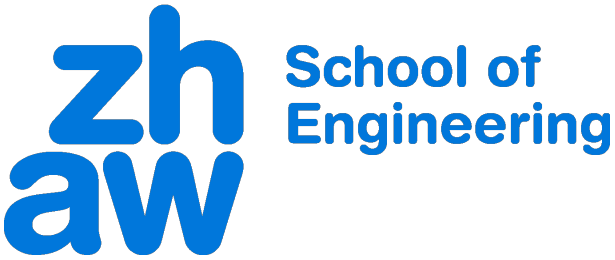
ZHAW Zurich University of Applied Sciences
Winterthur



Zusammenfassung INF2
Studienwochen 1-4

Written by: Severin Sprenger
October 13, 2025
Zf. INF2 SW 1-4

1 File reading/writing

1.1 Open file

```
FILE *fopen (const char *filename, const char *mode)
```

1.2 Close file

```
int fclose (FILE *stream)
```

1.3 Write text file

```
int fprintf (FILE *stream, const char *format, ...)
int fputc(int ch, FILE* stream) int fputs (const char* str, FILE* stream)
```

1.4 Read text file

```
int fscanf (FILE *stream, const char *format, ...)
int fgetc (FILE* stream)
char* fgets (char* str, int count, FILE* stream)
```
↪ Reading stops at LF (LF will be in return string) or until max num reached

1.5 Flush data buffer

```
int fflush (FILE *stream)
```

1.6 String tokenizer

```
char *strtok (char *str, const char *delimters)
```

1.7 Write binary file

```
int fwrite (const void *ptr, int size, int n, FILE *fp)
```

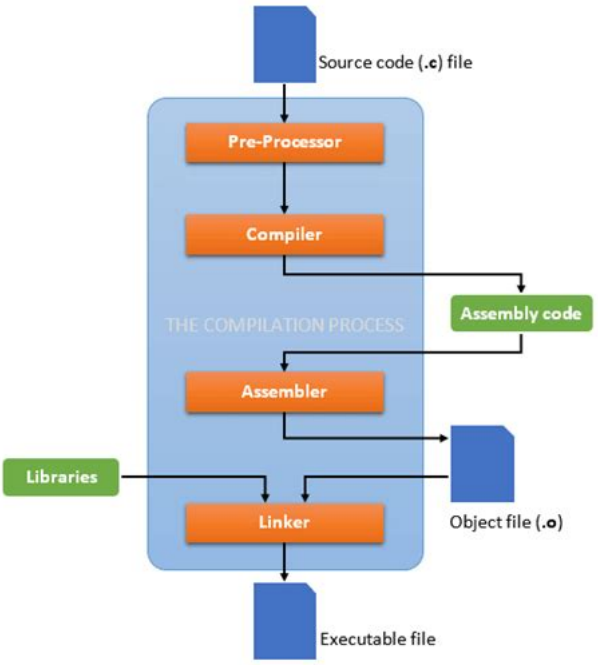
1.8 Read binary file

```
int fread (const void *ptr, int size, int n, FILE *st)
```

1.9 Access modes

`r`	Read
`rb`	Read binary
`w`	Write
`wb`	Write binary
`a`	Append
`ab`	Append binary

2 Compiler



`#include<...>` → Lib file from standard folder
`#include"..."` → Lib file from cwd
`#ifdef/#ifndef ... #endif` → Conditionals for pre-processor

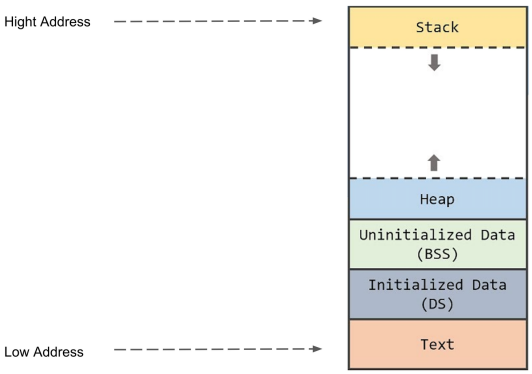
2.1 Constants

`__FILE__`	Name of the currently processed file
`__DATE__`	Date of translation
`__TIME__`	Time of translation
`__LINE__`	Line that is being processed
`__STDC__`	Defined when compiler in C mode
`__cplusplus`	Defined when compiler in C++ mode

2.2 Example macro

```
#define ALARM(text) \
    printf("******* "); \
    printf("%s", text); \
    printf("******* ");
```

3 Dynamic memory (`<stdlib.h>`)



Code: Program code

Data: Global and static variables

Heap: Dynamically allocated memory

Stack: Local data (Variables in functions), Parameters for functions

3.1 Allocate memory

```
void *malloc (size_t size)
```

3.2 Allocate memory and zero

```
void *calloc (size_t num, size_t size)
```

3.3 Reallocate memory

```
void *realloc (void *ptr, size_t size)
```

3.4 Free memory

```
void free (void *ptr)
```

4 Escape sequences

Sequence	ASCII	Description
\n	10	new line
\r	13	carriage return
\t	09	horizontal tab
\v	11	vertical tab
\f	12	form feed (Curser to start of next page)
\b	08	backspace
\a	07	bell
\'	39	apostrophe
\"	34	quote
\\	92	backslash
\nnn		char value in octal (n = 0..7)
\xhh		char value in hex

5 Format

Sequence	Output
%d or %i	int
%c	character
%e or %E	double in format [-] d.ddd e±dd
%f	double in format [-] ddd.ddd
%ot	int as octal
%s	string
%p	As pointer address
%u	unsigned int
%x pr %X	int as hex
%%	%

`%ni` output as int / flush right / n characters wide
`%0ni` output as int / flush right / n characters wide filled with 0
`%.nf` output as float / n digits after comma
`%+i` output as int / forces plus or minus sign
`%#x` int as hex / forces `0x` before number

5.1 Weird operators

`i++` → Value is returned before and then incremented
`++i` → Value is first incremented and then returned