/\*Aufbau -------------------------------------------------------------------\*/

/\* erstes Programm \*/

#include <stdio.h>              //Standard Ein-/Ausgabe

#include <math.h>

#include <stdlib.h>             //für rand()

#define KONSTANTE 9

#define MAKRO(x) (x)\*(x)        //"kurze Funktion", Klammerung wichtig!

int function (float var);     //Deklaration der Fkt.

int main(void)

{

    int var, var2 = 1;

    long langerint;

    float kommazahl;

    double langerfloat;

    long double nochlänger;

    char zeichen = 'n', nocheinzeichen,…;

    (float typecasting)

    funktionsaufruf = function(var);

    [Funktionsblock]

    return 0;

}

int function (int var)

{

[Funktionsblock]

}

------------------------------------------------------------------------------

/\*FELDER -------------------------------------------------------------------\*/

feld[5];                        //Feld, 5 Zeichen, vergiss nicht \0!

ZweiDFeld[5][5]                 //usw.

**FELDER**

/\*keine Überwachung der Feldgrenzen!!!\*/

/\*STRUCTS ------------------------------------------------------------------\*/

typedef struct

{

    int warteschlange[5];       //für jedes struct Feld der Größe 5

} Ort;

Ort O[18];                      //18 structs

O[5].warteschlange[0] = 7;      //in struct 5, speichere in warteschlange auf Platz 0

**STRUCTS**

/\*Hinweis: Union gibt es auch\*/

/\*POINTER ------------------------------------------------------------------\*/

\*p\_pointer = zeigtAufVar        //auch Feld möglich

**POINTER**

/\*speichert bloß Adresse\*/

/\*printf/scanf -------------------------------------------------------------\*/

printf("Text\n\n");

printf("Ergebnis: %i und %i", var, var2);

puts(„“);                   /\*put string, mit \n\*/

fflush(stdin);

scanf("%i", &zahl);

printf("\n-20s | %8.2f", titel, preis);

/\*20 Zeichen linksbündig String, 8 Zeichen und zwei Nachkommastellen rechtsbündig\*/

/\*% -- weitere Flags

i...    int

**FLAGS**

d...    dezimal

o...    oktal

x/X...  Hexadezimal

f...    Gleitkommazahl

e/E...  wissenschaftl. Zahl

c...    char

s...    Zeichenfolge

%%...   Prozentzeichen\*/

/\*Escapesequences ----------------------------------------------------------\*/

/\*\033[0m -- Zahl vor dem m

0...    reset

1...    fett

3...    kursiv <> 23

4...    unterstrichen <> 24

**ESCAPESEQUENCES - KONSOLENGESTALTUNG**

(5...   langsames blinken <> 25

6...    schnell)

9...    durchgestrichen <> 29

(10-19  Schriftarten)\*/

/\*Colours\*/

/\*3...Foreground-9...bright, 4...background-10...bright

9...    Standard

0...    schwarz

**FARBEN**

1...    rot

2...    grün

3...    gelb

4...    blau

5...    Magenta

**PRINTF/SCANF**

6...    Cyan

7...    weiß\*/

--> en.wikipedia.org/org/ANSI\_escape\_code

/\*OPERATOREN ---------------------------------------------------------------\*/

+, -, \*, /, %;

nur mit math: pow(basis, potenz); sin(rad); cos(); tan();

-=, +=, \*=, /=, %=

<, >, ==, <=, >=, !=

&& UND

**OPERATOREN**

|| OR

! NOT

(/\*ist das hier wahr?\*/) ? (/\*tue das, wenn wahr\*/) : (/\*ansonsten das\*/)

/\*IF  ----------------------------------------------------------------------\*/

If (/\*Bedingung\*/)

{

[Funktionsblock]

}

else

{

[Funktionsblock]

}

**IF**

/\*oder else if ()\*/

/\*WHILE  -------------------------------------------------------------------\*/

do

{

[Funktionsblock]

}

while (/\*Bedingung\*/);

while (/\*Bedingung\*/)

{

[Funktionsblock]

**WHILE**

}

/\*FOR  ---------------------------------------------------------------------\*/

for (i = 2 /\*Ausdruck\*/; i <= zahl /\*solange wahr...\*/; i++ /\*Reinitialisierung/Zählung der Schleife\*/)

{

ergebnis=ergebnis\*i;

}

**FOR**

/\*SWITCH -------------------------------------------------------------------\*/

switch (zu\_prüfende\_var)

{

case 'w':

    [Funktionsblock]

    break;

case '[hat diesen Wert?]':

    [Funktionsblock]

    break;

default:

    [Funktionsblock]

**SWITCH**

    break;

}

/\*SCHLEIFEN  ---------------------------------------------------------------\*/

continue;       //überspringt aktuellen Schleifendurchgang

break;          //Schleife verlassen (Switch!)

**SCHLEIFEN**

exit();         //Fkt. verlassen

/\*LESEN/SCHREIBEN  ---------------------------------------------------------\*/

FILE \*fp;

fp = fopen("zahlen.txt", "w");  //öffnen

fprintf(fp, "%d\n", i);         //Schreibt in Datei fp

fclose(fp);                     //schließt

/\*weitere Möglichkeiten: fputs, fputchar, fwrite\*/

fp = fopen("zahlen.txt", "r");

fscanf(fp, "%d", &temp);        //liest Zeilenweise

fclose(fp);

/\*weitere Möglichkeiten: fgets, fgetchar, fread\*/

/\*

r, read

w, write

a, append

b, binary

t, textmodus

**DATEIARBEIT**

\*/

/\*Hinweis: if fp == NULL (nach öffnen)--> Datei konnte nicht geöffnet werden\*/

/\*rand  --------------------------------------------------------------------\*/

srand(time(NULL));      //Initialisierung, einmal am Anfang notwendig (eine ändernde Größe wie time)

zahl = rand() % (obergrenze - untergrenze + 1) + untergrenze;

/\*Modulo --> Rest, max. so groß wie Modulo+1 (deswegen +1)

Untergrenze garantiert über letzten Summanden (mind. so hoch)

**RANDOM**

Obergrenze garantiert durch Abzug der Untergrenze vorher\*/

/\*string.h  ----------------------------------------------------------------\*/

strcpy(var, "Text");    //kopiert von 2 auf 1

**STRINGS**

strcat(var, var2);      //hängt 2 an 1 ran

strtok(var, delimiter); //trennt alles hinter Limiter ab

strcmp(var1, var2);     //Vergleicht 1 und 2; gibt integer zurück: 1 - 1 weicht als erstes ab, 0 - beide gleich, -1 - 2 wicht als erstes ab

/\*Tastatureingabe nicht mehr abwarten  conio.h -----------------------------\*/

if (kbhit()){}          //wenn eine Eingabe in der Tastatur getätigt wurde, tue das; sinnvoll für Schleifen

**KBHIT**

/\*Konsole leeren unter Windows / windows.h  --------------------------------\*/

system("cls");

**KONSOLE LEEREN**

/\*ctype.h  -----------------------------------------------------------------\*/

tolower(c)  //verwandelt Character (!) in Kleinbuchstaben

**KLEIN/GROSS**

toupper(c)  //Großbuchstaben

/\*--------------------------------------------------------------------------\*/

--> https://www.c-howto.de/tutorial/