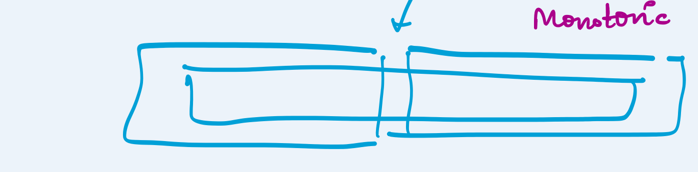
Tuesday, 9 March 2021 8:33 PM

Searching

- → Linear search
- → Binary Search [mid]

2

↓ Monotonic Functions



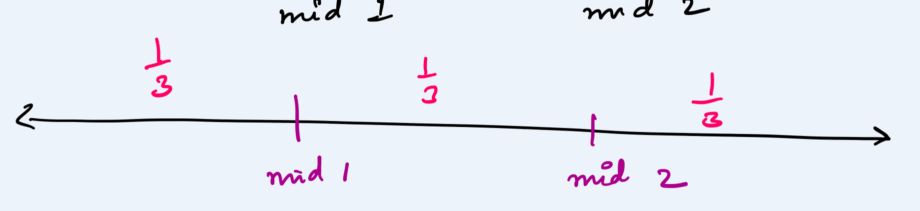
Ternary Search

3

→ Divide & conquer

→ division of array / function into 3 parts.

mid 1 mid 2



1. Search space
2. Find mid 1 and mid 2

mid 1 = l + (r−l)/3

mid 2 = r − (r−l)/3

1 2 3 4 5 6 7 8 9 10

l r

mid 1 mid 2

mid 1 = 0 + (9)/3 = 3

mid 2 = 9 − (9)/3 = 6

target → 6

target > arr[mid1] &&

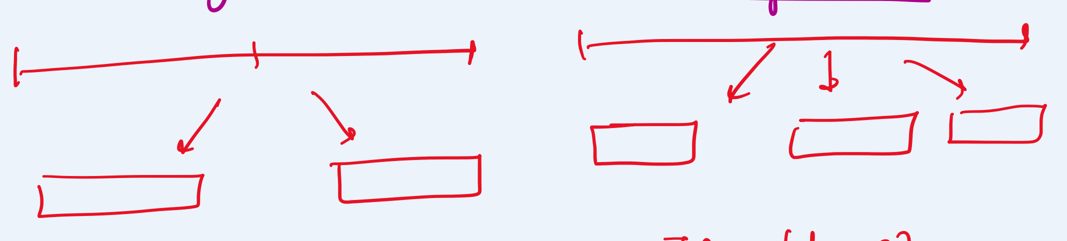
target < arr [mid2]

⇒ l = mid1 + 1;

r = mid2 − 1;

while (l<=r) {

Binary search Ternary search



TC → ($\log_2 n$) TC → ($\log_3 n$)

for each iteration,

Ternary search → 4 comparisions

Binary search → 2 comparisions

$T(n) = T(\frac{n}{3}) + 4c$

$T(n) = T(\frac{n}{2}) + 2c$

$T(n) = T(\frac{n}{3}) + 4c \Rightarrow T(n) = 4c\frac{\log(n)}{\log(3)} + c_1$

$T(n) = T(\frac{n}{2}) + 2c \Rightarrow T(n) = \frac{2c\log(n)}{\log(2)} + c_1$

BS → $T(n) = 2c\log_2 n + O(1)$

TS → $T(n) = 4c\log_3 n + O(1)$

Time taken for Ternary search is $2\log_3 2$ times the time taken by BS.

(TS) = $2\log_3 2$ (BS)

Since $2\log_3 2 > 1$

⇓

we get more comparisions for ternary search.

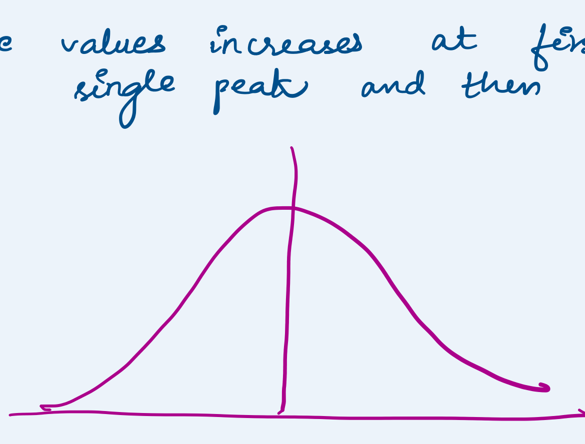
Linear function → TS → BS

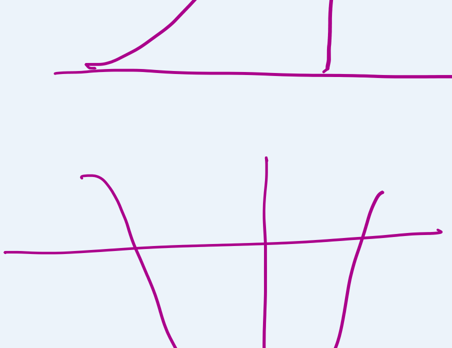
Unimodal distribution

one peak

→ A unimodal distribution is a distribution with one clear peak or most frequent value.

→ The values increases at first, reaches to a single peak and then it decreases.





1 2 3 3 3 4 4 5 5 7 6 5 5 5 4 2 2 2

→ bitonic sequence

↓

numbers are appearing in increasing order then suddenly start to decrease.

mid

1 2 3 3 3 4 4 5 5 7 6 5 5 5 4 2 2 2

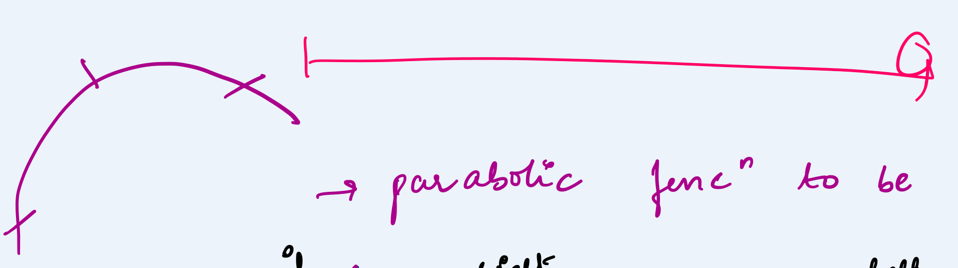
low high

mid = 5

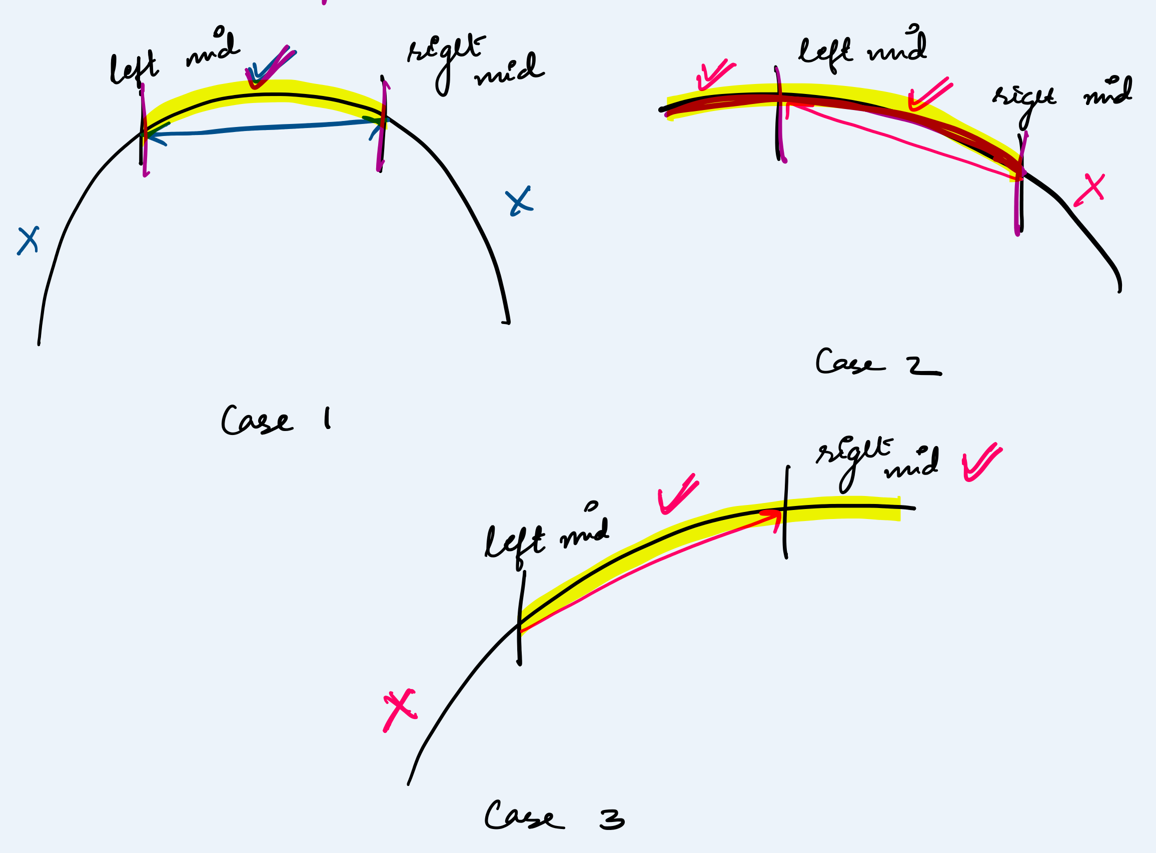
arr[low] = 1

arr[high] = 2

low ? 5 → ? low



→ parabolic func^n to be solved



Ternary search applications

→ unimodal function to determine the max or min value of a function

→ unimodal functions have single highest value

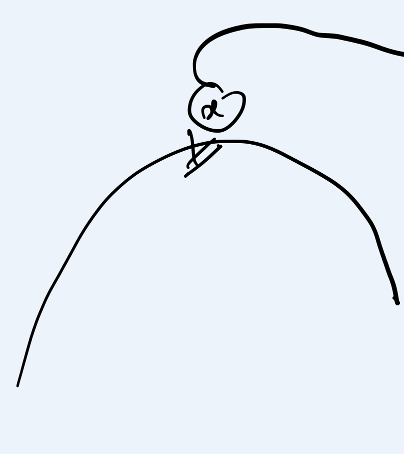
[a, b]

determine x

func(x) will be maximised

⇓

unimodal in nature



inc → [a, x]

dec → [x, b]

unimodal function

$f(x) = -ax^2 + bx + c$

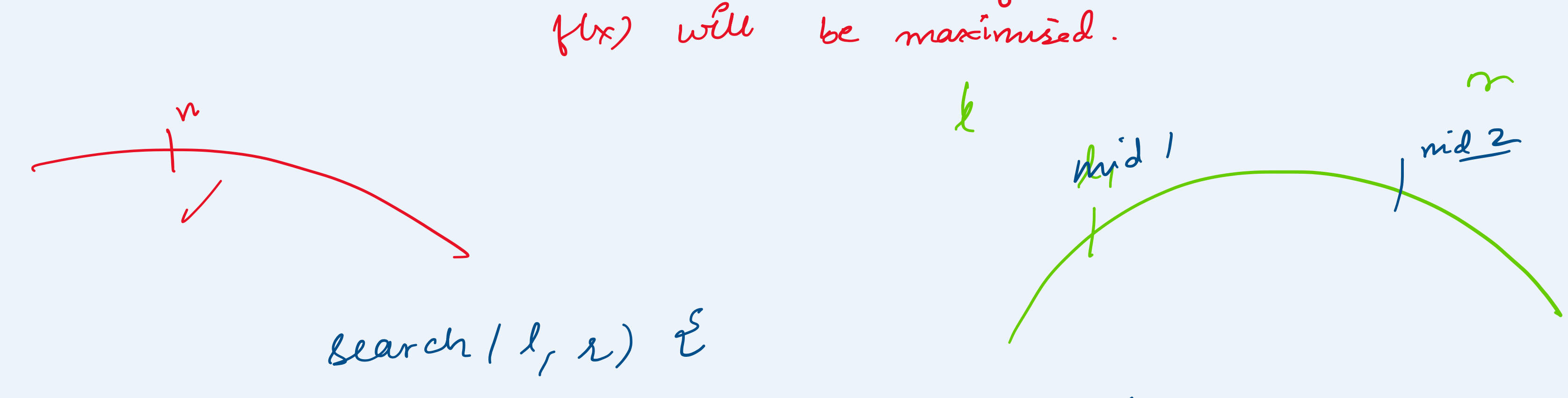
where

a = 1,

b = 2,

c = 3

determine the value of x where f(x) will be maximised.




```
search (l, r) {
    for( int i=0; i< 100; i++) {
        double mid1 = (l*2 + r)/3;
        double mid2 = (l+ 2*r) /3;
        if (f(mid1)> f(mid2) {
            r = mid2;
        else
            l= mid1;
    }
    double x = l;
    return f(x);
}

double f( double x) {
    return -x^2 + 2x - 3;
}
```