Einführung in C - Introduction to C

Prof. Dr. Eckhard Kruse

DHBW Mannheim

```c
#include <stdio.h>

int main()
{
   printf("Hello TINF21AI!\n");
   return 0;
}
```

Organisation – TINF21AI

- Termine:
  - s. Stundenplan
  - Pausen in der VL nach Bedarf
  - Selbststudium (s. Modulhandbuch): 174 h!

- Vorlesung und Übungen = viel eigenes Programmieren
  - Nutzen Sie Frage- und Feedbackmöglichkeiten in der VL!

- Fragen: Am besten direkt in der VL
  - E-Mail: eckhard.kruse@dhbw-mannheim.de
  - Raum 344 B, Tel. (0621) 4105 1262

- Verteilung der Folien und Übungsaufgaben/Code snippets
  - während/nach jeder Vorlesung per E-Mail-Verteiler

- Leistungsnachweis:
  - Programmentwurf (individuell + unter Aufsicht...)

Wie funktioniert Lernen?

Eigentlich selbstverständlich...

Bewährte Regeln für effizientes gemeinsames Arbeiten, Besprechungen usw. → gilt auch für diese Vorlesung:

- Pünktlichkeit (Vorlesungsbeginn, Pausenende)

- Anwesenheit: von Körper + Geist

- Anzahl gleichzeitig redender Personen ≤ 1 (Ausnahme: Übungen)

- Konzentration auf das Geschehen
  - Laptops nur für Übungsaufgaben

- Handys ausschalten, keine Telefonate

- ggf. Feedback zum Arbeitsprozess
 - Stoff zu schnell / zu langsam? Pausenbedarf?

Example: Real-life C code snippet...

```c
...
#include "mmio.h"
#include <direct/messages.h>

#include <core/system.h>

/* * Set up the extended FIFO. @note It will be turned on if ucovl->scrwidth > 1024. */

void uc_ovl_setup_fifo(UcOverlayData* ucovl, int scrwidth)
{
    u8* mclk_save = ucovl->mclk_save;

    if (!iopl(3)) {
        if (scrwidth <= 1024) { // Disable
            if (ucovl->extfifo_on) {
                dfb_layer_wait_vsync(dfb_layer_at(DLID_PRIMARY));
                outb(0x16, 0x3c4); outb(mclk_save[0], 0x3c5);
                outb(0x17, 0x3c4); outb(mclk_save[1], 0x3c5);
                outb(0x18, 0x3c4); outb(mclk_save[2], 0x3c5);
                ucovl->extfifo_on = false;
            }
        }
        else { // Enable
//….
        }
    }
}
…
```

WS 2021

DHBW Mannheim
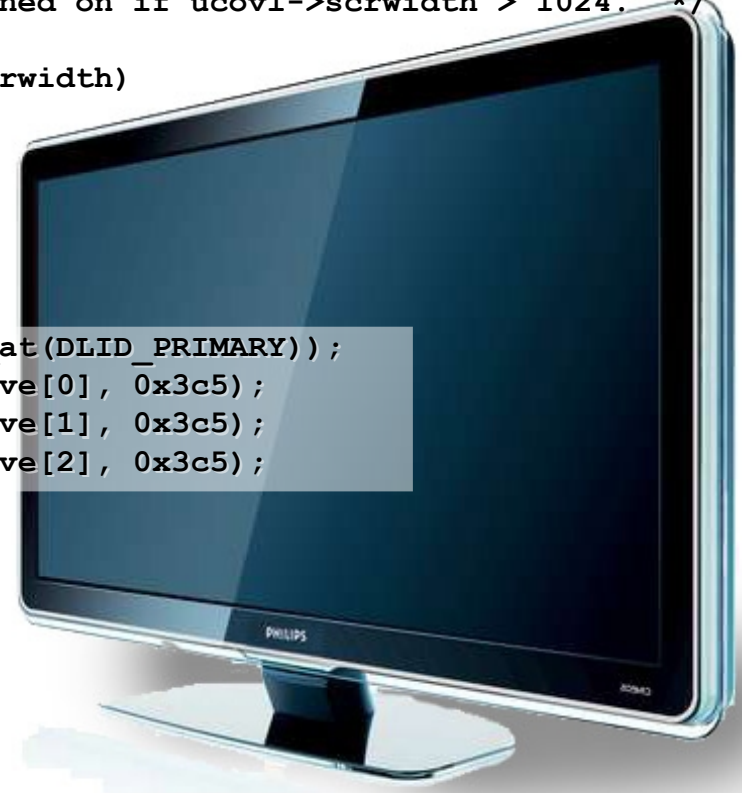
Motivation – Why C?

The programming language C is:

- Very old (* 1972)

- Cryptic (`--a->b? *++c: ++*c;`)

- Low-level (no object orientation)

- Dangerous, high risk of making errors ("everything is allowed")

Established, stable, standardized, omni-present, OS-independent, „runs everywhere"

fast, efficient, lean

provides 'full control', e.g. direct hardware access, OS drivers, realtime requirements...

perfect e.g. for embedded systems (Arduino, Pi, TVs, cars, ...)

„C forces you to build a mental model of what the computer is actually doing when you run your programs. … So if you want to write extremely good software — in any language — you should ignore all the advice to learn Python… and instead begin your journey with C. The view from the top is nice, but if you aspire to climb mountains, there's no better place to start than at the bottom." www.evanmiller.org/you-cant-dig-upwards.html

C is still one of the most important programming languages!

(And more modern languages, e.g. C++, Java, C#, have inherited a lot from it.

TIOBE Programming Community Index
Source: www.tiobe.com
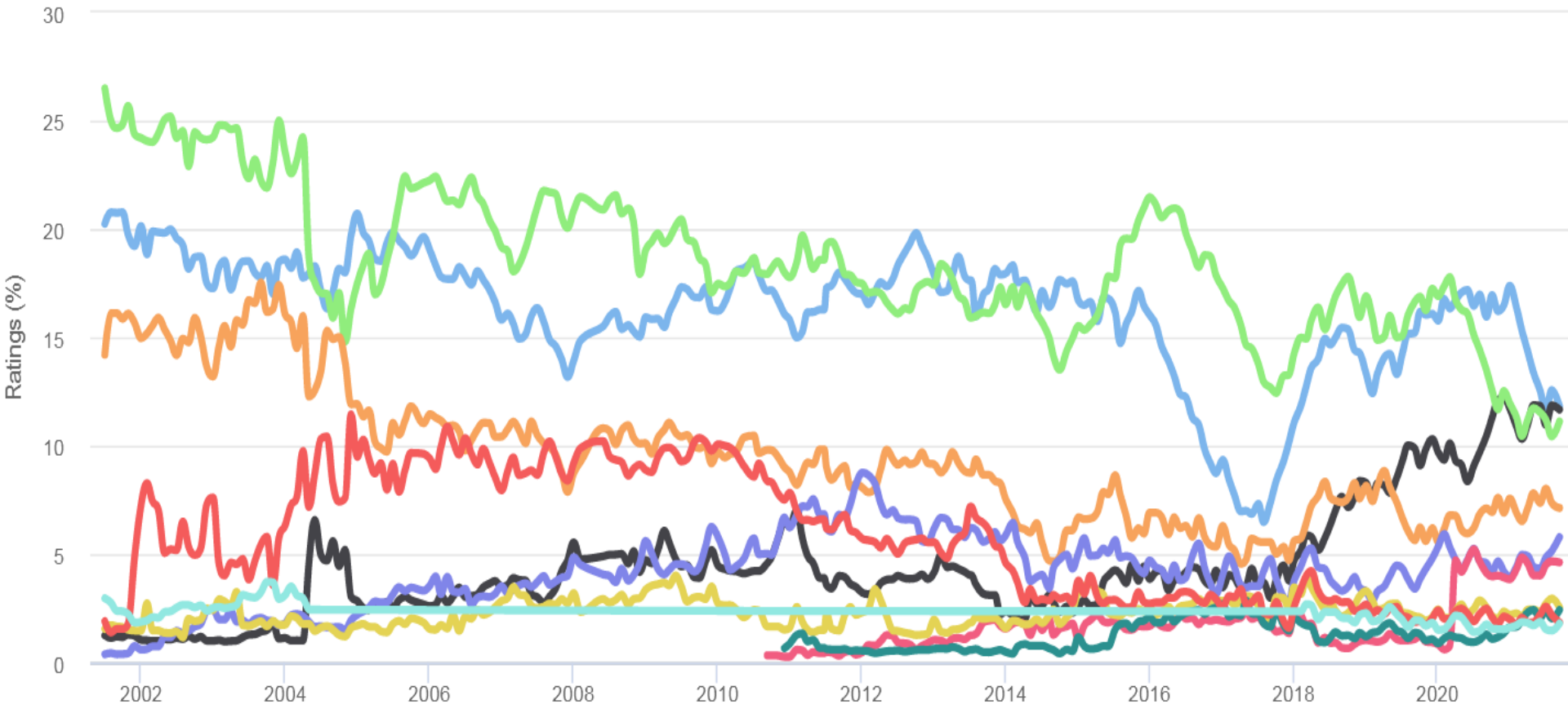
How to become a great C programmer?

data types and operators

Quick overview

functions and program structure

pointers and memory management

control flow

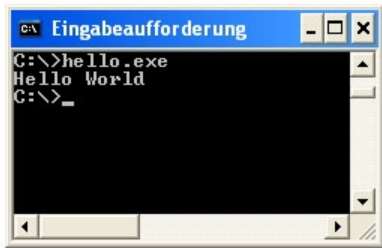libraries

real-life C projects

Examples

Exercises

DHBW   Mannheim
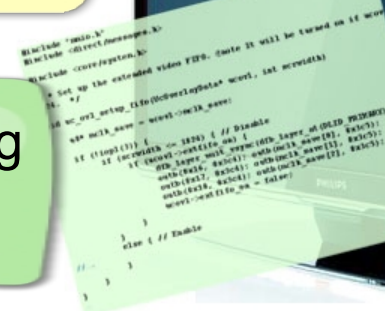Duale Hochschule Baden-Württemberg
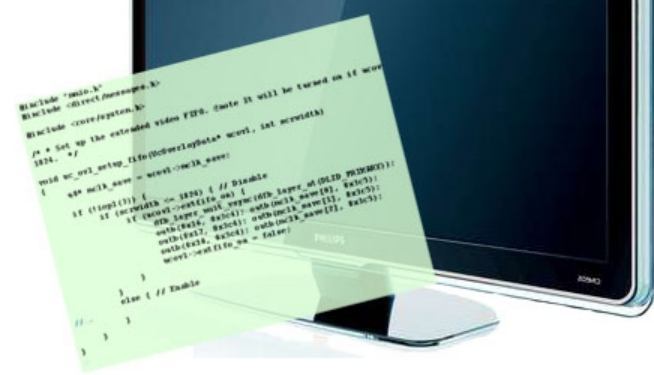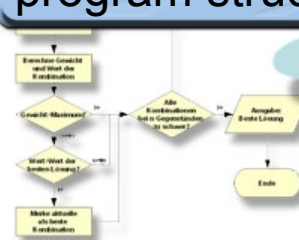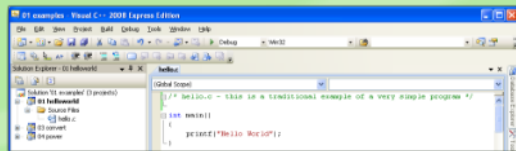
Goals – Why learn C? (→ Studienplan)

- Make your own programming experiences
- Get to know one of the most important programming languages
- Understand procedural programming concepts
- Understand key computing concepts (data structures, memory handling, pointers)

… and impress your friends by understanding

weird things: `a--->b? ++*++c: ++*--d;`

Einführung in C - Introduction to C

Quick overview of C concepts

Prof. Dr. Eckhard Kruse

DHBW Mannheim

Hello world

```c
/* hello.c – this is a traditional example of a very
   simple program */
#include <stdio.h>

main()
{
    printf("Hello World!\n");
}
```

Common functions provided in Standard Library

```
>lc hello.c

>hello.exe

Hello world!

>
```

This text, i.e. the ‚source-file‘, is compiled into a program, which then can be executed.
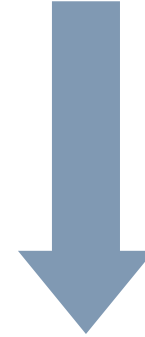
Variables + arithmetics

```
main( ) {
  int a, b, c, res;
  a = 5;  b = 7;  c = 18;
  res = (a*a+3*b+c)/2;
  printf("result is %d", res);
}
```

Explicit, static typing

The program is executed in a linear way, command by command.
Each command is terminated by a semicolon.

```
>calc.exe

result is 32

>
```

DHBW   Mannheim
Duale Hochschule Baden-Württemberg

Control flow

```c
/* Celsius – Fahrenheit converter */

main( )
{
  int lower, upper, step;
  float cel, fahr;
  lower=0; upper=100; step=10;

  for(cel=lower; cel<=upper; cel=cel+step)
  {
    fahr=9.0/5.0*cel+32.0;
    printf("%f  %f\n",cel,fahr);  /* output */
  }
}
```

Loop: Repeated execution of a command block
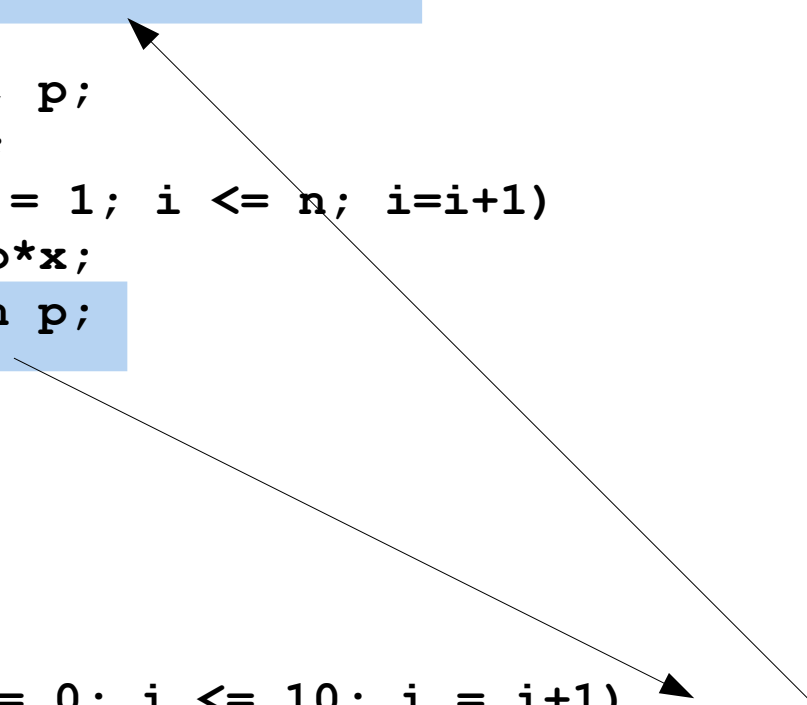
```
>convert.exe

0.0   32.0
10.0   50.0
20.0   68.0
30.0   86.0
40.0   104.0
50.0   122.0
60.0   140.0
70.0   158.0
80.0   176.0
90.0   194.0
100.0   212.0

>
```

DHBW Mannheim
Duale Hochschule Baden-Württemberg

Functions and program structure

```
int power(int x, int n)
{
    int i, p;
    p = 1;
    for(i = 1; i <= n; i=i+1)
        p=p*x;
    return p;
}


main()
{
    int i;
    for(i = 0; i <= 10; i = i+1)
        printf("%d %d %d\n", i, power(2,i), power(-3,i));
}
```

```
>power.exe
0 1 1
1 2 -3
2 4 9
3 8 -27
4 16 81
5 32 -243
6 64 729
7 128 -2187
8 256 6561
9 512 -19683
10 1024 59049
>
```

WS 2021

DHBW Mannheim
Duale Hochschule Baden-Württemberg

Arrays

```
/* prime sieve of Eratosthenes */

main()
{
    int i, j;
    int is_prime[200];

    for(i=0; i<200; i=i+1)
        is_prime[i]=1;

    for(i = 2; i<200; i=i+1)
        for(j= 2*i; j <200; j=j+i)
            is_prime[j]=0;

    printf("Prime numbers are: ");
    for(i=2; i<200; i=i+1)
        if(is_prime[i]>0)
            printf("%d ",i);
}
```

```
>eratosthenes_simple.exe
Prime numbers are: 2 3 5 7 11 13 17 19 23
29 31 37 41 43 47 53 59 61 67 71 73 79 83
89 97 101 103 107 109 113 127 131 137 139
149 151 157 163 167 173 179 181 191 193
197 199
>
```

No array bounds checking!

Strings

```c
#include <stdio.h>
#include <string.h>
#include <stdlib.h>

main()
{                        string = array of characters
   char name[100], c;
   int i, p1, p2;

   printf("Please enter your name:\n");
   gets(name);

   for(i=0; i<10; i=i+1)
   {                            string functions
      p1=rand()%strlen(name);
      p2=rand()%strlen(name);

      c=name[p1];
      name[p1]=name[p2];
      name[p2]=c;
      printf("Hello %s\n", name);
   }
}
```
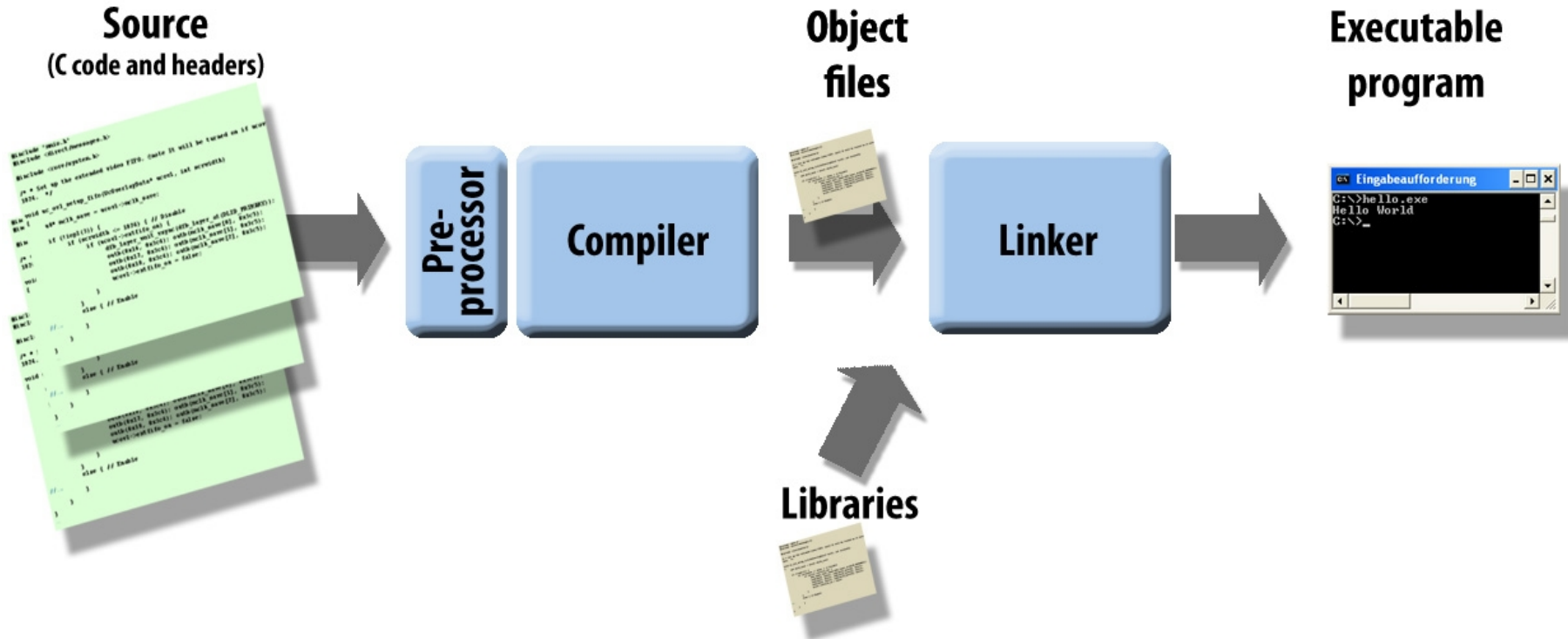
```
>string_shuffle.exe
Please enter your name:
Eckhard Kruse
Hello Eckhsrd Kruae
Hello Eckhsud Krrae
Hello Eckasud Krrhe
Hello Eck sudaKrrhe
Hello Eck srdaKurhe
Hello Eck srdaKurhe
Hello EckhsrdaKur e
Hello KckhsrdaEur e
Hello KckhdrsaEur e
Hello Kckhd saEurre
>
```

Introduction to C: 1. Introduction   **16**   Prof. Dr. Eckhard Kruse

WS 2021

DHBW   Mannheim
Duale Hochschule Baden-Württemberg

C is a compiled language

Source (C code and headers) → Pre-processor → Compiler → Object files → Linker → Executable program

Libraries → Linker

WS 2021

DHBW Mannheim
Duale Hochschule Baden-Württemberg

First steps in C

Exercise

1.1 First steps in C

The most important thing about learning programming is: **Try it yourself!**

a) Install a C compiler - choose according to your personal preference:

    a) Light-weight: z.B. Visual Studio Code, LCC-Win

    b) IDE, e.g. Visual Studio Community, Netbeans, Code::Blocks, Clion, Eclipse…

```
>lc hello.c

>hello.exe
Hello World!
>
```

b) Have a look at the example programs.

c) Compile and run them.

d) Modify them and see what happens.

Code snippets 101-106

Learning by doing!

1. Don't be afraid
2. Have a first look at what's there (source files…)
3. Get it compiled and run
4. See what happens
5. Try to modify it and check results
6. Accept that at first not everything needs to be understood fully
7. Try the debugger
8. Look for errors / improvement potential / possible extensions