1. **In the Binary Search algorithm, it is suggested to calculate the mid as beg + (end - beg) / 2 instead of (beg + end) / 2. Why is it so?**

Ans :-

In case of “ beg+(end-beg)/2 “ it is guaranteed that the expected result is no larger than the “ end “ and also in this case there is no fear of overflow.

This case can also be used for affine types like pointers and other random-access iterators, which can be subtracted to give a distance, but not added together.

1. **Write the algorithm/function for Ternary Search.**

Ans:-

Algorithm is -------

ternarySearch(array, start, end, key)

Begin

if start <= end then

midFirst := start + (end - start) /3

midSecond := midFirst + (end - start) / 3

if array[midFirst] = key then

return midFirst

if array[midSecond] = key then

return midSecond

if key < array[midFirst] then

call ternarySearch(array, start, midFirst-1, key)

if key > array[midSecond] then

call ternarySearch(array, midFirst+1, end, key)

else

call ternarySearch(array, midFirst+1, midSecond-1, key)

else

return invalid location

End