Programming

Data

input / output

Variables (naming rules)

memory buckets

Data Types

int, float, char (string), bool

Instructions

① - input / read

② - output / assign / set

③ - perform operation

④ - Conditionals

⑤ - Loops

chain nest

Patterns

different operations (+, -, /, *, %, &&, ||, !)

input / output —

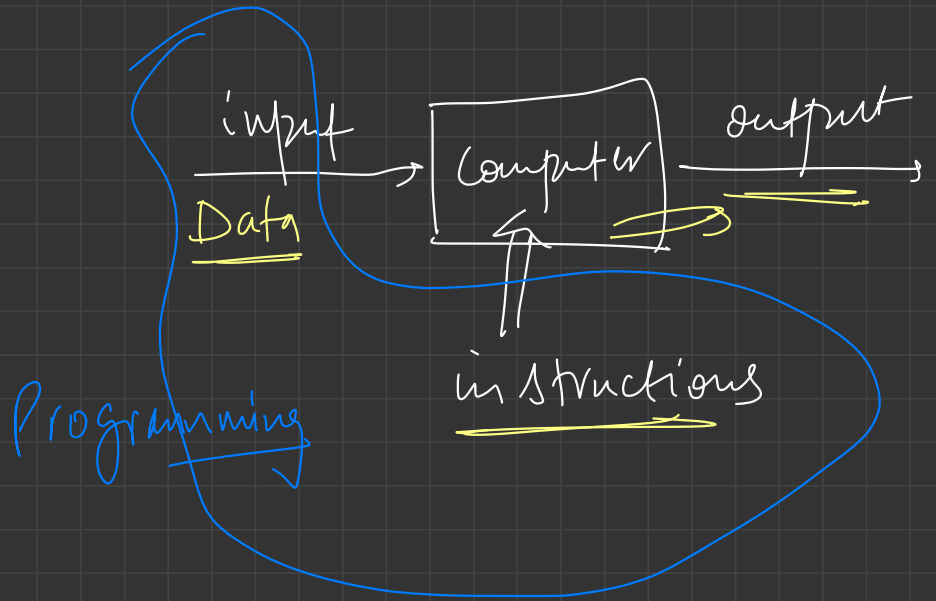
Loops ⇒ Patterns

I/O

Conditionals

Last class

Problem solve using computers



Programming (C++) → language we will use to instruct the computer

Data

Instructions

translator → compiler

translator → (0, 1) → Binary

Variables

abc → store → a location in the memory

→ variables naming rules –

① only consists of (letters, digits, _ (underscore))

② start with a letter or _ (digit)

(3) no spaces

→ keywords can't be variables

- include
- namespace
- while
- for
- int, char - _

✗ can't be used as variables names

any data (we take as input)

↳ is stored in a bucket called variable

↳ name

↳ datatype

Data Types

integers

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

int

decimal values

0.123, 2.341
-6.19, ...

float (double)

precision

character (symbol, a-z, A-Z)

0-9, !, @, #, $
:, _, ^, &, ...

char

boolean

0 false
1 true

bool

Computers only understand/know {0, 1}

How will computer understand?

integers ⇒ natural nos. 0, 1, 2, 3, 4, 5, 6, - - - - decimal system

↓ convert

Binary System

{ 1, 10, 11, 100, 101, 110, - - }

-1, -2, -3, -4 ↙

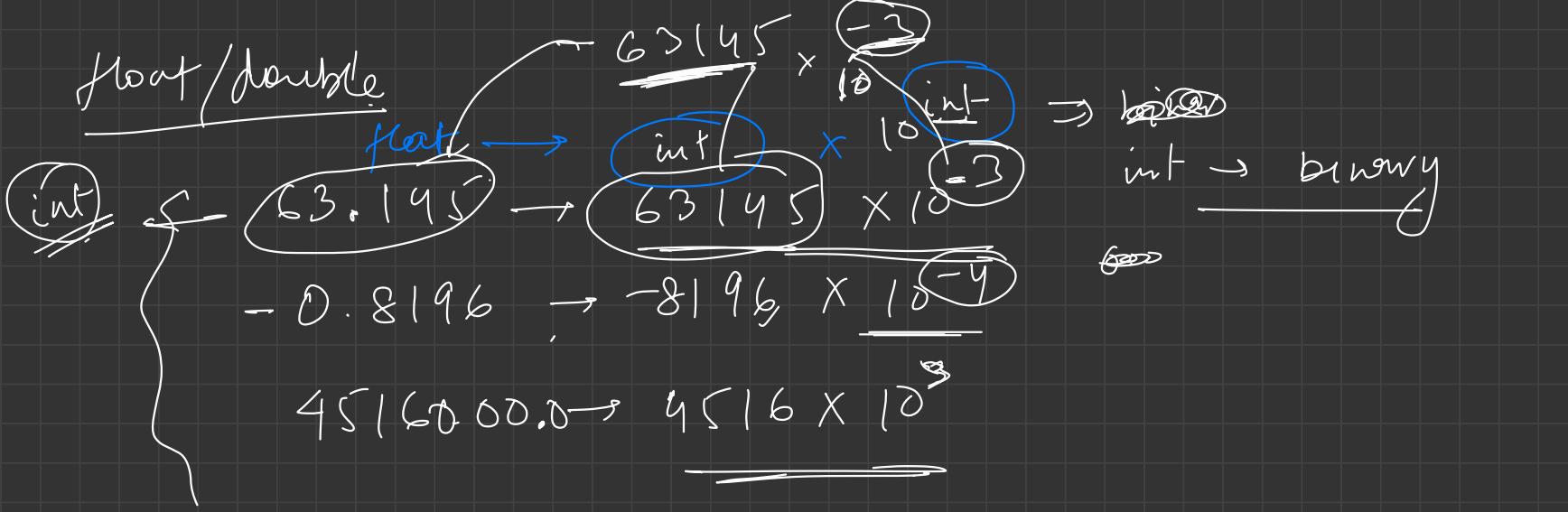
1 2 3 4 → 0100 → 1011 → 1100 → -4

Binary rep → complement (flip 0's and 1's)

add +1

(int) represented in binary!!

1 × (-3) → 0.001



Characters formula

Standard values (int) ⇒ characters

Using ASCII values (0-127)

Boolean

0 false

1 true

Instructions (orders)

→ read something (from input)
(from another variable)

→ write → output → to some variable

→ assign (giving some value to variable).

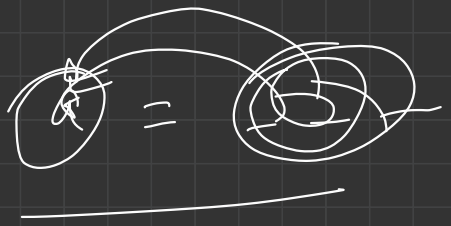
→ perform operations

arithmetic: (+, −, /, *, %)

logical op: (&&, ||, !)

$\sqrt{xy}$

= assignment operator.



relational op. == != > < >= <=
equality

((bit-wise)

→ Control Statements:

- Conditionals: if, else
- Loops: for, while

```
if (age is greater or equal to 18) {
    print (Adult)
} else (otherwise) {
    print (child)
}
```

Block-1

Block-2

Control Statements

Conditionals

doing something or not based on some condition

Loops

repetitive work

doing something again & again

if-else

if
else
for
while

Problem

1. if I give you age of a person (int) input

Can you tell me output whether he/she is an adult or a child? (age >= 18)

① take input

data
types
operations

logic based
↓
decision power

② decide
③ output.

Conditional Statements

If it rains today, we will not play.

↓

Rains Today (fact) → true ⇒ not play

→ false ⇒ play

CPP

if (condition → true) {
    do-this work;
}
else (false) {
    do-this work;
}

if (rains today) {
    not play;
}

dT vN : As syntax grammar of CPP

int age;
cin >> age;

Take as input marks of five subjects. Calculate the percentage and grade according to the given conditions:

If percentage < 40% : Grade F

If percentage >= 40% : Grade E

If percentage >= 60% : Grade D

If percentage >= 70% : Grade C

If percentage >= 80% : Grade B

If percentage >= 90% : Grade A

0-39

40-59

60-69

70-89

80-89

(90-100)

```
if(p>=0 && p<=39)
    cout<<"F";
if(p>=40 && p<=59)
    cout<<"E";
```

x >= 60

60, 61, 62, ---

```
if(p>=60 && p<=69)
    cout<<"D";
```

will it work??

Q. Print 1 to 50

solve?!

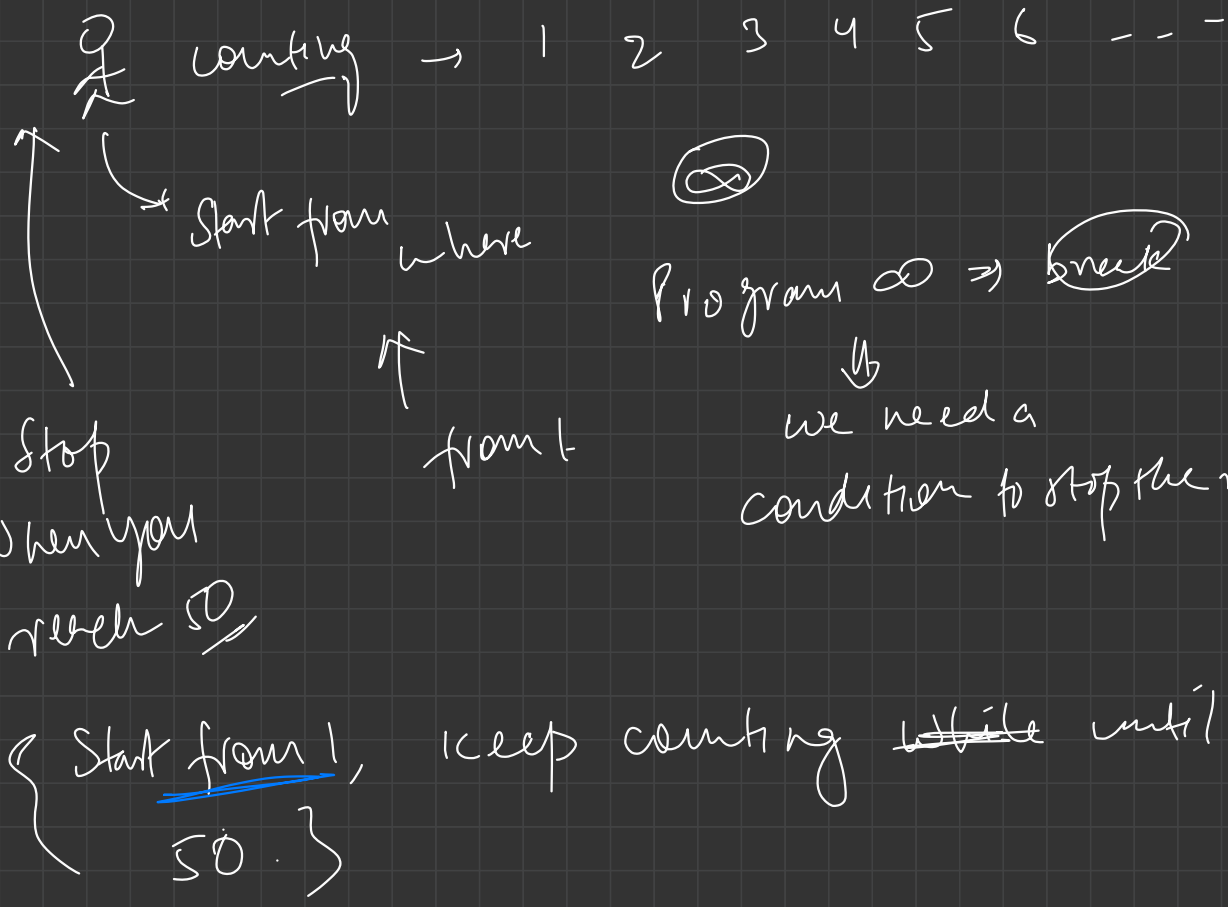
without using loops

(repetitive work)

1, 2, 3, 4, 5, 6...

+1 +1 +1 +1 +1

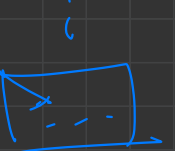
pattern


counting → 1 2 3 4 5 6 ---
Start from where
from 1
Stop when you reach 50
Program ∞ ⇒ break?
we need a condition to stop the repetition
{ Start from 1, keep counting ~~while~~ until you reach 50. }

CPP → start // initialization

end-condition

how to proceed??



CPP

```
int i=1;
while(i<=50){
    cout<<i;
    i=i+1;
}
```

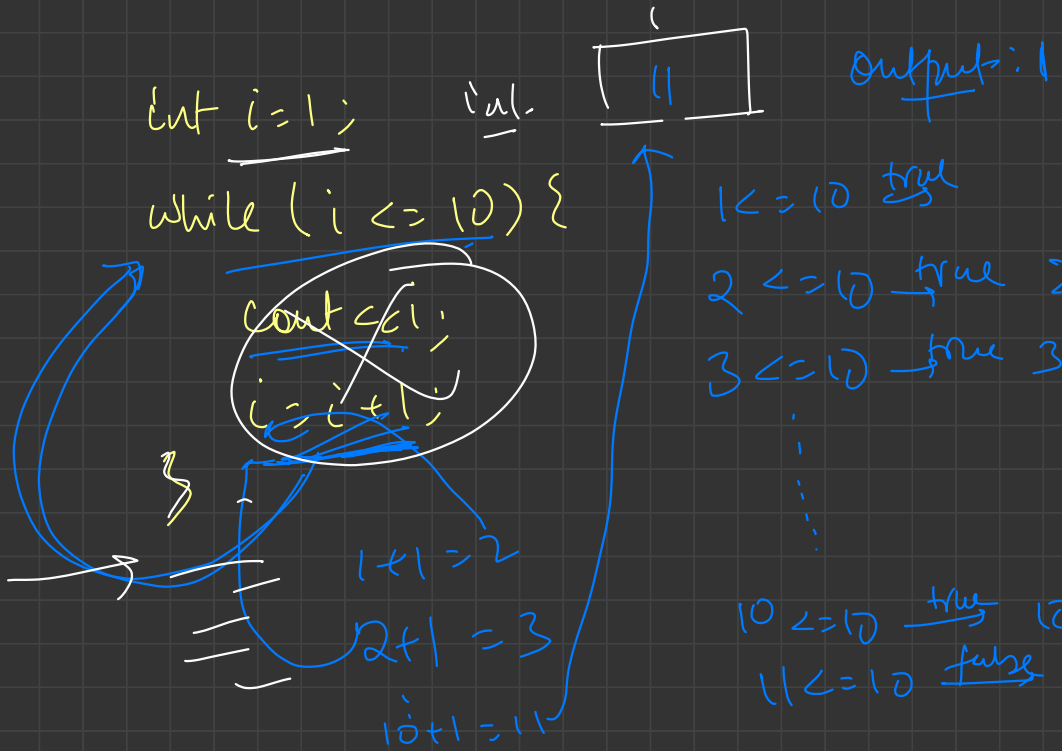
initial

repetitive part

true

incrementing

false



Q. Print odd numbers from 1 to 100.

①
```
int i=1;
while(i<=100){
    cout<<i;
    i=i+2;
}
```

②
```
int i=1;
while(i<=100){
    if(i%2==1)
        cout<<i;
    i=i+1;
}
```

③
```
int i=1;
while(i<=100/2){
    cout<<(2*(i)-1);
    i=i+1;
}
```

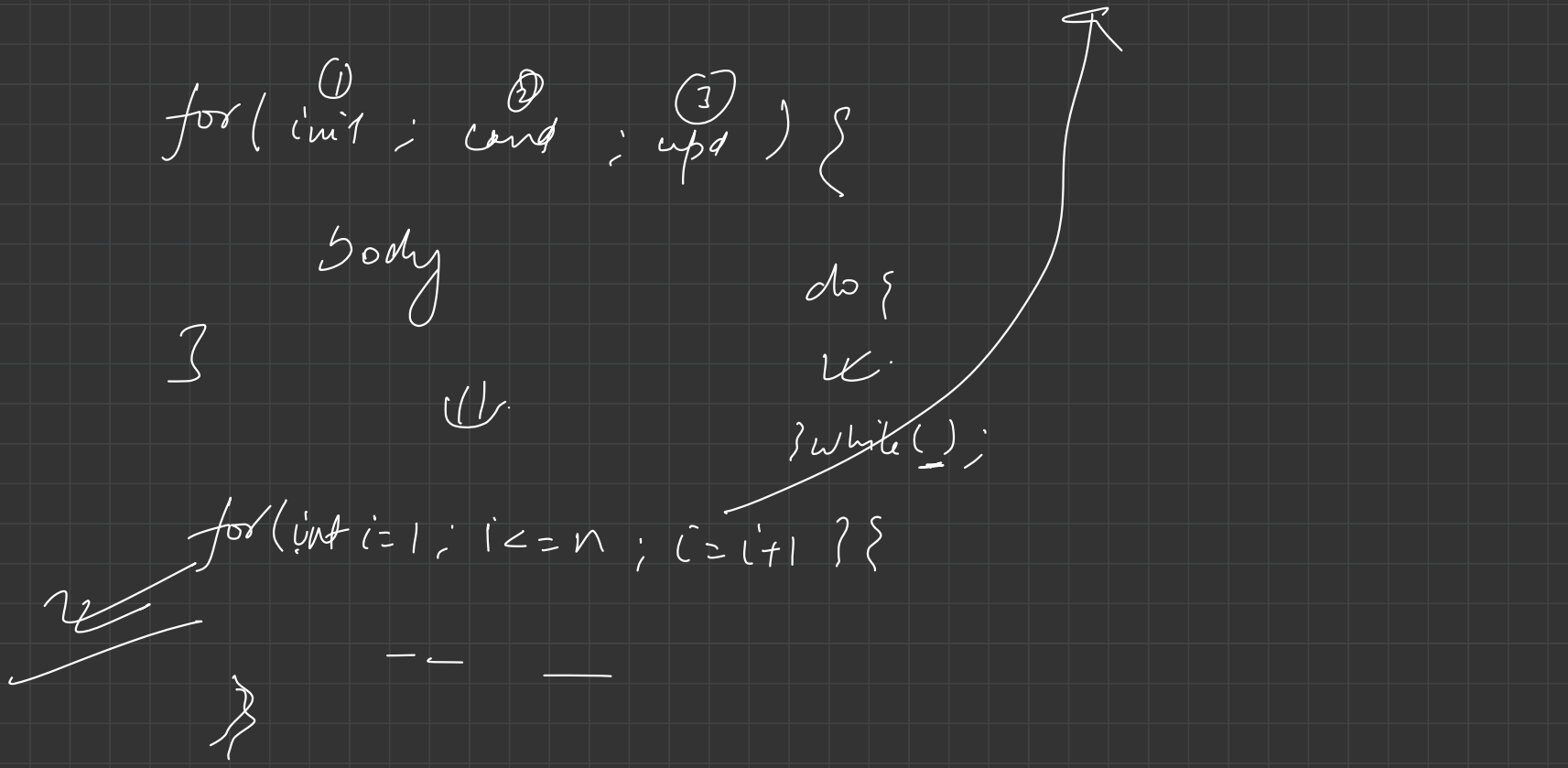
Nesting of loops:

Conditionals + Loops → Patterns.

i → error
i = 1 → ye

```
int i = 1;
while (i <= n) {
    // body
    i = i+1;
}
```

init
cond
body


for(init ; cond ; upd) {
①
②
③
body
}
do {
}while();
for(int i=1; i<=n ; i=i+1){
}