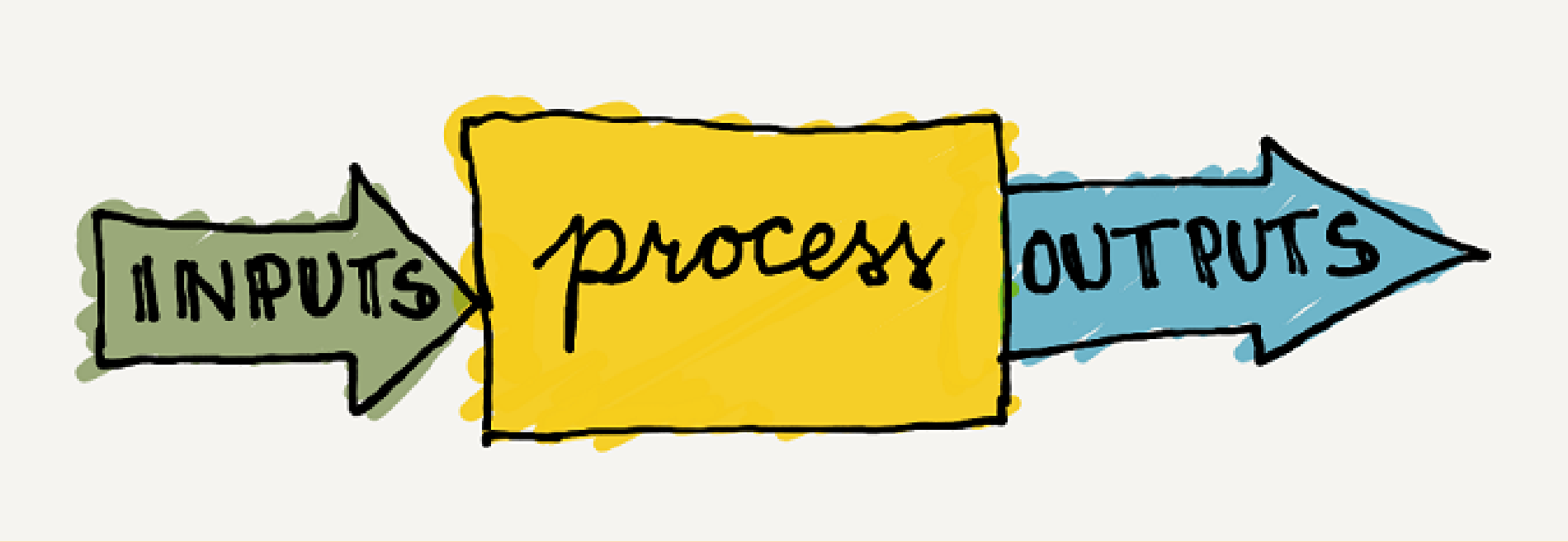
1. Variables, Types, I/O

Hengfeng Wei (魏恒峰)
hfwei@nju.edu.cn

Sep. 27, 2024

Review

Program = Input + Data + Operations + Output



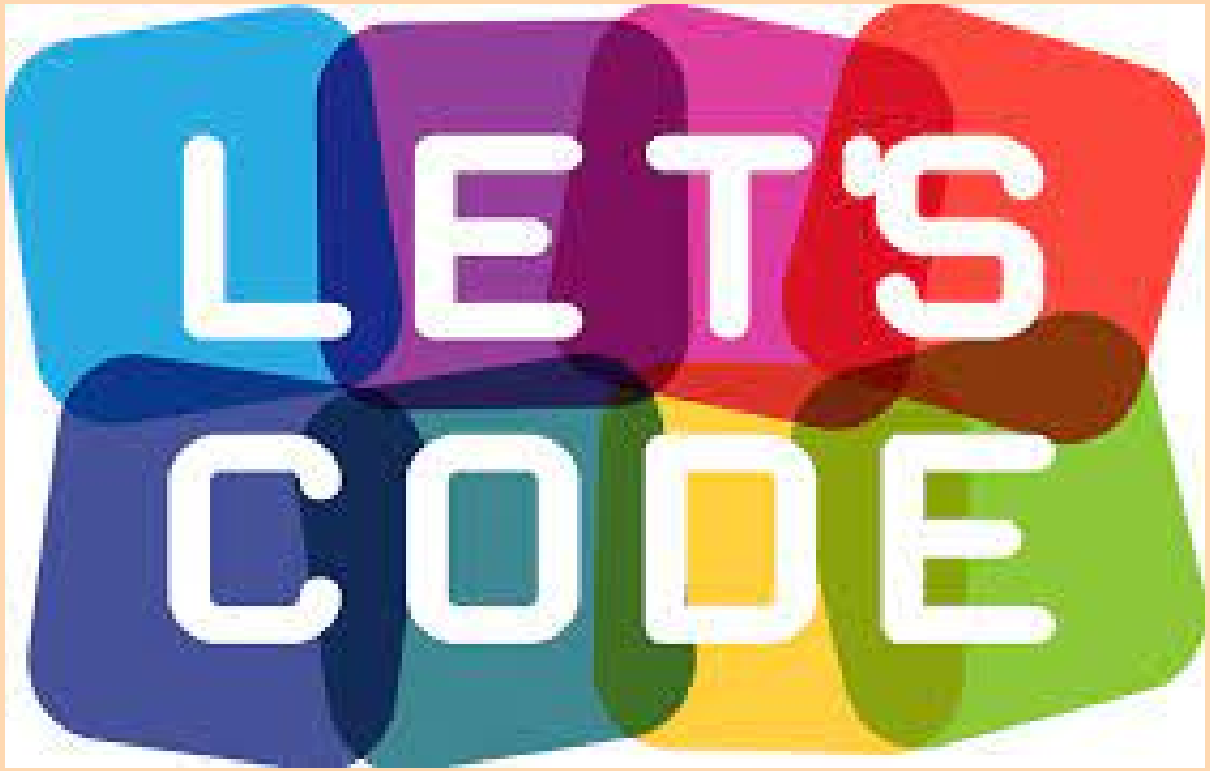
Overview

Variables (变量)　　Data Types (数据类型)

Operators (运算符)　　Expressions (表达式)

Assignment Statements (赋值语句)

I/O (Input/Output; 输入输出)



circle.c sphere.c mol.c

admin.c admin-scanf.c

Circle

Given a **radius** (say 10) of a circle,
to compute its **circumference** and **area**.

$$L = 2\pi r \qquad S = \pi r^2$$

- 每个结果各占一行
- 小数点后保留两位

Declaration/Definition (声明/定义)

```
int radius = 10;
```

- Declare/Define a *variable* called `radius`.
- The type of `radius` is `int` (integer).
- `radius` is initialized to 10.
- You can assign other integers to `radius`.
- `radius` refers to a location (`&radius`) in memory.

Identifiers (标识符)

```
int radius = 10;
```

`radius` is an *identifier*.

Warning: Do *not* start with `_`, which are reserved by C.

Always use meaningful identifiers in a uniform style!!!

Operators, Expressions, Assignment Statements

```
circumference = 2 * PI * radius;
```

Sphere

Given a radius (say 100) of a sphere,
to compute its surface area and volume.

$$A = 4\pi r^2 \quad V = \frac{4}{3}\pi r^3$$

- 每个结果占 1 行
- 小数点后保留 4 位
- 每个结果至少占 15 字符, 左对齐
  - `_____ : surface_area`
  - `_____ : volume`

mol

6 克氧气的分子数是多少?

$$Q = 6/32 \times 6.02 \times 10^{23}$$

两种格式输出, 结果均使用科学计数法表示

- 第一行结果, 小数点后保留 3 位
- 第二行结果, 保留 5 位有效数字

A (Naive) Administration System

- Name (EN)
- Gender (F/M)
- Birthday (mm-dd-yyyy)
- Weekday (Xyz.)

- C
- Music
- Medicine

- Mean (.d)
- Standard Deviation (.dd)
- Ranking (%)

For 罗大佑 only:

- 每组信息占一行
- 各项信息使用 `\t` 间隔
- 各项信息遵循特定格式要求

`char` and `<ctype.h>`

	_0	_1	_2	_3	_4	_5	_6	_7	_8	_9	_A	_B	_C	_D	_E	_F
0_ 0	NUL 0000	SOH 0001	STX 0002	ETX 0003	EOT 0004	ENQ 0005	ACK 0006	BEL 0007	BS 0008	HT 0009	LF 000A	VT 000B	FF 000C	CR 000D	SO 000E	SI 000F
1_ 16	DLE 0010	DC1 0011	DC2 0012	DC3 0013	DC4 0014	NAK 0015	SYN 0016	ETB 0017	CAN 0018	EM 0019	SUB 001A	ESC 001B	FS 001C	GS 001D	RS 001E	US 001F
2_ 32	SP 0020	! 0021	" 0022	# 0023	$ 0024	% 0025	& 0026	' 0027	(0028	) 0029	* 002A	+ 002B	, 002C	- 002D	. 002E	/ 002F
3_ 48	0 0030	1 0031	2 0032	3 0033	4 0034	5 0035	6 0036	7 0037	8 0038	9 0039	: 003A	; 003B	< 003C	= 003D	> 003E	? 003F
4_ 64	@ 0040	A 0041	B 0042	C 0043	D 0044	E 0045	F 0046	G 0047	H 0048	I 0049	J 004A	K 004B	L 004C	M 004D	N 004E	O 004F
5_ 80	P 0050	Q 0051	R 0052	S 0053	T 0054	U 0055	V 0056	W 0057	X 0058	Y 0059	Z 005A	[005B	\ 005C	] 005D	^ 005E	_ 005F
6_ 96	` 0060	a 0061	b 0062	c 0063	d 0064	e 0065	f 0066	g 0067	h 0068	i 0069	j 006A	k 006B	l 006C	m 006D	n 006E	o 006F
7_ 112	p 0070	q 0071	r 0072	s 0073	t 0074	u 0075	v 0076	w 0077	x 0078	y 0079	z 007A	{ 007B	| 007C	} 007D	~ 007E	DEL 007F

A `char` is actually an `int`.

C string

```
char first_name[] = "Tayu";
```

A C string is a null-terminated (`\0`) sequence of characters.

String literal: 'T', 'a', 'y', 'u', '\0'

- char first_name[5] = "Tayu";
- char first_name[10] = "Tayu";
- ~~char first_name[2] = "Tayu";~~

Conversion Specification

`%[flags][width][.[precision]]specifier`

- `%d` : decimal `int`
- `%f` : `double`
- `%e %E` : `double` $(-d.ddd \text{ e} \pm dd)$
- `%c` : `char`
- `%s` : `C string`
- `%%` : `%`

It is up to you to ensure that the type of the actual argument matches the type expected by conversion specifiers.

Undefined Behavior (UB)

printf-error.c

%[flags][width][.[precision]]specifier

- −: left-justified (otherwise, right-justified)
- +: always begin with a plus or minus sign

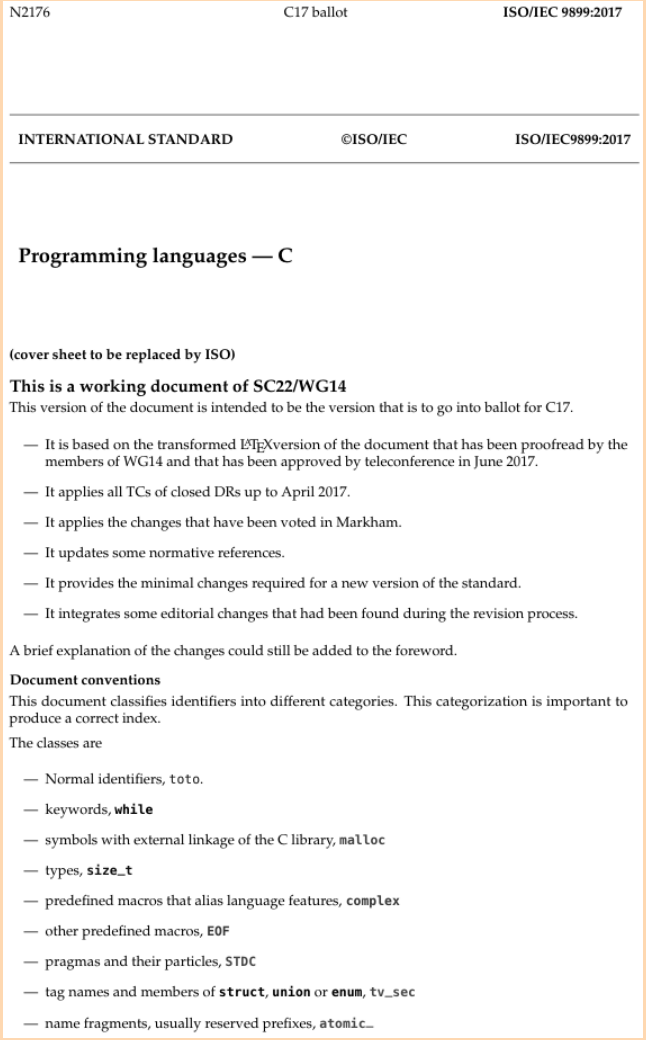
%[flags][width][.[precision]]specifier

- minimum field width
- padded with spaces if it has fewer characters

%[flags][width][.[precision]]specifier

- `%d`: minimum number of digits
  - expanded with leading zeros when needed
- `%f`, `%e`, `%E`: number of digits after `.`
  - default is 6
- `%s`: maximum number of characters

https://en.cppreference.com/w/c/io/fprintf



Programming languages — C

(cover sheet to be replaced by ISO)

This is a working document of SC22/WG14

This version of the document is intended to be the version that is to go into ballot for C17.

- It is based on the transformed LaTeXversion of the document that has been proofread by the members of WG14 and that has been approved by teleconference in June 2017.
- It applies all TCs of closed DRs up to April 2017.
- It applies the changes that have been voted in Markham.
- It updates some normative references.
- It provides the minimal changes required for a new version of the standard.
- It integrates some editorial changes that had been found during the revision process.

A brief explanation of the changes could still be added to the foreword.

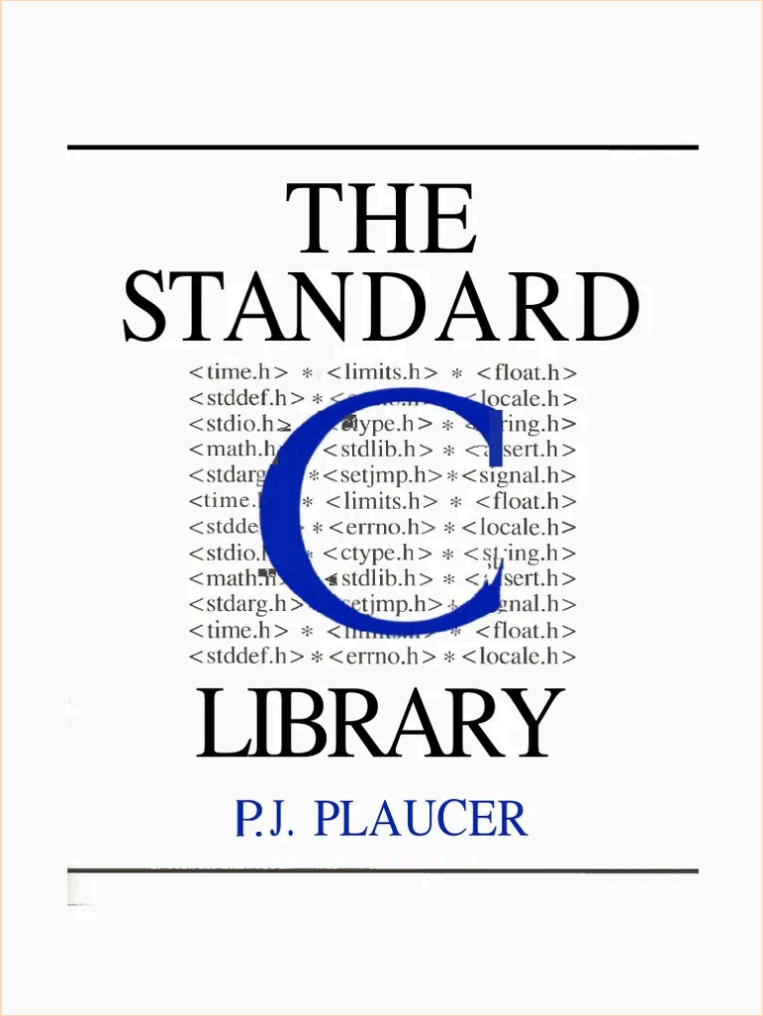
Document conventions

This document classifies identifiers into different categories. This categorization is important to produce a correct index.

The classes are

- Normal identifiers, `toto`.
- keywords, **`while`**
- symbols with external linkage of the C library, **`malloc`**
- types, **`size_t`**
- predefined macros that alias language features, **`complex`**
- other predefined macros, **`EOF`**
- pragmas and their particles, **`STDC`**
- tag names and members of **`struct`**, **`union`** or **`enum`**, `tv_sec`
- name fragments, usually reserved prefixes, **`atomic`**…

Section 7.21: `<stdio.h>`, P225--230



Chapter 12: `<stdio.h>`, P257--262


之乎者也
羅大佑
滾石唱片
RD-1008

`%[*][width]specifier`

- `%d`: skip white-spaces; match a decimal `int`
- `%lf`: skip white-spaces; match a `double`
- `%c`: match a `char` (do NOT skip white-spaces)
- `%s`: match a sequence of non-white-spaces
- `%%`: mathch a `%`

%[*][width]specifier

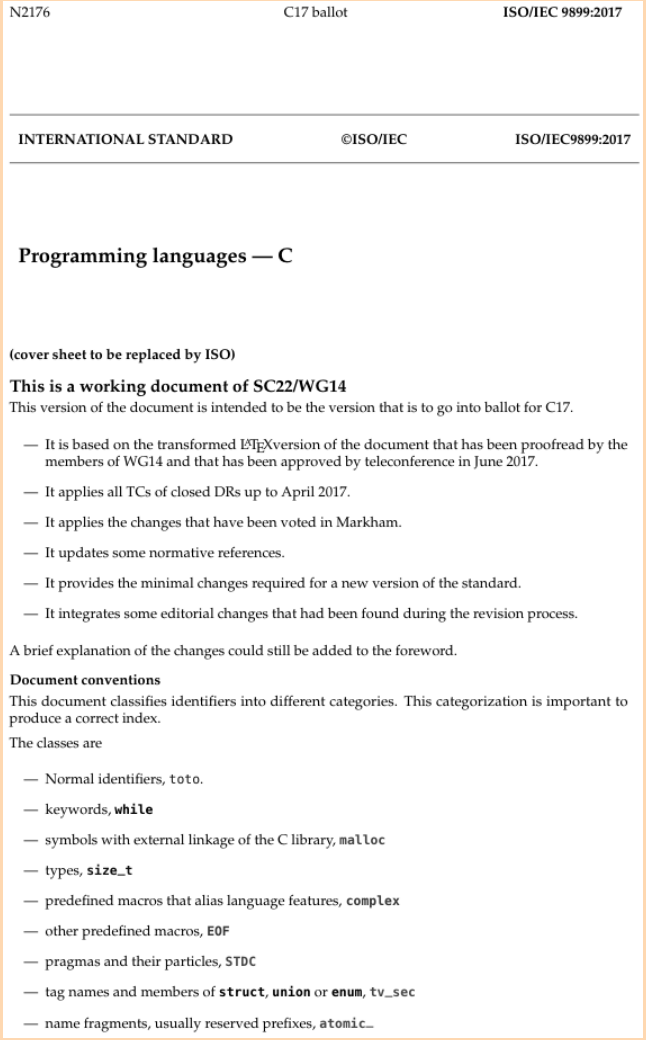
- *: assignment-suppressing

%[*][width]specifier

- maximum field width to scan

scanf-c17-ex2.c

https://en.cppreference.com/w/c/io/fscanf



N2176 C17 ballot ISO/IEC 9899:2017

INTERNATIONAL STANDARD ©ISO/IEC ISO/IEC9899:2017

Programming languages — C

(cover sheet to be replaced by ISO)

This is a working document of SC22/WG14

This version of the document is intended to be the version that is to go into ballot for C17.

— It is based on the transformed LaTeXversion of the document that has been proofread by the members of WG14 and that has been approved by teleconference in June 2017.

— It applies all TCs of closed DRs up to April 2017.

— It applies the changes that have been voted in Markham.

— It updates some normative references.

— It provides the minimal changes required for a new version of the standard.

— It integrates some editorial changes that had been found during the revision process.

A brief explanation of the changes could still be added to the foreword.

Document conventions

This document classifies identifiers into different categories. This categorization is important to produce a correct index.

The classes are

— Normal identifiers, `toto`.

— keywords, **`while`**

— symbols with external linkage of the C library, **`malloc`**

— types, **`size_t`**

— predefined macros that alias language features, **`complex`**

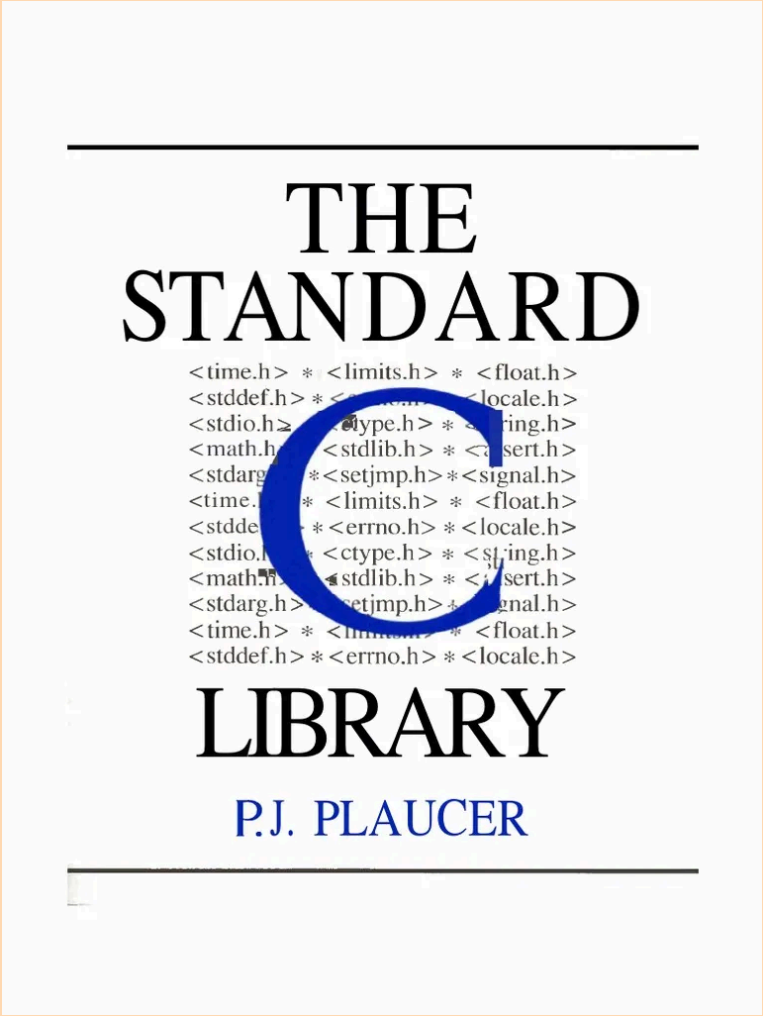
— other predefined macros, **`EOF`**

— pragmas and their particles, **`STDC`**

— tag names and members of **`struct`**, **`union`** or **`enum`**, `tv_sec`

— name fragments, usually reserved prefixes, **`atomic_`**

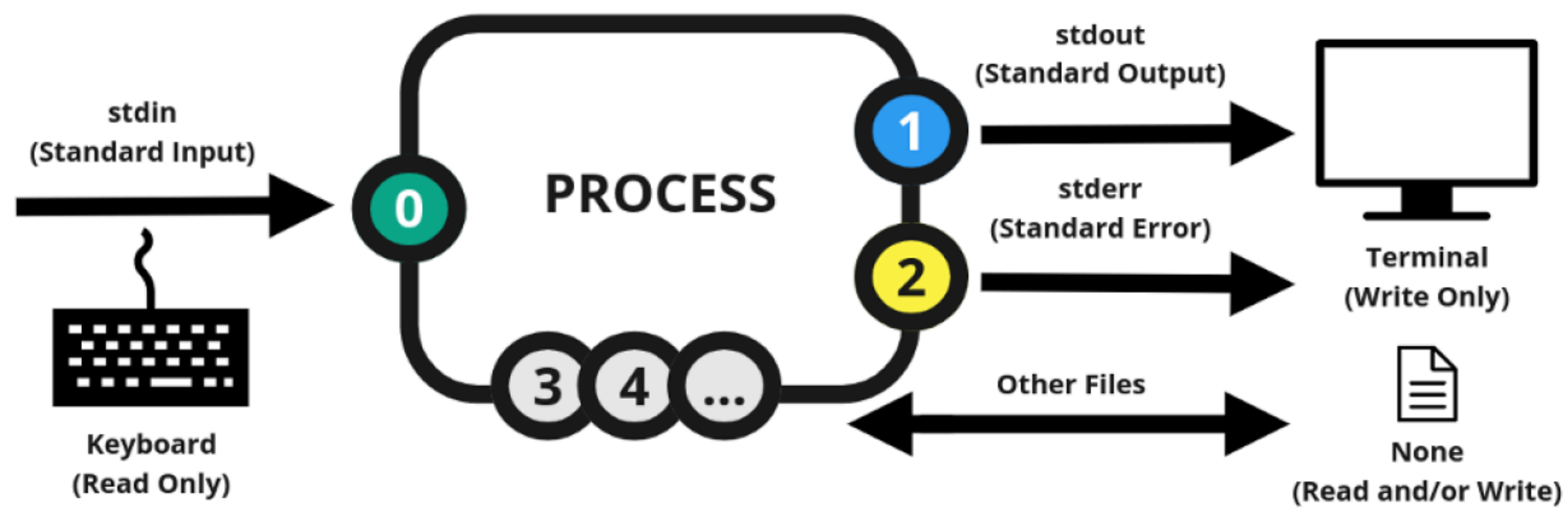
Section 7.21: `<stdio.h>`, P231--P237



Chapter 12: `<stdio.h>`, P263--P268

scanf-error.c

stdin, stdout, stderr



Input/Output Redirection

A beginners' guide away from `scanf`

Do NOT use `scanf`.

Why does everyone say not to use `scanf`? What should I use instead?


SEE
YOU
NEXT
WEEK