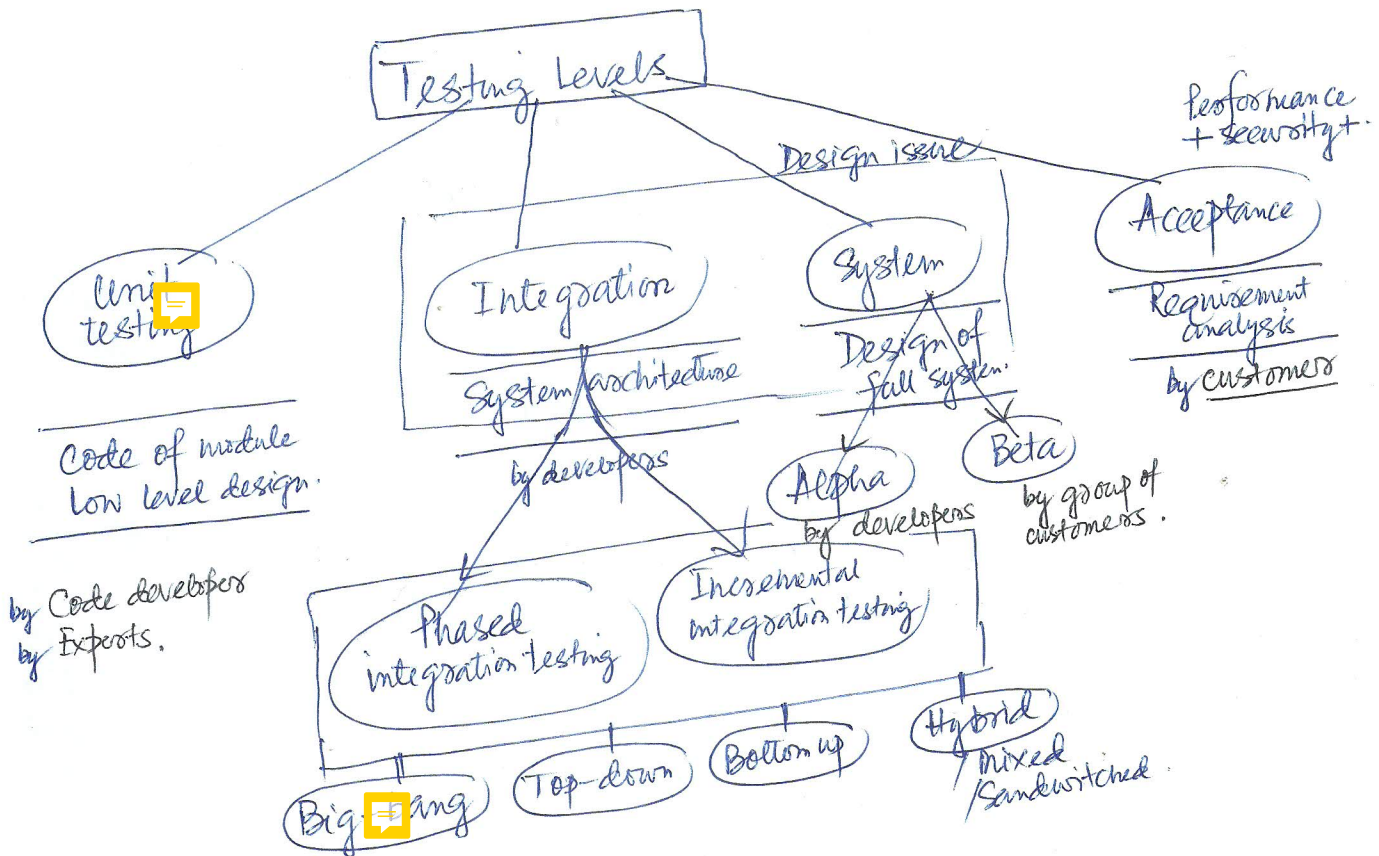
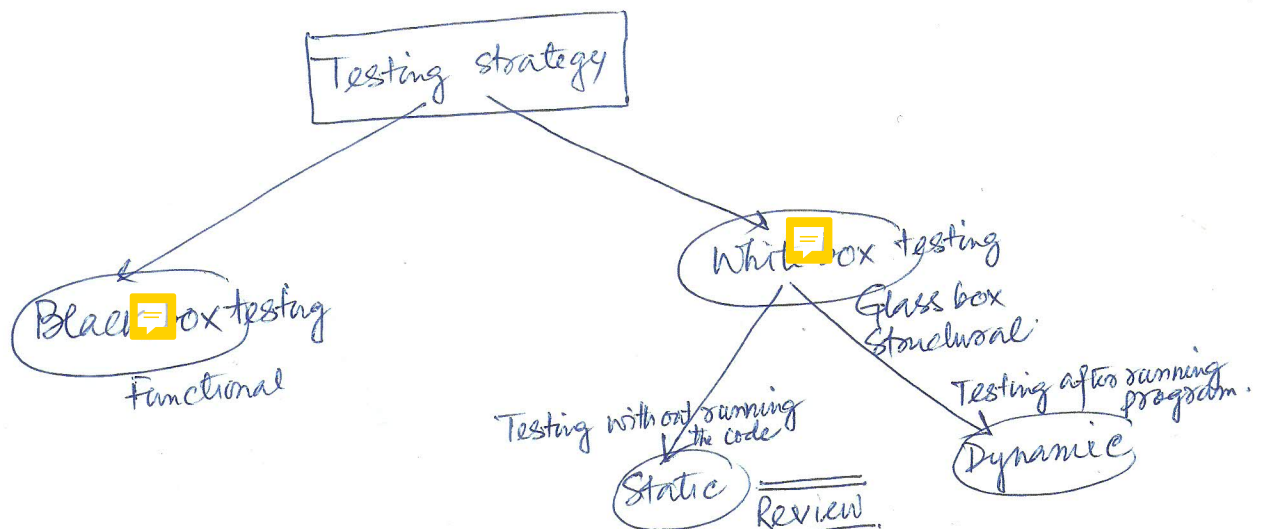


Testing

1.



2.

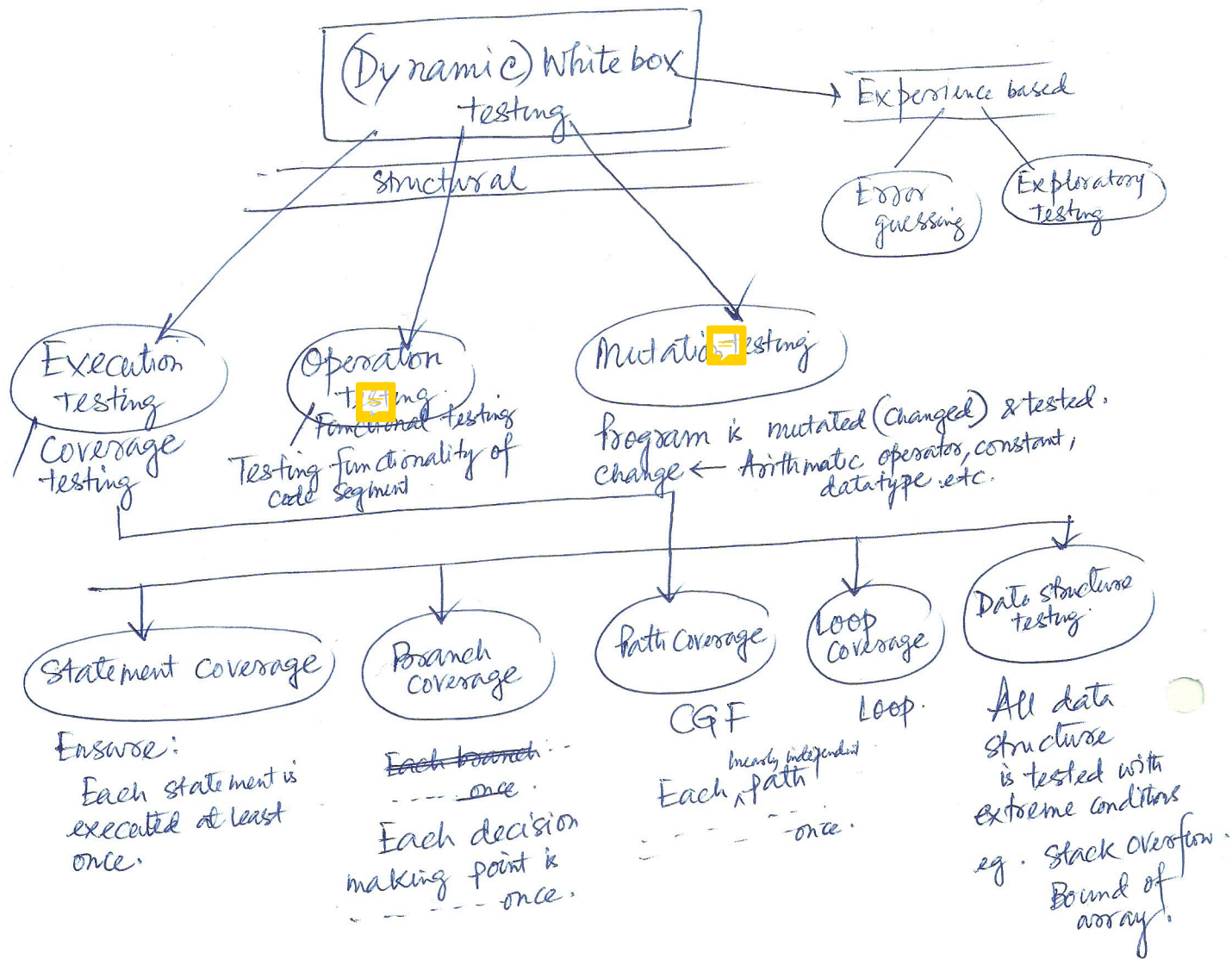


3.

Black box testing

- Boundary value analysis
- Equivalence class partitioning
- State transition testing
- Use case testing
- Decision tree testing
- Decision table testing
- User's story testing

- Desk checking by author
 - Code walk-through
 - Code inspection
- Analysis
- Data flow
 - Control flow
 - Data structure



CFG (Control flow graph)

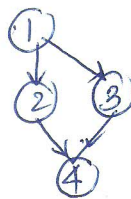
Sequence

1. $a = 10;$
2. $b = 2 \times a - 1;$



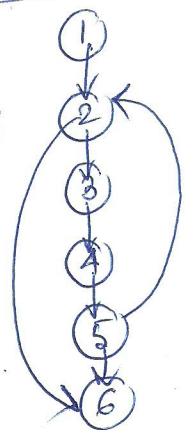
Selection

1. if ($a > b$)
2. $c = 3;$
3. else $c = 10;$
4. $c = c \times 2;$



Iteration

1. $i = 1;$
2. while ($i \leq 10$)
3. { printf("%d", i);
4. $i++;$
5. }
6. $i = i + 10;$



Data-flow testing

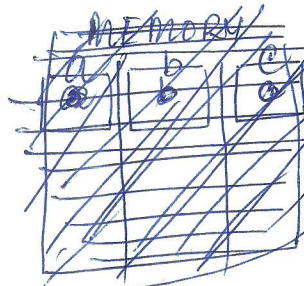
Test status of variable through paths available from CFG. It helps us to find

- variable declared but not used
- variable used but not declared
- variable declared multiple times
- variable declared after use
- Deallocating variable before use.

Desk checking a program/Algorithm/Pseudocode

Dry run

```
int a, b, c;  
printf("Enter two numbers");  
scanf("%d %d", &a, &b);  
c = a + b;  
printf("Result is %d", c);
```



SCREEN

Enter two numbers: 10 20
Result is 30.

MEMORY		
a	b	c
10	20	30

```
i ← 1  
while (i ≤ 3)  
{  
  x ← i + 10;  
  print x;  
  i ← i + 1;  
}
```

With trace table

Trace table

MEMORY	
i	x
1	11
2	12
3	13
4	

SCREEN

11 12 13

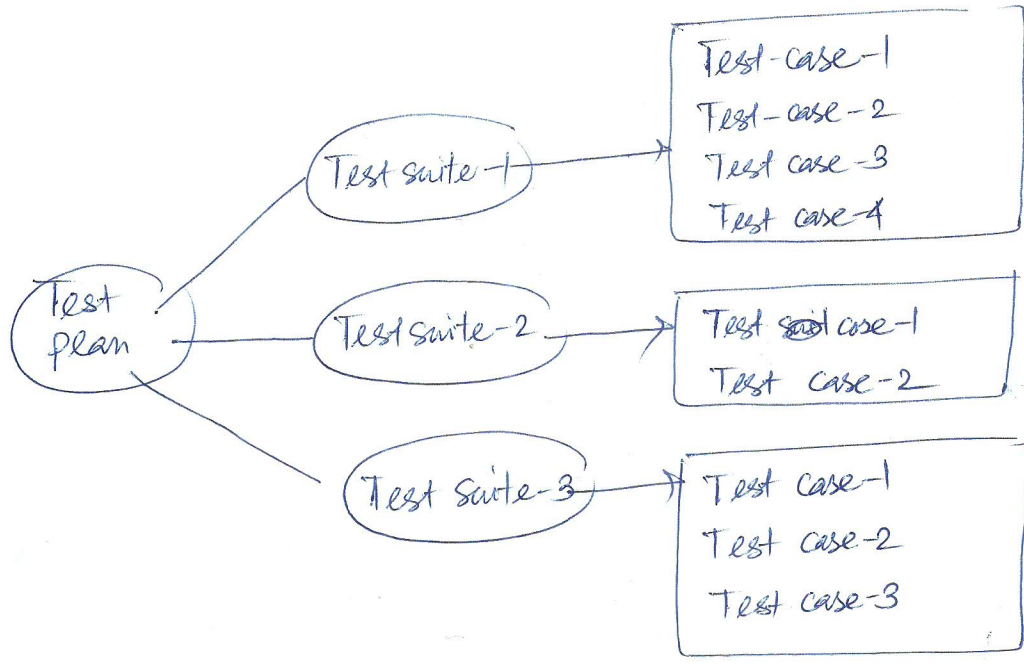
When Trace Table is not required

i	x
1	11
2	12
3	13
4	

screen

11 12 13

Test case, Test suite, Test plan



Test plan

Test suite-1				Test suite-2		Test suite-3		
Test case-1	Test case-2	Test case-3	Test case-4	Test case-1	Test case-2	Test case-1	Test case-2	Test case-3

Equivalence partitioning & Boundary value analysis

