4)

```
MajorityElement(array, start, end)
{
if start == end:
        return array[start]

mel = MajorityElement(array, start, (start + end - 1)/2)
mer = MajorityElement (array, (start + end - 1)/2 + 1, end)

if mel != Null :
        melcount=0
        for i in (start,end):
                if array[i] == mel :
                        melcount = melcount + 1
else if mer != Null :
        mercount=0
        for i in (start,end):
                if array[i] == mer :
                        mercount = mercount + 1
if melcount > (end – start+1)/2:
        return mel
else if mercount > (end – start+1)/2:
        return mer
else:
        return null
}
```

در این الگوریتم ازین موضوع استفاده میکنیم که اگر یک عضو در آرایه ای پر تکرار باشد، باید حتما در نیمه ی چپ یا نیمه راست آن هم عضو پر تکرار داشته باشیم. پس همواره عضو پر تکرار نیمه ی چپی و نیمه ی راستی آرایه را پیدا میکنیم و چک میکنیم که آیا در آرایه اصلی هم عضو پر تکرار هست یا خیر. چون به صورت بازگشتی و با نصف کردن انجام می شود پس تقسیم و غلبه است.

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

حال طبق روابط میدانیم که از رابطه ی بالا به $O(nlogn)$ میرسیم که مطلوب سوال است.