lab2_1.cpp > main()

```cpp
#include<bits/stdc++.h>
using namespace std;
void new_sort(vector<int>& a,int l,int r)
{
    //|A|==1
    if(r==l) return ;
    int m=(l+r)/2;
    new_sort(a,l,m);
    new_sort(a,m+1,r);
    int n=(r-l+1);
    for(int i=0;i<(n/2);i++)
    {
        if(a[i+l]>a[i+l+n/2])
        {
            swap(a[i+l],a[i+l+n/2]);
        }
    }
    new_sort(a,l,m);
    new_sort(a,m+1,r);
}
int main()
{
    vector<int> a={6,3,8,1,7,4,2,5};
    int n=a.size();
    //check if n is a power of 2
    if(n&(n-1)!=0)
    {
        cout<<"array size is not power of 2"<<endl;
        return 0;
    }
    cout<<"Given array"<<endl;
    for(auto &x : a) cout << x << " ";
    cout<<endl;
    new_sort(a,0,n-1);
    cout<<"After NEW SORT"<<endl;
    for(auto &x : a) cout << x << " ";
    cout<<endl;
}
```

```
c:\Users\umesh\OneDrive\Desktop\cso-221 lab\output>.\"lab2_1.exe"
Given array
6 3 8 1 7 4 2 5
After NEW SORT
1 5 3 7 2 6 4 8
```

1a) $T(n) = 4T\left(\frac{n}{2}\right) + o(n)$

$T(n) = 4T\left(\frac{n}{2}\right) + n$

$a = 4 \geq 1 \quad b = 2 > 1 \quad f(n) = n$

$\log_b a = \log_2 4 = 2.$

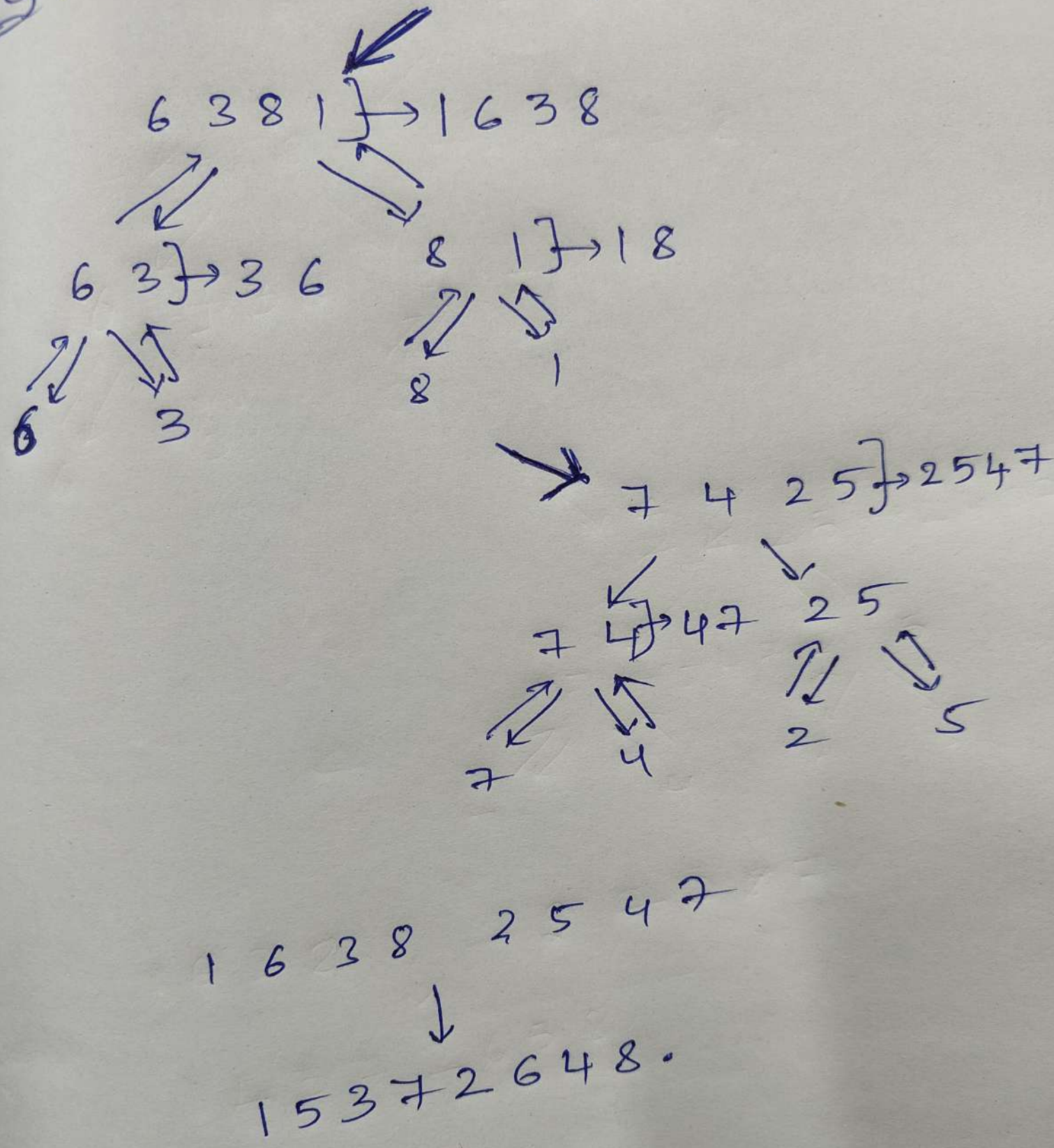
$n^{\log_b a} = n^2 > n = f(n)$

So, from masters theorm case-I;

$T(n) = O(n^2).$

1b) No, the algorithm docsnot sort correctly. Because, it is not checking all pairs of elements.

Eg:- 6 3 8 1 7 4 2 5



1 6 3 8 2 5 4 7

↓

1 5 3 7 2 6 4 8.

(a) $T(n) = 9T\left(\frac{n}{2}\right) + n^2$

$a = 9 \geq 1 \quad b = 2 > 1 \quad f(n) = n^2 > 0$

$\log_b a = \log_2 9$

$k = 3$

$n^2 = f(n) = O(n^{\log_2 9 - \epsilon})$ for some $\epsilon > 0$;

So; $T(n) = O(n^{\log_2 9})$.

(b) $T(n) = 2T(n-1) + 1$.

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

$2T(n-1) = 4T(n-2) + 2$

$4T(n-2) = 8T(n-3) + 4$

$\vdots$

$2^{(k+1)}$

$2^k T(n-k) = 2^{(k+1)} T(n-(k+1)) + 2^k$

$T(n) = 2^{k+1} T(n-(k+1)) + [1+2+4+8+\ldots+2^k]$

when; $k+1 = n \Rightarrow T(0) = 1$

$T(n) \approx 2^n + (1)[2^n]$

$T(n) = O(2^n)$

(c) $T(n) = 64\,T\left(\frac{n}{2}\right) + 2^n$

$a = 64,\ b = 2,\ f(n) = 2^n$

masters theorem

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

$$64\, 2^{\frac{n}{2}} \leq c\, 2^n$$

$$c \geq \frac{64}{2^{n/2}}$$ for large n;

$\exists c < 1$

hence; $T(n) = O(2^n)$.