;
using std::endl;

class Test{

private:
  int N;

public:
  void setN(int N_in) { N = N_in; };
  int getN(){ return N; };
  int* getptrN(){ return &N; };

};


int main(){

  Test A;
  A.setN(5);
  int* ptr = A.getptrN();
  cout << A.getN() << endl;
  *ptr = 10;
  cout << ptr << endl;
  cout << A.getN() << endl;

  return 0;
}
```