→ 23 Aug 2023

Day 1:- Introduction to Programming.

→ Algorithm:-

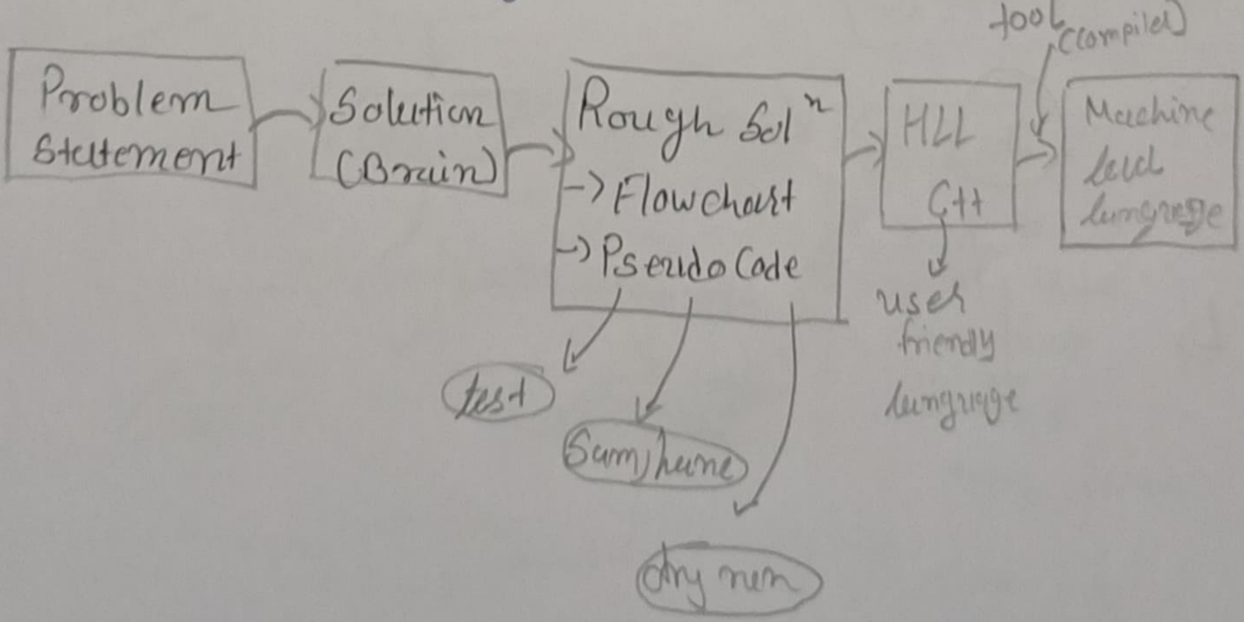
Sequence of steps to Sove a Problem

→ How to Approach a Problem?
(Thought Process)

Ⓐ Understand the Problem

Ⓑ input Values

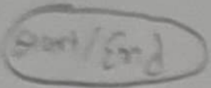
Ⓒ Create logic / Algorithm



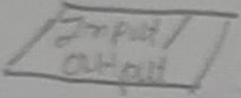
⟹ Flowcharts

A flowchart is a type of diagram that represents an algorithm, workflow or Process.

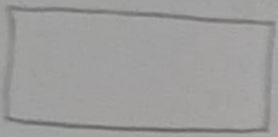
→ Components of Flowchart

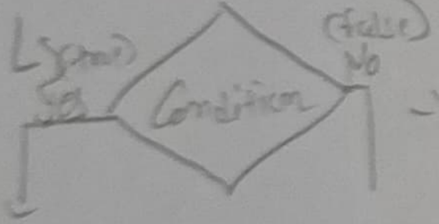
↳ 

⟹ terminator

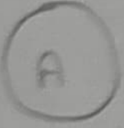
↳ 

⟹ I/O block

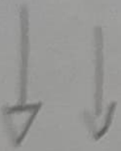
↳ 
⟹ Process block/
Calculation block
declaration/ initialization

↳ 

→ Decision making block.

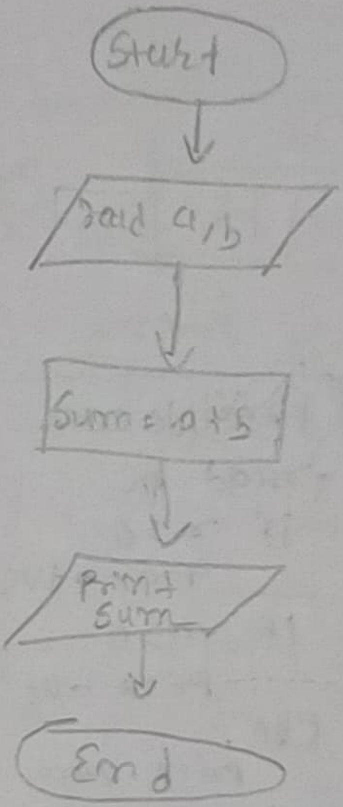
↳ 

→ Connector

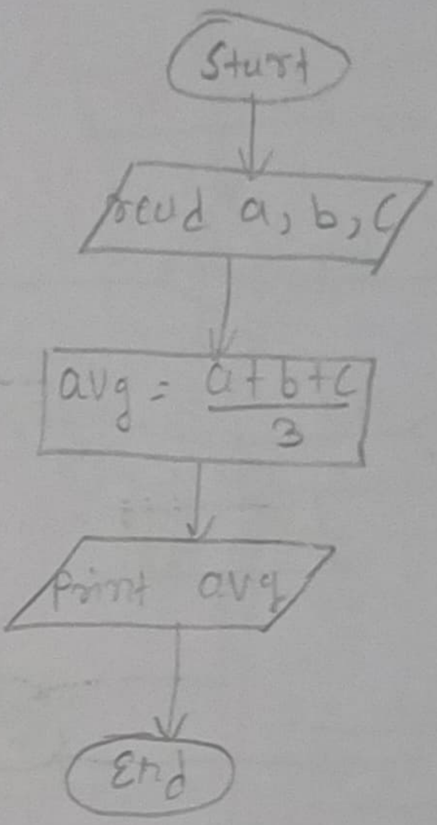
↳ 
→ Arrow
(flow of Execution)

Examples of flowchart

(1) Print Sum of a & b

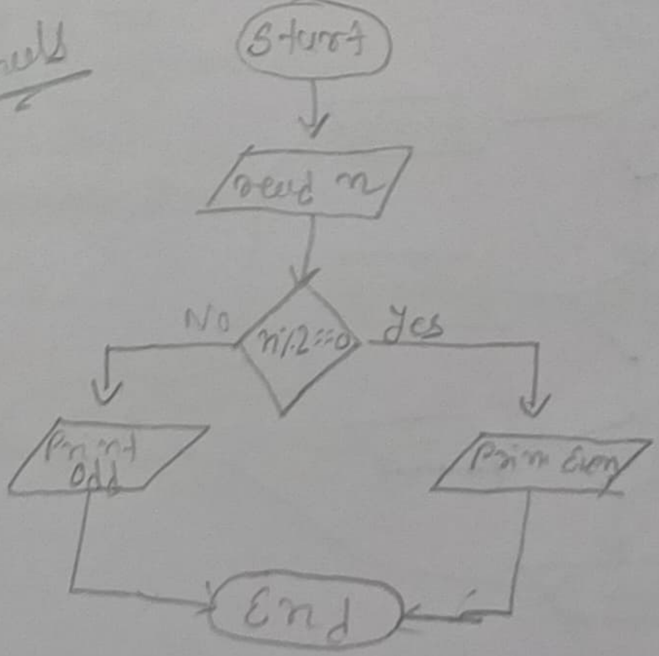


(2) Avg of a, b, c



(3) Check num is Even or odd

Conditionals

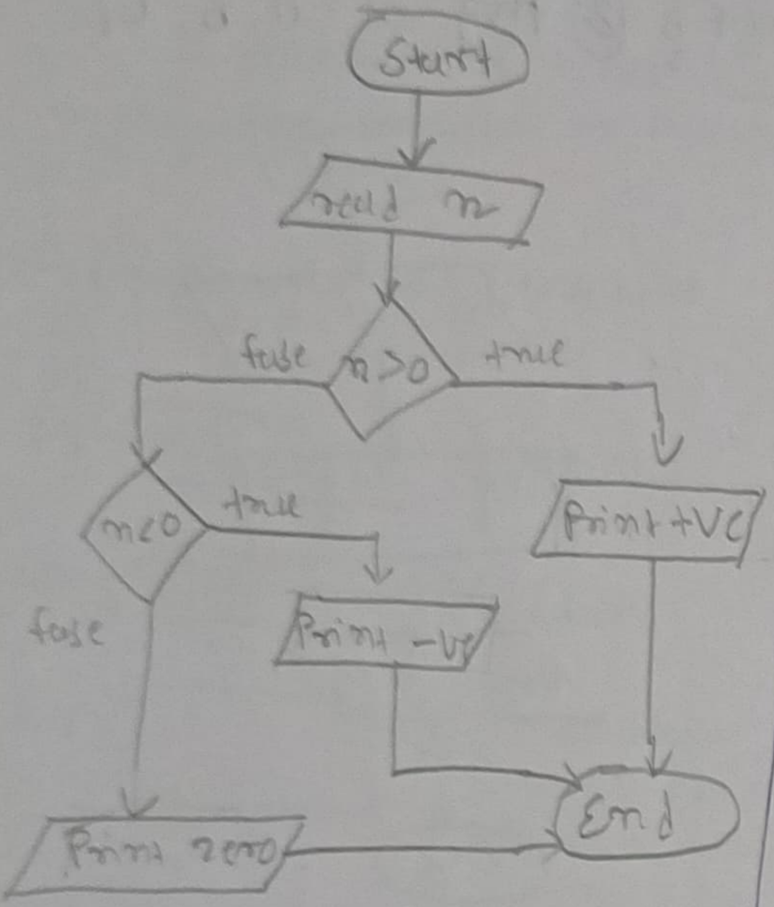


n → divid by 2
└→ rem = 1 = odd
└→ rem = 0 = Even

Pseudo Code

```
-> read n
-> initial value
     rem = n % 2
  if rem = 0
       Print Even
else
       Print odd
```

④ Check +Ve, -Ve or 0



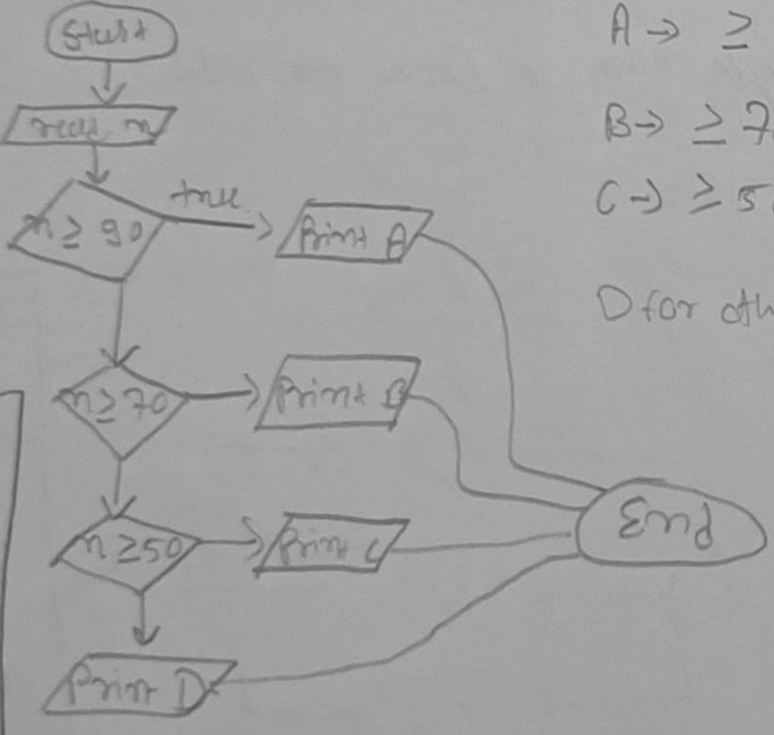
num = n

n>0 → true +ve
nik Jao
↓ false
13 < 0 → true -Vc
nik Jao
↓ false
Print zero

Pseudocode

```
-> read n
  if n>0
      Print +ve
  If n>0
      Print -ve
else,
    print zero
```

⑤ Student ~~ff~~ Grade Flowchart



rules

A → ≥ 90

B → ≥ 70

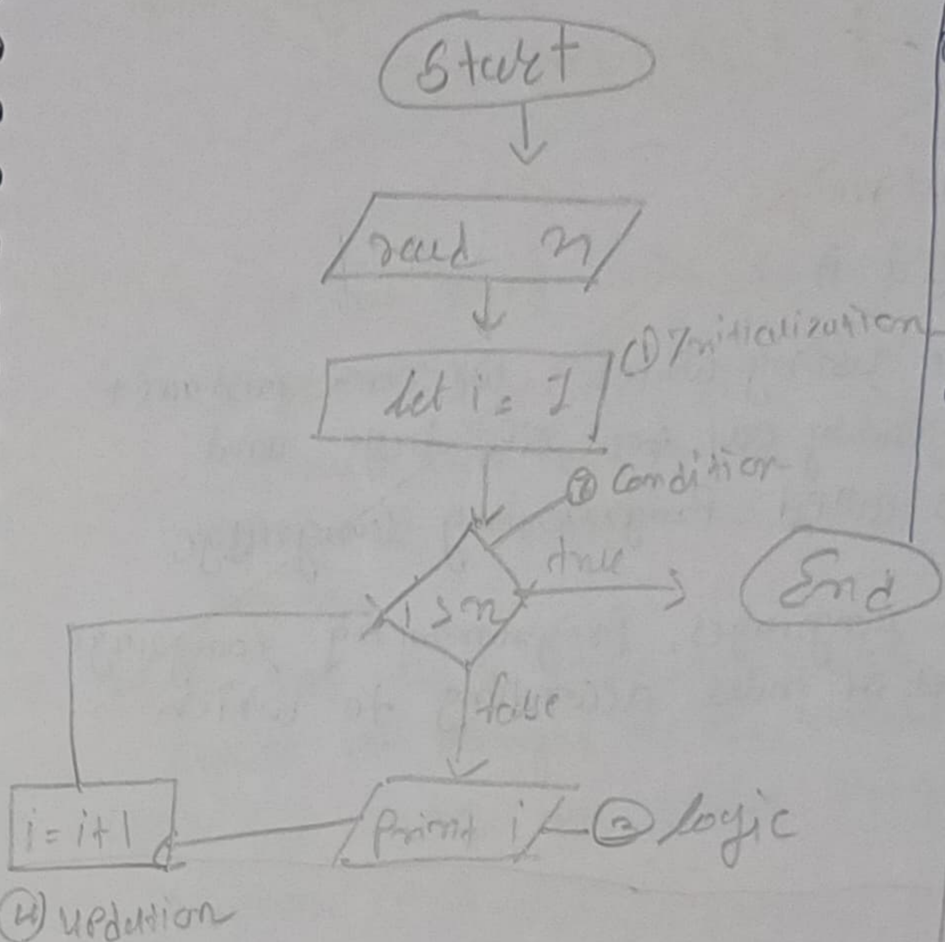
C → ≥ 50

D for others

sequence is Important

```
-> read n
-> if n>=90
      Print A
elseif n>=70
      Print B
else if
      n>=50
         Print C
else
     Print D
```

⑥ Print Counting from 1 to N → Looping



```
① read n
② initialize i = 1
③ if i <= n
      read a
        Print a
        i = i+1
④ repeat step 3
```

Dry run
n:4
i=1
1>4 → false
"1"
i=i+1 = 1+1 = 2
2>4 → false
"2"
i=i+1 = 2+1 = 3
3>4 → false
"3"
4>4 = false
"4"
i=i+1 = 4+1 = 5
5>4 → true
End

→ four steps of looping
1. Initialization
2. Condition
3. logic
4. updation

Pseudo Code:- fake Code

→ Day - 2

Write your first C++ Program

① Why do we need Programming language?

↳ A programming language which, uses can instruct that Computer to carry out real life tasks and computation is called a programming language

↳ It acts as a language which we could Easily Express our thought to the machine

↳ It has rules by which Programs could be written in it.

$$\begin{tikzcd}
\boxed{\text{High Level Language}} \arrow[r] & \boxed{\text{Compiles}} \arrow[r] & \boxed{\text{Machine Level Language}}
\end{tikzcd}$$

Note
Every lang has its own Compiler & interpretes

→ How Compiler work

$$\begin{tikzcd}
\boxed{\text{Sourse Code}} \arrow[r] & \boxed{\text{Compiler}} \arrow[r] & \boxed{\text{machine Code}} \arrow[r] & \boxed{\text{output}}
\end{tikzcd}$$

→ How interpreter work?

$$\begin{tikzcd}
\boxed{\text{Sourse Code}} \arrow[r] & \boxed{\text{interpreter}} \arrow[r] & \boxed{\text{output}}
\end{tikzcd}$$

②⟹

Compiler	Interpreter
↳ A Compiler takes the Entire Program in one go	↳ An interpreter takes a single line of Code at a time
↳ The Compiler generates an inter~~machine~~midiate machine Code	↳ The interpreter Never Produces any intermediate machine Code.
↳ The Compiler is best suited for the Production Environment	↳ An interpreter is best suited for a software development Environment.
↳ The Compiler is used by programming language C, C++, Java, C# etc.	↳ An interpreter is used by programming language Python, php, Perl, ruby etc.

③⟹ Where to Code ?

↳ Vs Code ↳ X- Code

↳ Code block ↳ Sublime

④ ~~Lets~~

④ Lets Write down first Code:

1) Code Excecution always Starts from int main() function.
2) int main() function is an inbuilt Method.
3) function is an entity/block of Code in which we Probvide inputs and we may get output.

$$\overset{\text{Return type}}{\text{int}}\ \overset{\text{function name}}{\overbrace{\text{main}}}\text{( ) \{}$$

```
int main() {
    - - - - - - --       } Scope
    - - - - - - - - -
}
```

⑤ => Print "Namaste Dunia"

① # include <iostream> → Pre Processor directial used to inclute the file.

② using namespace std; → Standerd Namespace

③ int main() {
    cout << "Namaste Dunia" << endl; → Semicolon is used to End of the line (terminate the line)
}

$$\underset{\text{used to Print}}{\text{cout}} << \underbrace{\text{"Namaste Dunia"}}_{\text{String}} \underset{\text{its keyword to Enter the Coursor in the new line}}{\overset{\text{Standerd to run}}{<<\text{endl}}};$$

* Endl - new line / Next line
* "\n" - new line Charactor
* // - Comment
* Cin - taking input in C++ →

```
int a;
cin>>a>>endl;
cout<<a<<endl;
```

⑥ → Variable and Datatype:

Vasiable: It is a Containes that is used to store the data Values:

1) Vasiable Cam store some information
2) uses cam use that information lates
3) uses cam Change that information later
4) a Vasiable name given to memory location
5) all the opesations done on the vasiable Effects that memory location.

Example

int → 4 bytes
↳ int a = 10;

Name of memory location

a $\boxed{10}$

4 byte block in memory

C++ Data type : type of data + Size

① Built in/Premitive
- ↳ integes :- int, long, short
- ↳ float - float, double
- ↳ Charalter - Chas
- ↳ Boolean - bool
- ↳ void

② Derived
- ↳ Arrays
- ↳ Pointess

③ uses defined
- ↳ Structuses
- ↳ Unions
- ↳ classes
- ↳ Enumesations.

Note:-
Data types ⇒ ① which type of data
② Size of data

↳ Size of Data types are Machine Dependent.

Datatype	32 bit	64 bit
char	1 byte	1
Short	2	2
int	4	4
long	4	8
long long	8	8
double	8	8
float	4	4

to find size of variable types by using the Method size of().

↳ int a;

```
cout<<sizeof(a)<<endl; // 4 byte
```

Note
Smallest addressable memory size at least 1 byte Hona Chahiye.

$\left.\begin{array}{l}\text{true} = 1\\ \text{false} = 0\end{array}\right\}$ 1 bit

0	0	0	0	0	0	0	1

In case of true.

⇒ Operators:-

1) Arithmetic → +, -, *, /, %
2) Relational → <, >, <=, >=, !=, ==
3) Assignment → =
4) Logical → &&, ||, !
5) Bitwise → &, |, ^, ~

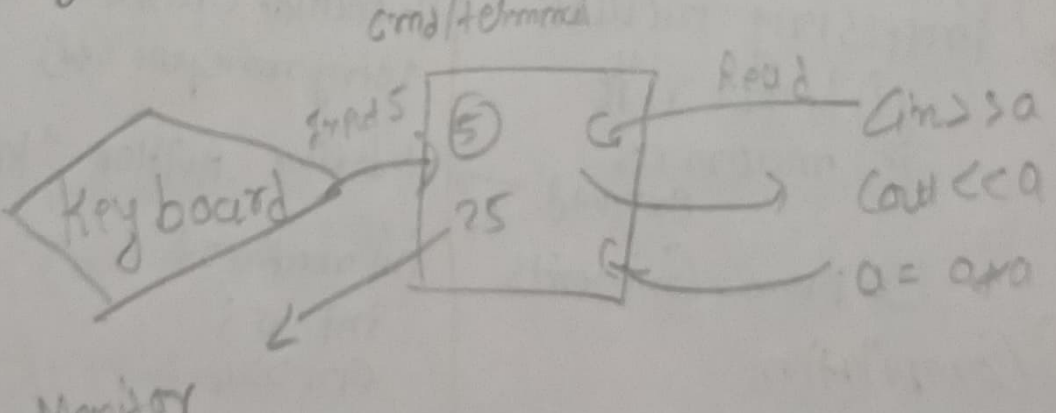
⟹ Compilation Process

Tesminal / CLI (command line Intesface)
Intelect aith foldes

To run a Program when
g++ main.cpp <> .exe file
uses/bin/g++ created

When you take an input from user through keyboard iostream has definition about Cin, Cout.

Physical world



iostream Contains Command which lets os enable Printf & cout backword Compatability

.CPP → .exe → Windows
.out → Unix, Linux, Mac OS

g++ → Uses/bin/g++
↓
Compiler

directly
OS
↳ out
↓
return 0

indirectly
P1 P2
OS my pro Yours pro
return 0

g++ main.CPP
↳ textfile
↓
Program → g++ Convert Program into .exe .out
↓
.exe

A ① Processing include
① header file
② macro as expand →
③ Remove Comments

B ② Compilation
Preprocess Code
↓
Machine Code

```
# include <iostream>
using namespace std;

#define AUTHOR "KARAN"

int main(){
    int a;
    cin >>4
    Cout << "written by"
                 <<AUTHOR;
```

① Lexical Analysis

$$\begin{tikzcd}
\text{"int"} \arrow[d] & \text{"a"} \arrow[d] & \text{"="} \arrow[r, bend left] & \text{Operator} \\
\text{Keyword} & \text{Identifier} & \text{5 ;} \arrow[r] & \text{Constant}
\end{tikzcd}$$

② Parsing ⇒ tokens to Syntax tree

$$\begin{tikzcd}
\text{int} \arrow[d] \\ \text{a} \arrow[d] \\ \text{=} \arrow[d] \\ \text{5} \arrow[d] \\ \text{;}
\end{tikzcd}$$

int a=5;

Syntax error { int 5=a

③ Symantic Analysis

check meaningfullness

type mismatch
↓
gives error → int a= "Karan";

undeclared Variable Check.

int a=5;
Cout << b << endl;

④ Intermediate Code generation

.asm
assambly format

int a=5; $\boxed{\begin{array}{l}\text{Var a;}\\ \text{move 5 a;}\\ \text{Print} \to \text{A}\end{array}}$

Processor dependent

⑤ Optimization

Remove Dead Code

for ex Data, That is not being used

Ⓒ object file generation.

$$\begin{tikzcd}
1.cpp \arrow[d] & 2.cpp \arrow[d] & 3.cpp \arrow[d] & \\
1.o & 2.o & 3.o \arrow[r] & Unix \\
1.obj \arrow[d] & 2.obj \arrow[d] & 3.obj \arrow[d] \arrow[r] & windows \\
Compiled & C & C &
\end{tikzcd}$$

Ⓓ Linking → Multiple obj / o's

$$\begin{tikzcd}
 & Multiple\ obj\ /\ o's \arrow[d] & \\
 & Combine \arrow[d] & \\
windows & final\ executable \arrow[d] & mac \\
.exe \arrow[u, bend left] & & .out \arrow[u, bend right]
\end{tikzcd}$$

$$\begin{tikzcd}
1.cpp \arrow[r] \arrow[d] & cin,\ iostream & & main.cpp \arrow[d] & \\
1.o \arrow[d] & & & Compile \arrow[d] & .exe \\
cin \arrow[uur, bend right, "link"] & & & .o \arrow[d] & Linking \arrow[u] \\
 & & & Result \arrow[r] & machine\ code \arrow[u]
\end{tikzcd}$$

g++ → mac X

g++ → wind ✓