

Hardwarenahe Programmierung

Übungsaufgaben – 12. Dezember 2019

Diese Übung enthält Punkteangaben wie in einer Klausur. Um zu „bestehen“, müssen Sie innerhalb von 70 Minuten unter Verwendung ausschließlich zugelassener Hilfsmittel 12 Punkte (von insgesamt 24) erreichen.

Aufgabe 1: Kondensator

Ein Kondensator der Kapazität $C = 100 \mu\text{F}$ ist auf die Spannung $U_0 = 5 \text{ V}$ aufgeladen und wird über einen Widerstand $R = 33 \text{ k}\Omega$ entladen.

- (a) Schreiben Sie ein C-Programm, das den zeitlichen Spannungsverlauf in einer Tabelle darstellt. (5 Punkte)
- (b) Schreiben Sie ein C-Programm, das ermittelt, wie lange es dauert, bis die Spannung unter 0.1 V gefallen ist. (4 Punkte)
- (c) Vergleichen Sie die berechneten Werte mit der exakten theoretischen Entladekurve: $U(t) = U_0 \cdot e^{-\frac{t}{RC}}$ (3 Punkte)

Hinweise:

- Für die Simulation zerlegen wir den Entladevorgang in kurze Zeitintervalle dt . Innerhalb jedes Zeitintervalls betrachten wir den Strom I als konstant und berechnen, wieviel Ladung Q innerhalb des Zeitintervalls aus dem Kondensator herausfließt. Aus der neuen Ladung berechnen wir die Spannung am Ende des Zeitintervalls.
- Für den Vergleich mit der exakten theoretischen Entladekurve benötigen Sie die Exponentialfunktion `exp()`. Diese finden Sie in der Mathematik-Bibliothek: `#include <math.h>` im Quelltext, beim `gcc`-Aufruf `-lm` mit angeben.
- $Q = C \cdot U$, $U = R \cdot I$, $I = \frac{dQ}{dt}$

Aufgabe 2: Fehlerhaftes Programm: Hüpfender Ball

Das auf der nächsten Seite abgedruckte GTK+-Programm (Datei: [aufgabe-2.c](#)) soll einen hüpfenden Ball darstellen, ist jedoch fehlerhaft.

- (a) Warum sieht man lediglich ein leeres Fenster? Welchen Befehl muß man ergänzen, um diesen Fehler zu beheben? (3 Punkte)
- (b) Nach der Fehlerbehebung in Aufgabenteil (a) zeigt das Programm einen unbeweglichen Ball. Welchen Befehl muß man ergänzen, um diesen Fehler zu beheben, und warum? (2 Punkte)
- (c) Erklären Sie das merkwürdige Hüpfverhalten des Balls. Wie kommt es zustande? Was an diesem Verhalten ist korrekt, und was ist fehlerhaft? (5 Punkte)
- (d) Welche Befehle muß man in welcher Weise ändern, um ein realistischeres Hüpf-Verhalten zu bekommen? (2 Punkte)

Hinweis: Das Hinzuziehen von Beispiel-Programmen aus der Vorlesung ist ausdrücklich erlaubt – auch in der Klausur.

Allgemeiner Hinweis: Wenn Sie die Übungsaufgaben zu dieser Lehrveranstaltung als PDF-Datei betrachten und darin auf die Dateinamen klicken, können Sie die Beispiel-Programme direkt herunterladen. Dadurch vermeiden Sie Übertragungsfehler.

Viel Erfolg!

```

#include <gtk/gtk.h>

#define WIDTH 320
#define HEIGHT 240

double t = 0.0;
double dt = 0.2;

int r = 5;

double x = 10;
double y = 200;
double vx = 20;
double vy = -60;
double g = 9.81;

gboolean draw (GtkWidget *widget, cairo_t *c, gpointer data)
{
    GdkRGBA blue = { 0.0, 0.5, 1.0, 1.0 };

    gdk_cairo_set_source_rgba (c, &blue);
    cairo_arc (c, x, y, r, 0, 2 * G_PI);
    cairo_fill (c);

    return FALSE;
}

gboolean timer (GtkWidget *widget)
{
    t += dt;
    x += vx * dt;
    y += vy * dt;
    vx = vx;
    vy = 0.5 * g * (t * t);
    if (y + r >= HEIGHT)
        vy = -vy * 0.9;
    if (x + r >= WIDTH)
        vx = -vx * 0.9;
    if (x - r <= 0)
        vx = -vx * 0.9;
    gtk_widget_queue_draw_area (widget, 0, 0, WIDTH, HEIGHT);
    g_timeout_add (50, (GSourceFunc) timer, widget);
    return FALSE;
}

int main (int argc, char **argv)
{
    gtk_init (&argc, &argv);

    GtkWidget *window = gtk_window_new (GTK_WINDOW_TOPLEVEL);
    gtk_widget_show (window);
    gtk_window_set_title (GTK_WINDOW (window), "Hello");
    g_signal_connect (window, "destroy", G_CALLBACK (gtk_main_quit), NULL);

    GtkWidget *drawing_area = gtk_drawing_area_new ();
    gtk_widget_show (drawing_area);
    gtk_container_add (GTK_CONTAINER (window), drawing_area);
    gtk_widget_set_size_request (drawing_area, WIDTH, HEIGHT);

    gtk_main ();
    return 0;
}

```