The master method

It solves the recurrences of the form:

$$T(n) = a\,T\left(\frac{n}{b}\right) + f(n)$$

$a \geq 1$, $b > 1$, $f(n)$ is monotonically positive.

compute $n^{\log_b a}$

compare $n^{\log_b a}$ with $f(n)$

Three cases

case 1: $f(n) = O(n^{\log_b a - \epsilon})$ for $\epsilon > 0$.

solution: $T(n) = \theta(n^{\log_b a})$

Ex^m $T(n) = 4T(\frac{n}{2}) + cn$

$a = 4, \; b = 2$ $\quad n^{\log_b a} = n^{\log_2 4} = n^2$, $\quad f(n) = cn$

$cn = O(n^{\log_2 4 - \epsilon})$ for $\epsilon = 1$

solution: $T(n) = \theta(n^2)$

Case 2: $f(n) = \theta(n^{\log_b a})$

Solution: $T(n) = \theta(n^{\log_b a} \cdot \log n)$

||

$f(n) = \theta(n^{\log_b a} \cdot \log^k n)$

Solution:

$T(n) = \theta(n^{\log_b a} \log^{k+1} n)$

Ex^m $T(n) = 4T(\frac{n}{2}) + n^2 \log^5 n$

Solution: $T(n) = \theta(n^2 \log^6 n)$

Case 3: $f(n) = \Omega\left(n^{\log_b a + \epsilon}\right)$

Additional condition

$$a \cdot f\left(\frac{n}{b}\right) \leq c\, f(n) \quad \text{for some } c,\ 0 < c < 1$$

Solution: $T(n) = \theta\left(f(n)\right)$

Ex^{m} $T(n) = 4T\left(\frac{n}{2}\right) + n^3$

$f(n) = n^3, \quad n^{\log_b a} = n^2$

Case 3 cn. $f(n) = \Omega\left(n^{\log_b a + \epsilon}\right)$ for $\epsilon = 1$

$$af\left(\frac{n}{b}\right) \leq c\, f(n)$$

$$\Rightarrow 4 \cdot \left(\frac{n}{2}\right)^3 \leq c \cdot n^3$$

$$\Rightarrow \frac{n^3}{2} \leq c\, n^3$$

$$\Rightarrow c \geq \frac{1}{2}$$

for $\frac{1}{2} \leq c < 1$ this condition satisfies.

solution:

$$T(n) = \theta(n^3)$$

Array

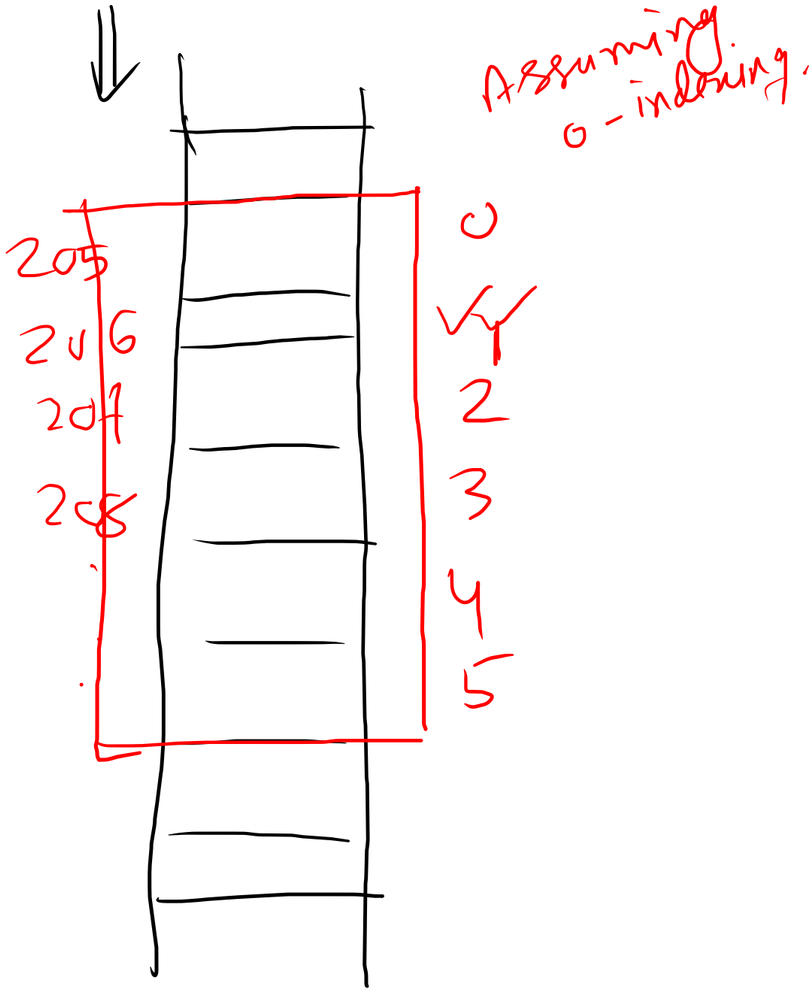
An array is a

finite -

ordered -

homogeneous -

collection of data element.



Terminologies:

elements: the values that we want to store

size: the number of elements that an array can store.

Base: The address of the memory location of the first element.

Type: the data type of the elements. $A[L..U]$

index: an integer value. vary from L to U.

index $A[i] = L + i - 1$ $size(A) = U - L + 1$

word: the size of an element.

capacity: number of element currents stored in the array.

Q: Why we need array?

Suppose we want to store marks of 100 students.

If we single variable.

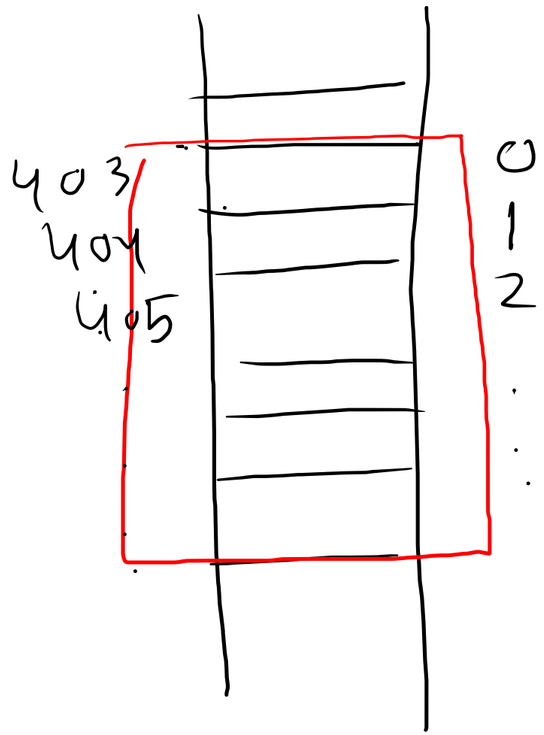
int mark1, mark2, . . . _mark100

define same name indexed.

int mark[100] = { . . . . . }.

one dimensional array

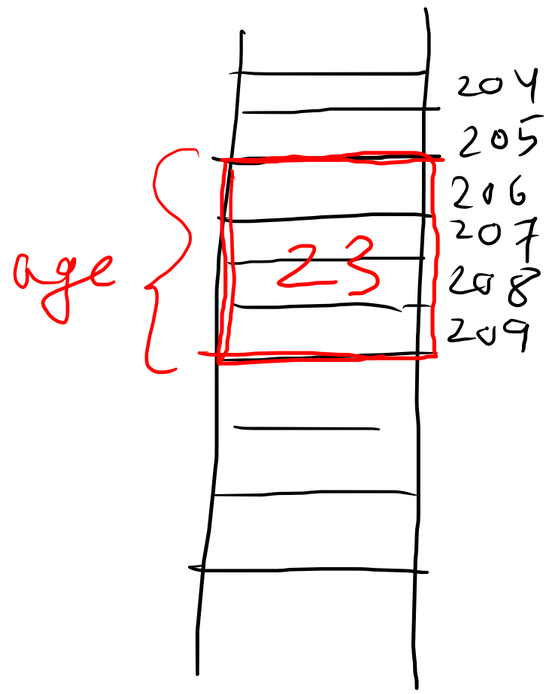
One index is required



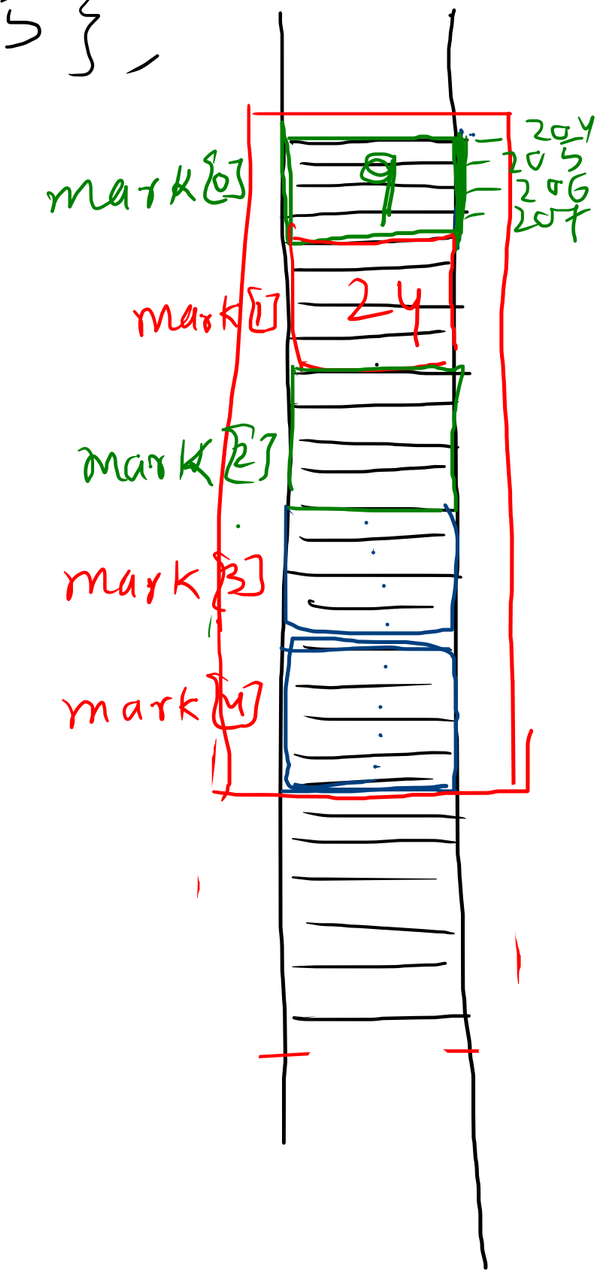
How array is stored in the memory

How a variable store in the memory?

```
int age = 23;
```



```
int mark[5] = { 9, 24, 3, 6, 35 };
```



Starting address of mark[2] is 212