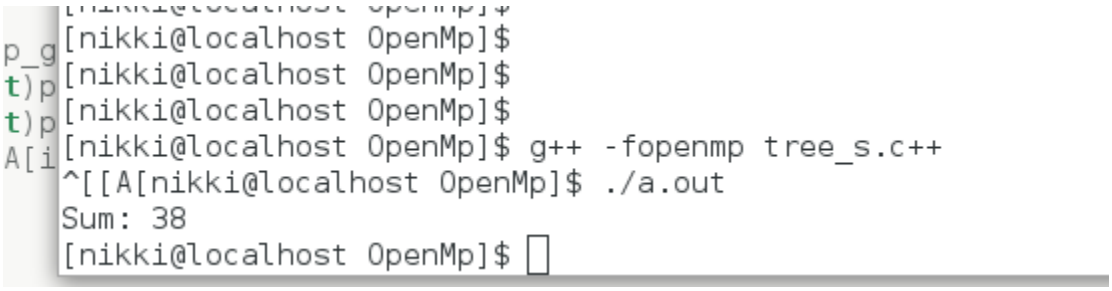
Parallel Tree Summation

```
#include <iostream>
#include "omp.h"
#include <time.h>
#include <math.h>
#define N 8
int main()
{
    int A[N]={7,0,2,9,5,1,8,6};
    int x=(int)log2(N),temp,last;
    omp_set_num_threads((N/2)+1);
    last=A[N-1];
    for(int d=0; d<x; d++)
    {
        #pragma omp parallel
        {
            if(omp_get_thread_num()!=0){
                int i=(omp_get_thread_num()-1)*pow(2,d+1);
                int a=(int)pow(2,d+1);
                int b=(int)pow(2,d);
                A[i+a-1]=A[i+b-1]+A[i+a-1];
            }
        }
    }
    A[N-1]=0;
    for(int d=x-1; d>=0; d--)
    {
        #pragma omp parallel
        {
            if(omp_get_thread_num()!=0){
                int i=(omp_get_thread_num()-1)*pow(2,d+1);
                int a=(int)pow(2,d+1);
                int b=(int)pow(2,d);
                temp=A[i+b-1];
                A[i+b-1]=A[i+a-1];
                A[i+a-1]=temp+A[i+b-1];
            }
        }

    }
    std::cout<<"Sum: "<<A[N-1]+last<<"\n";
    return 0;
}
```


[nikki@localhost OpenMp]$
[nikki@localhost OpenMp]$
[nikki@localhost OpenMp]$
[nikki@localhost OpenMp]$ g++ -fopenmp tree_s.c++
^[[A[nikki@localhost OpenMp]$ ./a.out
Sum: 38
[nikki@localhost OpenMp]$