(1) Given an array of [4, -2, 5, 3, 10, -5, 2, 8, -3, 6, 7, -4, 1, 9, -1, 0, -6, -8, 11, -9] integers find the maximum and minimum product that can be obtained by multiplying two integers from the array.

Sol:-

array is [4, -2, 5, 3, 10, -5, 2, 8, -3, 6, 7, -4, 1, 9, -1, 0, -6, -8, 11, -9]

we need to consider the Largest and smallest Products that can be formed by selecting two numbers from the array.

1) sort the array.

sorted array

[-9, -8, -6, -5, -4, -3, -2, -1, 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11]

2) Identify possible candidates for maximum Product.

3) Identify possible candidates for minimum Product.

Calculating maximum Product:-

* The two largest positive numbers are 10 and 11

10 x 11 = 110

* The two smallest negative numbers are -9 and -8.

-9 x -8 = 72.

The maximum Product is 110.

Calculating minimum Product is.

The largest Positive and negitive number is 11 and -9

11 x -9 = -99

The smaller negitive numbers are

-9 x -8 = 72

-99 is smaller than 72 so

maximum Product = 110, and minimum Product = -99

(2) Demonstrate the Binary search method to search for the key=23 from the array={2,5,8,12,16,23,38,56,72,91}.

Sol:- given key=23 and array={2,5,8,12,16,23,38,56,72,91}

1. Initialize Pointers

low=0 and high=9

Calculate $mid=\left[\frac{low+high}{2}\right]=\left[\frac{0+9}{2}\right]=4$

Compare arr[mid] with key:

arr[4]=16

since 16<23 update low=mid+1=5

Calculate $mid=\left[\frac{low+high}{2}\right]=\left[\frac{5+9}{2}\right]=7$

Compare arr[mid] with key:

arr[7]=56

since 56>23 update high=mid-1=6

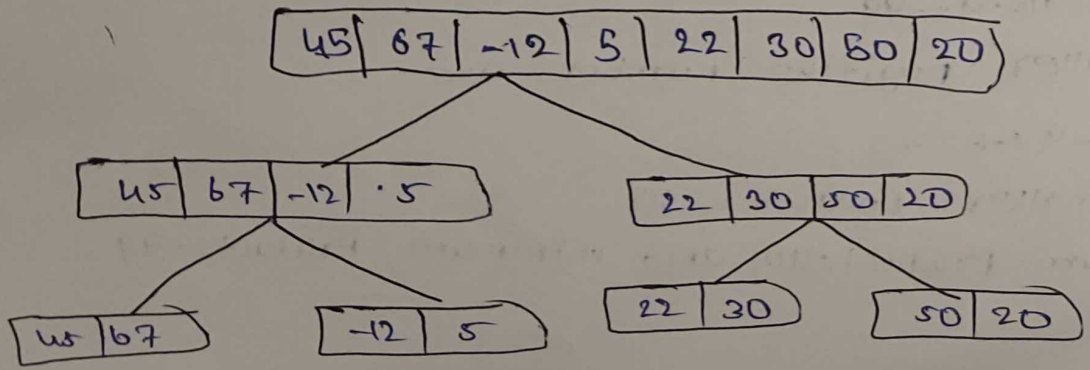
$mid=\left[\frac{5+6}{2}\right]=5$

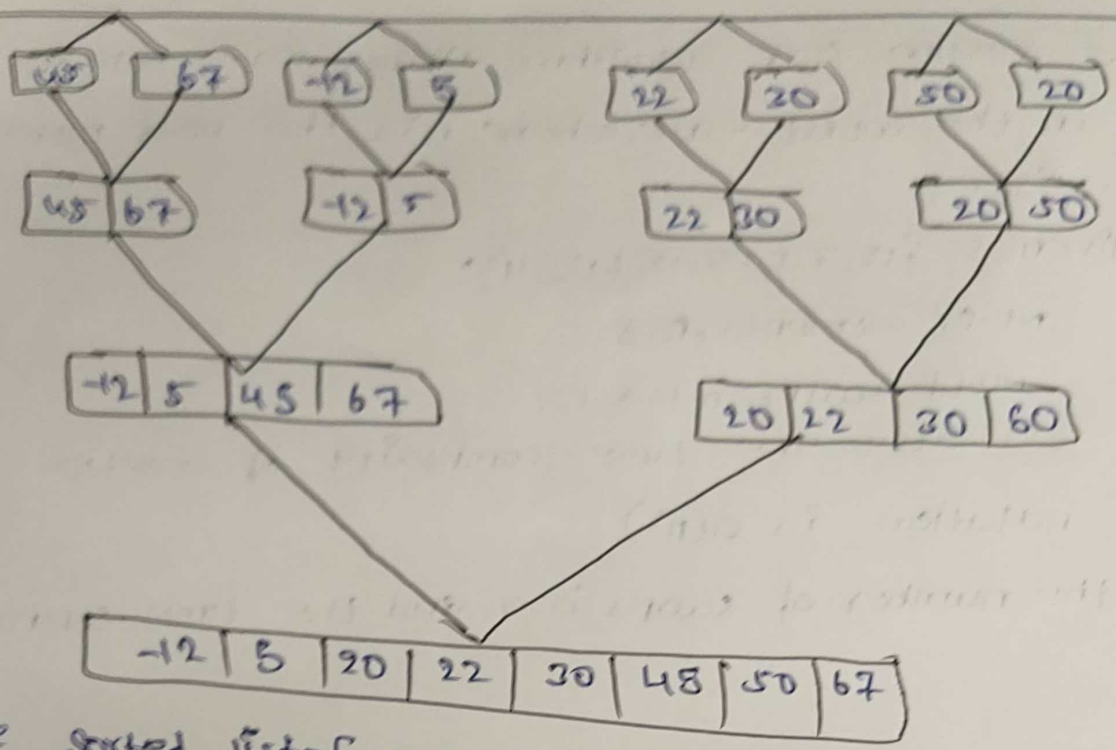
arr[mid]=arr[5]=23

23=23 the key is found at index 5.

∴ The key=23 is found at index 5.

(3) Apply merge sort and other List of 8 elements, Pata d=[45,67,-12,5,22,30,50,20). set up recurrence relation for the number of key comparisions made by mergesort.





∴ The sorted list = [-12, 5, 20, 23, 30, 45, 50, 67]

Recurrence Relation for comparisons:

$$T(n) = 2T(n/2) + O(n)$$

if $n=1$, $T(1) = 0$. Base case

⇒ At each level of recursion we make at most $n-1$ comparisons to merge two halves of size $n/2$ so it becomes.

$$T(n) = 2T(n/2) + (n-1)$$

Solving recurrence relation we get

$$T(n) = n\log_2(n) - n + 1$$

$$T(n) = O(n \log n)$$

∴ The recurrence relation is $T(n) = 2T(n/2) + O(n)$

or more precisely.

$$T(n) = n\log_2(n) - n + 1$$

(4) Find the no. of times to perform solving swapping for selection sort also estimate the time complexity for the order of notation set s (12, 7, 5, -2, 18, 6, 13, 4)

The selection sort algorithm always makes exactly n-1 swaps in the worst case, where n is the no of elements in the list.

given S= {12, 7, 5, -2, 18, 6, 13, 4}:

No of elements, n=8

No of swaps = n-1 = 8-1 = 7

Time complexity - The time complexity of selection sort in Big O notation is $O(n^2)$

So, the number of swaps is 7, and the time complexity is $O(n^2)$

(5) Find the index of the target value 10 using binary search from the following list of elements [2, 4, 6, 8, 10, 2, 14, 16, 18, 20].

given list = {2, 4, 6, 8, 10, 2, 14, 16, 18, 20} and value = 10

Low = 0 and high = 9

$mid = \frac{low + high}{2} = \frac{0+9}{2} = 4$

List[4] = mid = 10 mid == value

since 10 == 10 the target is found at index

∴ The target value = 10 is found at index.