Microcontroller / C-Programmierung

Selbststudium Semesterwoche 1

1. Aufgabe 1-15 (Buch S. 26)*: Umrechnung Fahrenheit→Celsius mit Funktion.

```
#include <stdio.h>
#include <math.h>

float CelsiusToFahrenheit(float value);
float FahrenheitToCelsius(float value);

void main(void)
{
	char c = 0;
	float input = 0.0f;

	while(c!=51)
	{
		system ("CLS");
		printf("\n TEMPERATURE CONVERTOR\n");
		printf(" ====================\n\n");
		printf(" 1 - to convert from Celsius to Fahrenheit\n");
		printf(" 2 - to convert from Fahrenheit to Celsius\n");
		printf(" 3 - to exit\n\n ");

		do
		{
			c = getchar();

		}while(c==10);

		if(c==49)
		{
			printf("\n Enter value in \xA7\C: ");
			scanf("%f", &input);
			printf(" %.2f \xA7\C is %.2f \xA7\F", input, CelsiusToFahrenheit(input));
			printf("\n\n Press [ENTER] to return...\n ");
			getchar();
			getchar();
		}
		else if(c==50)
		{
			printf("\n Enter value in \xA7\F: ");
			scanf("%f", &input);
			printf(" %.2f \xA7\F is %.2f \xA7\C", input, FahrenheitToCelsius(input));
			printf("\n\n Press [ENTER] to return...\n ");
			getchar();
			getchar();
		}


	}
}

float CelsiusToFahrenheit(float value)
{
	return (value * ((float)9/(float)5)) + 32;
}

float FahrenheitToCelsius(float value)
{
	return (value - 32) * ((float)5/(float)9);
}
```

2. Aufgabe 1-19 (Buch S. 19) *: Reverse
Hinweis: Falls Sie ein Beispiel im Buch nicht nachvollziehen können, implementieren Sie doch das Programm in NetBeans und verwenden den Debugger, um das Programm schrittweise zu durchlaufen und den Programmablauf und die Veränderung von Werten zu verfolgen.

```
#include <stdio.h>
#define SIZE 1024

void main(void)
{
	char a[SIZE];
	int strLength;
	int i = 0;

	// reading line by line
	while(gets(a) != NULL)
	{
		strLength = strlen(a);

		// printing each line reversed
		for(i=strLength-1;i>=0;i--)
		{
			printf("%c",a[i]);
		}

		// finally add a new line character
		// at the end of each line
		printf("\n");
	}
	getchar();
}
```

3. Power
Schreiben Sie eine Funktion power, welches Potenzen berechnet. Zum Testen schreiben Sie eine main Funktion, welche die Potenzen von 20 bis 210 und -30 bis -310 ausgibt. Vergleichen Sie anschliessend ihre Lösung mit dem Buch S. 24.

```
#include <stdio.h>
#include <math.h>

double power(int base, int exp);

void main(void)
{
	int base = 0;
	int exp = 0;
	char buffer[50];

	system ("CLS");
	printf("\n");
	printf(" EXPONENTIAL CALCULATION \n");
	printf(" ======================== \n");

	printf("\n Enter base: ");
	scanf("%d", &base);
	printf(" Enter exponent: ");
	scanf("%d", &exp);

	printf(" Result = %lf", power(base, exp));

	printf("\n\n Press [ENTER] to exit...\n ");
	getchar();
	getchar();
}

double power(int base, int exp)
{
	double returnvalue = 1;
	int i;
	for(i=0; i<abs(exp); i++)
	{
		returnvalue = returnvalue * base;
	}

	if(exp<0)
	{
		returnvalue = 1 / returnvalue;
	}
	return returnvalue;
}
```

4. Wörter zählen
Schreiben Sie ein Programm, das Wörter zählt und das Ergebnis ausgibt. Zum Testen rufen Sie das Programm auf der Kommandozeile auf und verwenden die Standardeingabe aus einer Datei (z.B. longestLine.exe < LongestLine.c)
Hinweis 1: Das EOF Zeichen geben Sie über die Tastatur mit der Tastenkombination 'Ctrl' + 'D' ein.
Hinweis 2: Verwenden Sie eine Zustandsvariable (mit den zwei Werten IN und OUT), welche Sie zum Merken verwenden, ob Sie in einem Wort sind oder nicht. Je nach eingelesenem Zeichen ändert Sich der Zustand. Wenn der Zustand auf IN wechselt, erhöhen Sie den Wörterzähler. Vergleichen Sie anschliessend ihre Lösung mit dem Buch S. 20

```
#include <stdio.h>
#define SIZE 1024

void main(void)
{
	char a[SIZE];
	int chars = 0;
	int letters = 0;
	int words = 0;

	int i = 0;
	int in = 0;
	int strLength = 0;

	// reading line by line
	while(gets(a) != NULL)
	{
		strLength = strlen(a);
		chars += strLength;

		// reading characters of a line
		for(i=0; i<strLength; i++)
		{
			if((a[i] >='A') && (a[i]<='Z') ||
			   (a[i] >='a') && (a[i]<='z'))
			{
				// set an in-pointer to prevent
				// counting the same word twice
				if(in==0)
				{
					words++;
					in = 1;
				}

				// counting all letters (A-Z + a-z)
				letters++;
			}
			else
			{
				// release the in-pointer
				// to mark the beginning of a new word
				in = 0;
			}
		}
		// release the in-pointer
		// to mark the beginning of a new line
		in = 0;
	}

	printf("\n");
	printf("Characters:\t%d \n", chars);
	printf("Letters:\t%d \n", letters);
	printf("Words:\t\t%d \n", words);
	getchar();
}
```

5. Längste Eingabezeile *
Schreiben Sie ein Programm, dass eine Reihe von Textzeilen einliest und die längste Zeile ausgibt. Verwenden Sie folgende Programmstruktur:

```
while (es gibt noch eine Zeile)
    if (sie ist länger als die bisher längste)
       Zeile speichern
       Zeilenlänge speichern
längste Zeile ausgeben
```

Zum einlesen einer Zeile verwenden Sie eine Funktion getline, und zum Speichern der Zeile eine Funktion copy. Vergleichen Sie anschliessend ihre Lösung mit dem Buch S. 28.

```
#include <stdio.h>
#include <string.h>
#define SIZE 1024

void main(void)
{
	char a[SIZE];
	char longestLine[SIZE];
	int longestLineCount = 0;
	char strLength;

	// reading line by line
	while(gets(a) != NULL)
	{
		// check whether the actual line
		// is longer then the present longest line
		strLength = strlen(a);
		if(strLength > longestLineCount)
		{
			longestLineCount = strLength;
			strcpy(longestLine, a);
		}


	}
	printf("Longest line counts %d characters:\n", longestLineCount);
	printf("%s", longestLine);
	getchar();
}
```