In Lab:

1) Marc loves cupcakes, but he also likes to stay fit. Each cupcake has a calorie count, and Marc can walk a distance to expend those calories.

If Marc has eaten j cupcakes so far, after eating a cupcake with c calories he must walk at least 2^j*c miles to maintain his weight. For example, if he eats 3 cupcakes with calorie counts in the following order: [5,10,7], the miles he will need to walk are $(2^0*5)+(2^1*10)+(2^2*7)=5+20+28=53$. This is not the minimum, though, so we need to test others orders of consumption. In this case, our minimum miles is calculated as $(2^0*10)+(2^1*7)+(2^2*5)=10+14+20=44$,

Given the individual calorie counts for each of the cupcakes, determine the minimum number of miles Marc must walk to maintain his weight. Note that he can eat the cupcakes in any order.

Sample Input:
```
3
1 3 2
```

Sample Output:
```
1 1
```

Code:

```c
#include <math.h>

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#include <assert.h>

#include <limits.h> #include <stdbool.h>

int cmpfunc (const void * a, const void * b)

{

    return ( *(int*)a - *(int*)b );
```
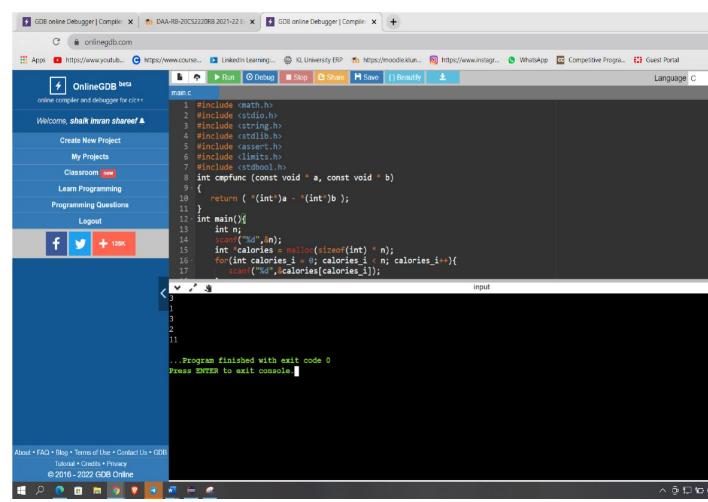
```c
}
int main(){
    int n;
    scanf("%d",&n);    int *calories =
malloc(sizeof(int) * n);    for(int calories_i = 0;
calories_i < n; calories_i++){
scanf("%d",&calories[calories_i]);
    }
    qsort(calories,n,sizeof(int),cmpfunc);
long long int a=0; int i;    for(i=n-1;i>=0;i--
){    a=a+calories[i]*pow(2,n-1-i);
    }
    printf("%lld",a);    //
your code goes here
return 0;
}
```