Idea

With a given number of grades we need to calculate the average of these. If the number of grades is a power of 2, in each recursive stage we divide the left side average summed with the right side average by two (shown in the output file). If not, we just sum up all grades and divide it by the number of subjects, obtaining the final grade.

Methodology

We used the Divide and Conquer algorithm. The problem was solved by recursively breaking down the problem into two or more sub-problems, until these were simple enough to be solved directly (until we could achieve the base case: l == r). Then we just calculated the average in each recursive stage, in both types of iterations.

CodeProblem2_powerof2.java

In this case we are working with "n" subjects where "n" is a power of 2. Because of that we will always be working with two equal division, so we just need to divide by 2 the sum of the grades in each recursive stage to get the corresponding average.

CodeProblem2_aux.java

In this case we are working with "n" subjects where "n" may be not a power of 2. However, here we are only calculating the final average, instead of the average in each of the recursive stages.

CodeProblem2.java

In this case we are working with "n" subjects where "n" may be not a power of 2. In adittion, all the averages in each of the corresponding stages are calculated.

This case has been done adding these criteria:

When comparing averages, if the difference between one and another in terms of position (l and r) is greater than 1 and it is even, then, if we are not in the last recursive stage :

$$\text{Average} = \text{averageL} + \text{averageR} + \text{averageL} * \sqrt{(r - l)} / \sqrt{(r - l)} + 2$$

In the last stage, if the number of subjects is odd, we will always have 1 more grade in the left side of the division, that's why:

```
average = (avgLeft*(subjects/2) + avgRight*(subjects/2) + avgLeft)/subjects;
```

Efficiency

We used the Divide and Conquer algorithm, so the efficiency is O(log(n)).

Simple iteration (power of 2):

```
public static double averageDC(int[] grades, int l, int r, FileWriter fw) throws IOException
{
    //Empty Case
    if (l > r)
    {
        return 0;
    }
    // Base Case: if there's just a single element it is the average of the array itself
    if (l==r)
    {

        return grades[l];
    }


    int mid = (l+r)/2;
    double avgLeft = 0;
    double avgRight = 0;

    avgLeft = averageDC(grades, l, mid, fw);
    avgRight = averageDC(grades, mid+1, r, fw);

    double average = (avgLeft + avgRight)/2;

    return average;
}
```

T(n) = 1 + 1 + 1 + t(n/2) + t(n/2) + 1

n = 2^k x^k = T(n)

x^k = 4 + 2x^(k-1)

x^(k-1) * (x-2) = 0 x(H) = A

x^(k-1) * (x-2) = 4 x(P) = Bk

X = A + Bk = A + log(n)B

= 0(log(n))

In every case, the efficiency will be 0(log(n)), because we have no added more recursive calls neither loops.

Input file

The first line indicates the number of subjects. The next lines indicate the grade achieved in a specific subject.

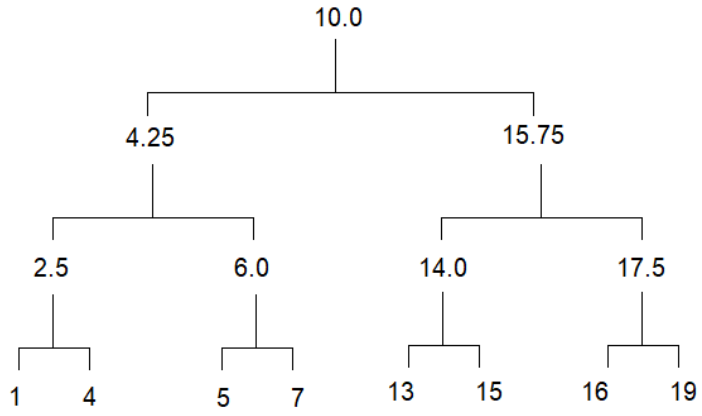
Output file

Example of the simple iteration:

output_d&c: Bloc de notas

Archivo Edición Formato Ver Ayuda

```
Average Left: 1.0
Average Right: 4.0
Calculated average: 2.5

Average Left: 5.0
Average Right: 7.0
Calculated average: 6.0

Average Left: 2.5
Average Right: 6.0
Calculated average: 4.25

Average Left: 13.0
Average Right: 15.0
Calculated average: 14.0

Average Left: 16.0
Average Right: 19.0
Calculated average: 17.5

Average Left: 14.0
Average Right: 17.5
Calculated average: 15.75

Average Left: 4.25
Average Right: 15.75
Calculated average: 10.0

Final average value: 10.0
```



Example of the final iteration:

output_d&c: Bloc de notas

Archivo Edición Formato Ver Ayuda

```
Average Left: 1.0
Average Right: 4.0
Calculated average: 2.5

Average Left: 2.5
Average Right: 5.0
Calculated average: 3.2322330470336316

Average Left: 5.0
Average Right: 6.0
Calculated average: 5.5

Average Left: 5.5
Average Right: 9.0
Calculated average: 6.525126265847083

Average Left: 3.2322330470336316
Average Right: 6.525126265847083
Calculated average: 4.878679656440357

Final average value: 5
```

example_d&c: Bloc de notas

Archivo Edición Formato Ver Ayuda

```
6
1
4
5
5
6
9
```