1 - Getting ready.

Computer's Do?

- Cpu (fetch memory and execute)
  - registers (workplace)
    - hold number
    - hold address
    - Keep Instruction
  - Instructions (MOV, PUSH, POP, ADD, CALL)

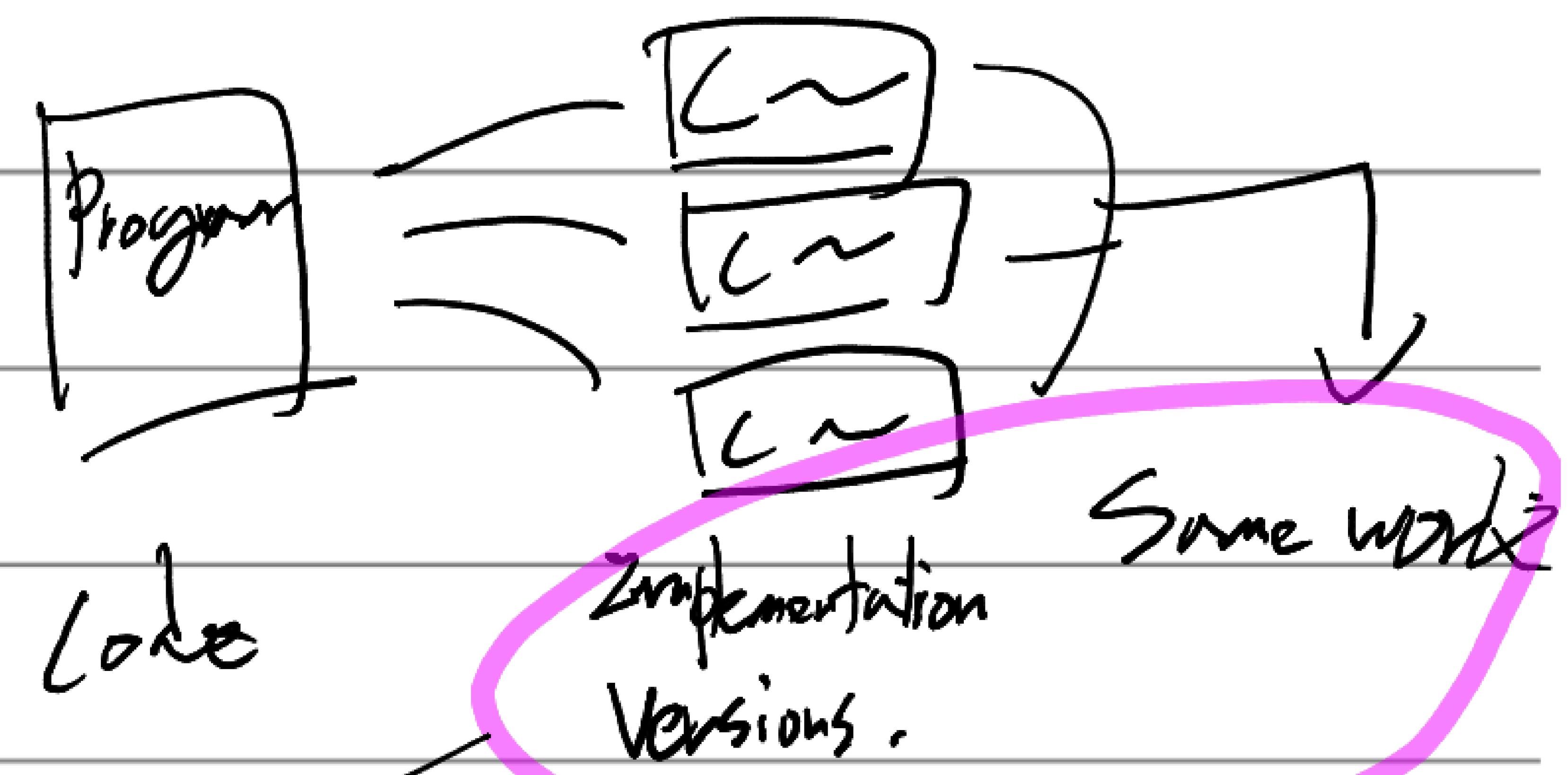
High-Level language and Compilers.

- no numeric code.
- Instruction are Complicated thing
  (not too much detailed to bother.)

Where Compiler
Do mean (translate)

Language standard.

now, many C Implementations
are available.



→ 이 Process가 성공적이려면
표준을 수행해야 한다.

1. KnR, Classic C → Language STD
2. ANSI/ISO standard → LIB STD

"그 어떤 언어는 이 상으로
라이브러리에 의존한다."

Standard

1. ANSI C (1989)
2. ISO C (1990)

] refered to C89, C90

ANSI (American National Standard Institute)

"keep these spirit of C" →

Target hard to problem to solve
not to uniform and abstract.

- Keep the language small and simple
- Provide only one way per operation
- make it fast (eventhough less portable)
- Trust programmer
  Let them do what should be

C99 standard (1994)

Internationalization
- ⊕ International charset support

Correction of deficiencies
- ⊕ figure out evident deficiencies.

Improve Computational usefulness
- ⊕ Improve critical numeric calculation

new tech trend
ie) - multi core
- concurrency

C11 standard (2007)
new guiding principles.

+. "trust the programmer" should be tempered for contemporary security concerns

Using C: 7 steps

Step	Result
1: Define program objective	Conceptual picture
2. Design a program	Clear map and plan
3. Write a Code	first product
4. Compile	result
5. Run a program	fact checking
6. Test and debug	Improvements
7. Maintain and modify	Finish of product.

1. Define program objective (목표 정의 단계)
('Clear idea of what program to do.')

☑ Think in general terms (no psudo code)

"Think in terms of..."
- Information to handle
- calculation and manipulation
- what to report back

by the program.

2. Design a program (어떻게 풀어갈 것인가?)

Components
- user Interface
- Structure of program
- User target
- Dead line, Scedule check

☑ Think in general terms (psuedo code)

+ how to represent data
+ which method to process data

"어떻게 데이터를 보여줄 것인지를 잘 정의한다면, 디자인과 데이터 처리를 간편하게 할 수도 있다."

3. Write the code

Clear design
- feat - method
- flow and architecture

→ (yes) → ? Implementable → then, → do Translate to Code

"where we have to put knowledge of C to work."

Translate Process

1. understand
2. Think
3. write

↓

"이 프로젝트의 동작은 너의 프로그래밍 환경에 의존한다."

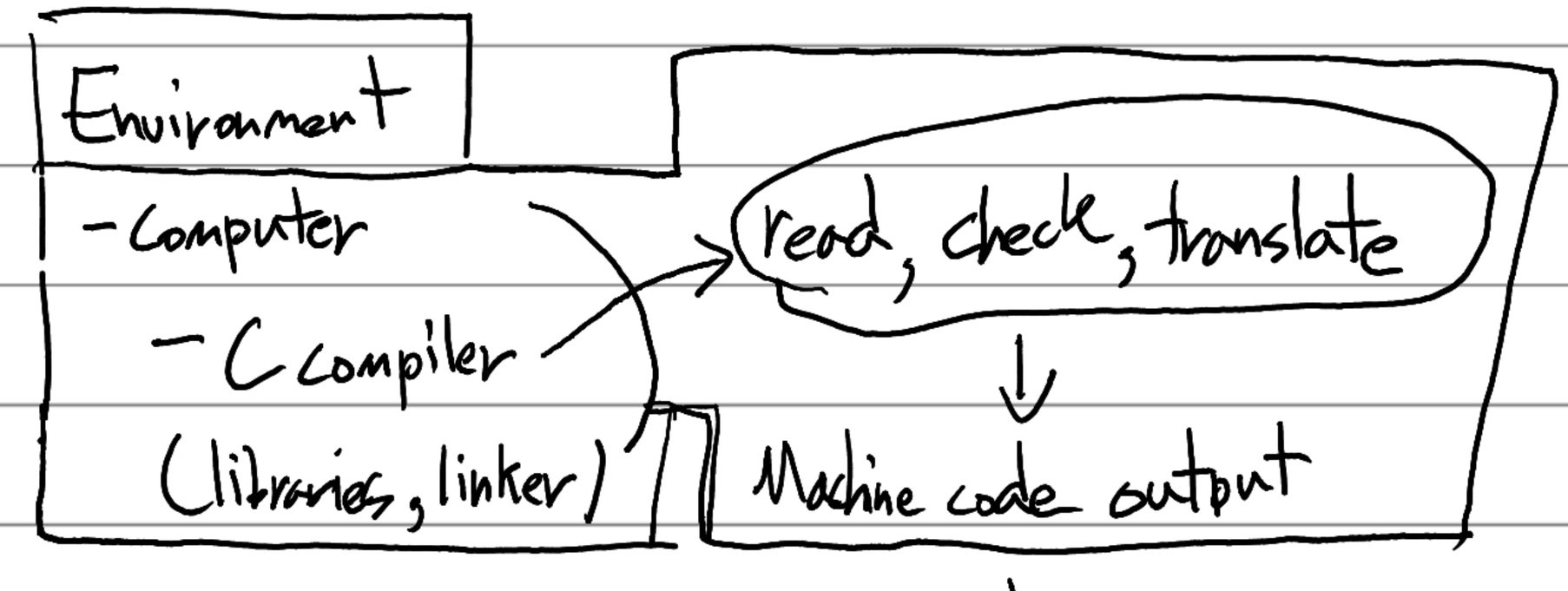
(text editor, Communication network, studies, habbits)

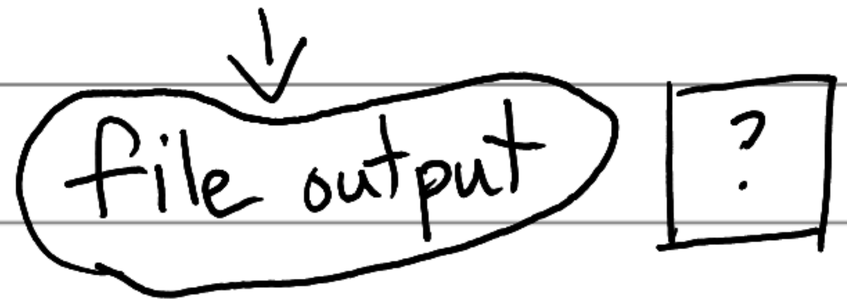
"이 과정에서 집중의 분산을 조심해야 한다." ie. - Comment (simplest way)
- Product document.

---

4. Compile

Code → in →





---

5. Run the Program : 'See it works.'

6. Test and Debug

"Making mistake" ← Natural part of learning.

learning ? ⟶ remind often of your fallibility.

Opportunity for Errors

- basic design error
- Implement error
- Overlook unexpected input error
- typing error
- wrong usage of c
- etc...

↳ handled by, 1. Compiler Catchable
2. Compiler Uncatchable (yourself)

---

7. Maintain and modify

"Document → Program" — clearly and
— followed sound design practices.

then,

fix and improve is fucking easy.

Commentary

사실 위에 단계들처럼 linear하고 단순한 과정은 아니다.
→ 실제로는 과정과 과정을 순환하는 것이 반복 된다.
→ 초보자들은 흔히 1,2(계획단계)를 무시하고 3(구현단계)로 가는 경향이 있다.
→ 쓰레기를 완성하는 과정으로 바뀌어간다.

"Develop the habit of planning before code."
- 시간을 벌게 되고
- 만족할 수 있게 된다.

"Use the pen and pencil to jot down the
- Object of program
- Outline of design."

Mechanics of C

Source code → Compiler → Object file (Modular system. (Instruction by programmer)) → executable file

Linker
+ Startup code (by OS dependant)
+ library code (Precompiled)
→ executable file

tip. 컴파일이 Linker 역할을 같이 수행하는 경우가 대부분이지만, Compile, Link 가 separate 인 시스템도 있다.

	GNU	VS	LLVM
Compiler name	✓ gcc	= =	✓ clang
start from	1987	<	✓ 2000
developed by	✓ GNU	>	Illinois Univ
adopted by	✓ Most of modern OS	>	FreeBSD at 2012, mac

Programmin exercise

1. define the program objectives and design the program

A program that converts inches to centimeters (1 inch = 2.54cm)

Company wants the program set up so that it

prompt user to enter inch value.

OBJ: read input return output

Design, 1. get input + str to number

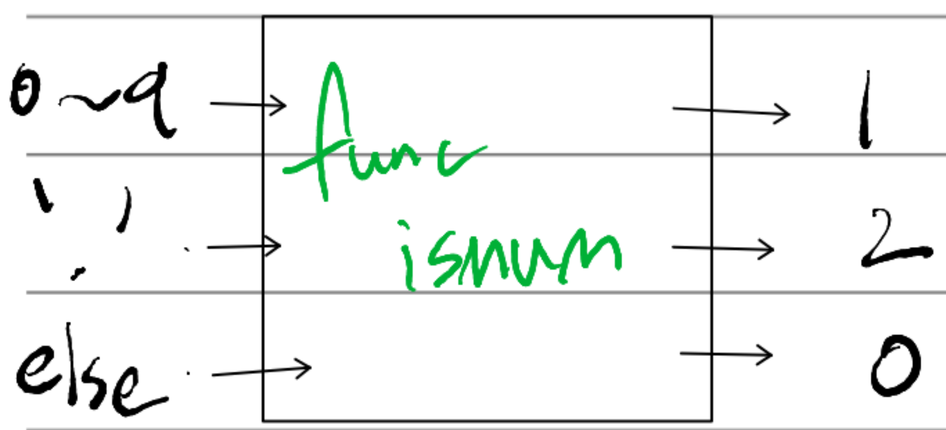
2. check is number or not ① func-1

L> not? return with no output.

3. covert it to centimeter value. ② func-2

4. output to user. + number to str

5. end prog.



while isnum

num -> getnum

If dot

while isnum

num -> getnum