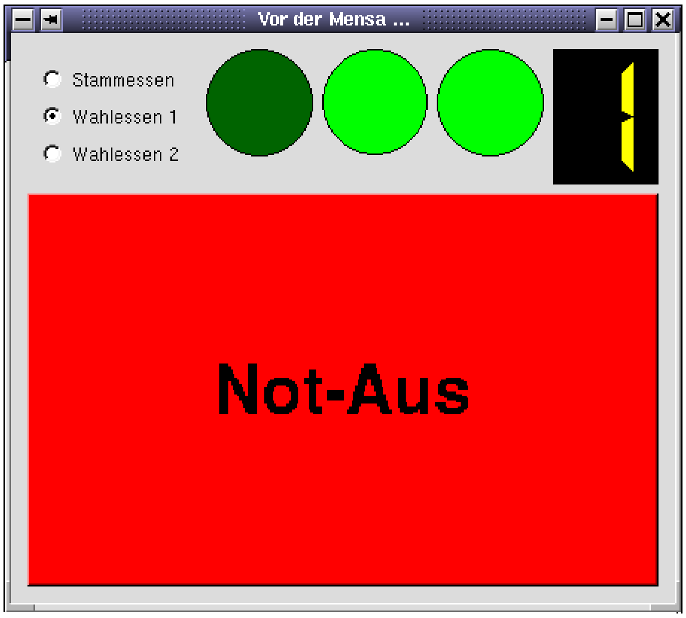
Dr. M. Grabert
Johannes Mayer

Objektorientierte Programmierung mit C++ (WS 2001/2002)

Abgabetermin: 31. Januar 2002

12 „Volkshochschule“ :-) (3+3 Punkte)

Jetzt wird's mal richtig schön praktisch. Bauen Sie mit dem Qt-Designer nach folgendem Bild einen Qt-Dialog mit Funktionalität und OHNE Programmieren nach:

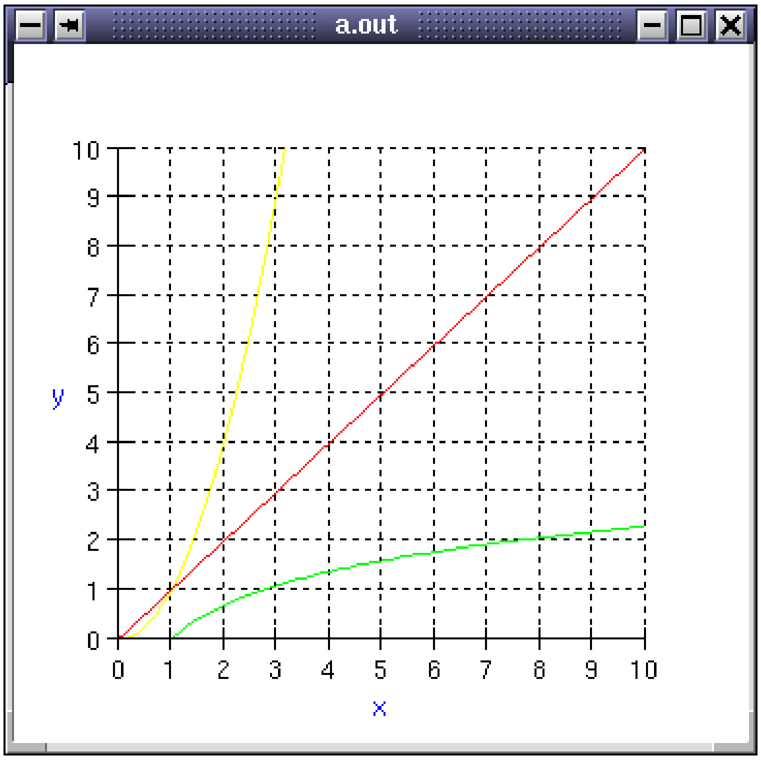


Was soll passieren: Die Anordnung soll genauso wie im Bild sein. Das Widget soll sich vergrößern lassen und dabei soll nur der „Not-Aus“-Button vertikal wachsen. Die anderen wachsen nur horizontal. Bei den Radiobuttons soll maximal einer gleichzeitig aktiv sein. Ist der oberste aktiv, sollen alle 3 LEDs leuchten, beim mittleren nur die rechten beiden LEDs und beim untersten Radiobutton nur die rechte LED. (Außer nach Drücken von „Not-Aus“.) In der LCD-Anzeige soll die Nummer des aktiven Radiobuttons angezeigt werden (von 0 bis 2). Außerdem sollen beim Drücken von „Not-Aus“ alle LEDs ausgehen. (Sie bekommen das Binary *main* zur Veranschaulichung der gewünschten Funktionalität.)
So, das war's aber noch nicht ganz! ;-) Jetzt wird's nämlich erst spannend. Eine LED ist ein primitives von mir selbst geschriebenes QWidget, das Sie im Quellcode bekommen (*led.h*

und *led.cc*). Das LED-Widget sollen Sie zunächst in den Designer einbinden und dann für Ihr Widget – wie beschrieben – verwenden. Zum Lösen dieser Aufgabe dürfen Sie keine Zeile Quellcode selbst schreiben. (Die Hauptdatei *main.cc* ist nämlich ebenfalls gegeben.) Lediglich das Makefile müssen Sie in Anlehnung an das aus der Vorlesung noch entwerfen.

13 Funktionsplot (7+7 Punkte)

Nun haben Sie ja schon was für's Berufsleben wieder kennen gelernt – nämlich Qt als Beispiel für eine umfangreiche Klassenbibliothek, die sogar in der Praxis verwendet wird. Wow – und das an der Uni! :-) Und jetzt lernen Sie schon wieder was: In der Praxis schaut man auch darauf, dass Software wiederverwendbar ist. Bei Qt sind die QWidgets die Programme, welche in jeder Qt-Anwendung wieder verwendet werden. Schrieben Sie nun auch ein wiederverwendbares Programm, einen Funktionsplotter. Dieser soll ein QWidget sein und etwa folgendes Aussehen besitzen:



Natürlich soll die Größe des Funktionsplots von der Größe des Widgets abhängen. Wie jedes QWidget soll auch ihr Widget einige Anpassungsmöglichkeiten (durch Methodenaufrufe) besitzen. So soll der Definitions- und Wertebereich einstellbar sein. Außerdem soll man angeben können, ob ein Gitter dargestellt wird, oder nicht. Des weiteren muss es natürlich möglich sein, die darzustellenden Funktionen von aussen einzustellen. (Die Anzahl der darstellbaren Funktionen sei beliebig.) Für jede Funktion, die zur Darstellung angegeben wird, soll es auch möglich sein, eine eigene Farbe zu spezifizieren. Außerdem soll die Darstellung von jeder Koordinatenachse jeweils auch logarithmisch möglich sein (unabhängig voneinander).
Schreiben Sie also eine Klasse `FunctionPlot`, die von `QWidget` abgeleitet ist, und implementieren Sie in dieser Klasse – unter Verwendung von `QPainter` – das Zeichnen von Funktionen wie beschrieben. Schreiben Sie außerdem eine kleine Qt-Anwendung (`main.cc`) zum Test ihres Widgets.

Viel Spass und Erfolg!