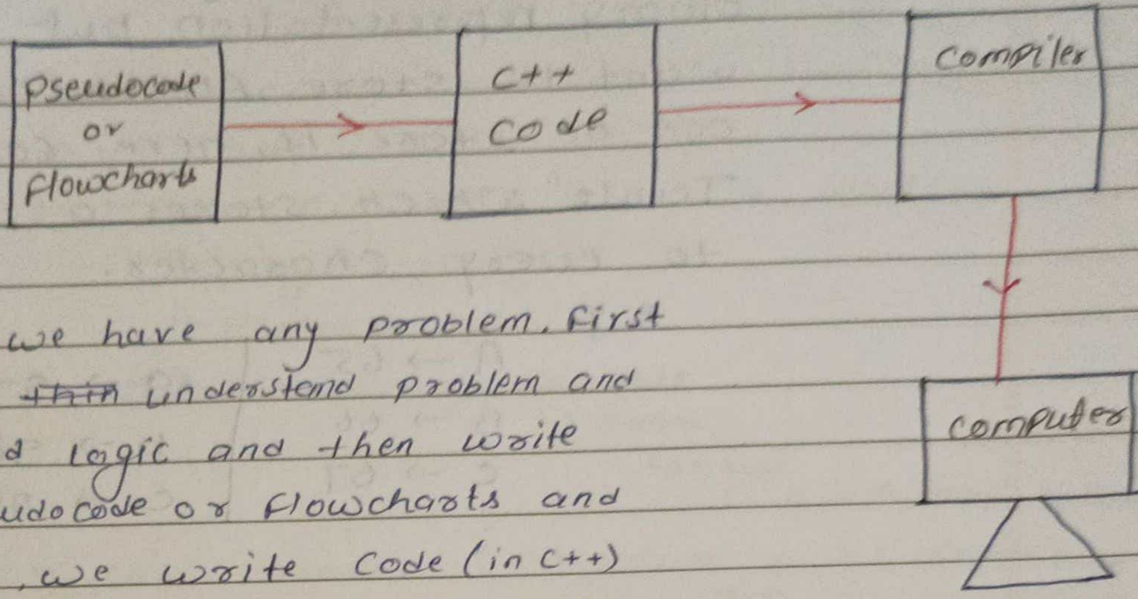
Date 20/09/2023
Page

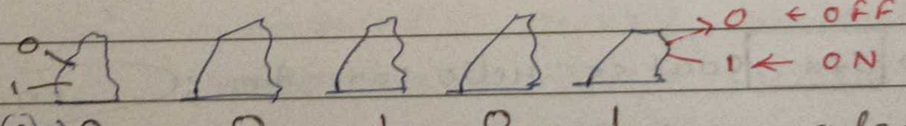
starting with C++

computer memory unit



→ if we have any problem, first we ~~thin~~ understand problem and build logic and then write pseudocode or flowcharts and the, we write code (in C++) and that code is given to compiler to convert that HLL code to machine level language so that computer can understand and perform actions.

⇒ use of compiler
- convert code HLL to MLL
- check error in code
- optimization



Transistors (s) → 0_{2^4} 0_{2^3} 1_{2^2} 0_{2^1} 1_{2^0} → computer understand only Binary.
1 Bit →

1 Bit → Binary representation

8 Bit = 1 Bytes

(2^{10}) 1024 bytes = 1 KB (Kilo bytes)
(2^{10}) 1024 KB = 1 MB (Mega bytes)
(2^{10}) 1024 MB = 1 GB (Gega bytes)
(2^{10}) 1024 GB = 1 TB (Tera bytes)

e.g:- 4 → 100
10 → 1010

if we have to store numeric data we can store it by converting into binary representation. but if we want to store Alphabet then how can be store it, here comes "ASCII Table" which stores a unique no. to every character.

e.g.

A → 65		a → 97	
B → 66		b → 98	
C → 67		c → 99	
⋮	⋮	⋮	⋮

Refer: ASCII Table (from internet).
(American standard code for information Interchange)

Write first code in C++.

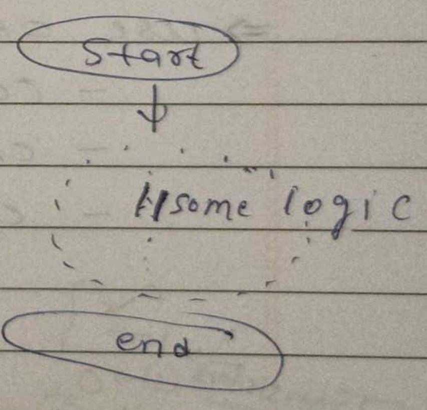
```
#include<iostream>
using namespace std;

Start ->int main()
{
  std::cout << "Hello coder Army";
  cout << "OM";
};
  ↖ end
```

if we do not write "using namespace std" then we have to write.

if we use "using namespace std" ~~in header then we~~ don't need to write std::cout << ...

Flow charts



```
                                   output
Cout<< 2+3 ;                 ← 5
cout<< 6×8   ;               ← 48
coul << "2+3" ;              ← 2+3
cout << "6+8" ;              ← 6+8
```

if we write anything in " "(double cotte") the that statement will be printed as it is.

software needed to run C++ code

1. VS code
2. MinGW - Minimalist GNU For Windows
   i. Mingw32-gcc-g++
   ii. mingw32-base
3. set environmen variable
   ↳ path
      ↳ new
         ↳ C:\MinGW\bin
4. code runner (extention in VS code)

All the above steps is for windows.

⇒ To print in C++,

#1
```
cout<<"Hello Coder Army";
cout<<"Hello coder Army";
```

output :

```
Hello coderArmyHello coder Army.
```

to print in every statement in new line we have two ways.

(i) `cout<<"Hello coder Army"<< endl;` ← end line

(ii) `cout<<"Hello coder Army \n";` ← new Line

Refer :

Variables & Datatypes

in Real Life we use
Alphabet/char : a, b, c, ...
Number : 1, 2, 3, 4, 5.....
Word : How are you, ...
Yes or No (Gesture)

In ~~the~~ same way to make understand above example ~~to~~ our computer we have concept of variable & Datatypes.

for :

Number :	INT	: 1, 2, 3, 4 ...
	Float	: 1.2, 2.8, 3.68
	double	: 1.234,
Alphabet :	Char	: a, b, c ....
word :	String	: "Hello" .....
Yes, No :	Boolean	: 1 or 0.

(1.) Integer

int name = 10 ;

↑ data type　↑ Variable　↑ Assignment operator

$\boxed{10}$

Name ← 4 bytes.

4x8 = 32 bit
4 Bytes = 32 bit

$\underbrace{0\ldots\ldots 0000000000000000000}_{28\ bit}\overset{2^3}{1}\overset{2^2}{0}\overset{2^1}{1}\overset{2^0}{0}$

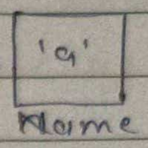
(2.) Character:

char Name = 'a';

1 Byte Memory 'b', '1', '%', '*'.

8 bit

'a' = 97 → 0 1 1 0 0 0 0 1 →



Variable Naming Rule:

64
32
96

1. A-Z or a-z
2. 1-9
3. _

variable name can be of any combination of (A-Z), (a-z), (0-9) & _ (underscore) but with one condition, that variable name should not start with numeric value. i.e (0-9)

e.g

A_bc ✓		aA2 ✓
_Ab ✓		_AB ✓
_a ✓		1AB X
a ✓		
2ab X		

* variable name should be meaningful.

e.g.: int homeLoan = 100; ✓

char p = 'aa'; X

↗ Data type ↑ Variable name ↖ Character can store a single character.

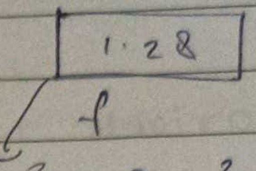
③ float

float f = 1.28;

datatype ↑ variable name ↑

↓

4 bytes = 32 bit



0 1..?..00...? 0
Binary representation.

④ double

double d = 4.23456789...;

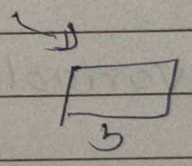
8 bytes = 64 bit

Large decimal place.

⑤ Long

Long int b = 298763458

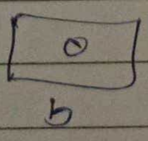
8 byte = 64 bit.



⑥ Boolean

~~boolean~~ bool b = 0
= 1
= true
= false

1 Byte = 8 bit



Comments

// ← two slace represents comments in our program. comments is used for documention. and that will be ignored by the compiler.

e.g

```
#include<iostream>
using namespace std;
int main()
{
    int a=10; // variable declaration
    int b;
     b = 10;
    int a, b. c;
    a =10;
    b = 20;
    c = 30;
    cout<< a+b << endl<< a+ c << endl ;
}
```

Negative Number

int a = 23, 57, 128;
= (-5), (-6)

Signed bit ↓ ↓

_ _ _

-2^2 $0 - 2^2 - 1$

-4, -3, -2, -1 0, 1, 2, 3

+ve → 0 _ _ _

-ve ← 1 0 0	= ~~0~~ -4	0 0 0	= 0
1 0 1	= ~~-1~~ -3	0 0 1	= 1
1 ~~1~~ 0	= ~~-2~~ -2	0 1 0	= 2
1 1 1	= ~~-3~~ -1	0 1 1	= 3

$-2 \longrightarrow 010$

↓

~~10~~

~~1's → 01~~

~~Complement~~

~~2's~~

1's compl. $\underset{-ve}{1}\,0\,1$

2's " $+1$

$\underline{1}\ \underline{1\,0} \leftarrow \boxed{-2}$

$-3 \longrightarrow 011$

1's $\underset{-ve}{1}\,0\,0$

2's $+1$

$\underset{-ve}{\underline{1}}\,0\,1$

4 bit

$-2^3 \qquad +2^3$

$-2^3, \dots -1 \qquad 0 \dots (2^3-1)$

$\underset{-ve}{1}01 \longrightarrow 101$

1's 010

2's $\overset{+ve}{}\ +1$

$\underline{011}$

↓

$\boxed{-3}$

$10101 \longrightarrow \underset{-ve}{1}\ 0101$

1's $\rightarrow 01010$

2's $+1$

01011

$\boxed{-11}$

$\underset{+ve}{0}101 \rightarrow \boxed{+5}$

32 bit

$2^{31} \swarrow \quad \searrow 2^{31}$

$-2^{31} \dots, -2, -1$

$0 \dots 2^{31}-1$