Joseph Karl V. Boncavil

The main inspiration for this project is because this was a topic in our math in the modern world and I wanted to study while making my program.

APPORTIONMENT is a method of dividing a

whole into various parts.

It began when the Constitution of the U.S.

(1790) first attempted to decide on “should be”

number of representatives based on the number

of voters per state.

A screenshot of a cell phone

Description automatically generated



This is the code for the divisor for the standard quota, total number of population/ total resources.

A screen shot of a computer code

Description automatically generated

Standard quota code, the “int s” is what column it will set the standard quota.

A computer screen shot of a code

Description automatically generated

This is for calculation lower quota, “int g” is what column to get the values, and “int s” is what column to set the lower quota

A screen shot of a computer program

Description automatically generated

The code for calculating the extra seats/resources to be allocated, the larger decimal points will get the extra seats/ resources.

A computer screen shot of a program code

Description automatically generated

This is the code for hamilton’s apportionment. It adds the extra seats/resources if there are any to the lower quota “LQ”.

A screenshot of a cell phone

Description automatically generated

A screen shot of a computer program

Description automatically generated

The code for jefferson’s method, just adjusting the divisor meaning the divisor gets smaller so the the standard quota will be modified then setting it to the appropriate column same and the appropriate lower quota will be set to the appropriate column which is the final apportionment.



This is how I adjust the divisor, it’s in a loop until I get the desired divisor.

A screenshot of a cell phone

Description automatically generated

A screen shot of a computer screen

Description automatically generated

Basically the same with Jefferson but this uses the upper quota, meaning it adjusts the divisor based on the upper quota.

A computer screen shot of a code

Description automatically generated

This is the upper quota code.

A screenshot of a cell phone

Description automatically generated

A screen shot of a computer program

Description automatically generated

This is the code for webster’s method it adjusts the divisor based on the need if we need a larger number then the divisor becomes smaller and vice versa.

A computer screen shot of a code

Description automatically generated

This is the rounding off based on the nearest integer. Basically the traditional rounding method.

A screenshot of a computer

Description automatically generated

A math problem with red text

Description automatically generated with medium confidence

A screen shot of a computer code

Description automatically generated

A screen shot of a computer program

Description automatically generated

These are the codes for the huntingtonhill apportionment. Getting the geometric mean ”GM”. Then setting the apportionment if Standard quota “SQ” is < than GM, then use the lower quota else use the upper quota.

In the GM code, int g1, int g2 is where you get the values of the lower quota and upper quota, then int s is where I set the GM.

In the huntington hill qpportionment. Int lq and int uq is where I get the values for lower and upper quota. Int g1 is where I get the values for standard quota and int g2 is where I get the values for geometric mean.. int s is where I set the apportionment.

Sample inputs and outputs:

So imagine Sir rosal gave a class budget 200 USB sticks. So how does groups james, jyx and lian split this?

So in the top box, the one labeled total resources to be allocated, we input 200.

Next, we add rows by clicking the add row button.

Input here is under the 1st column, james, jyx and lian.

Then under the 2nd column, how many students are under james, jyx, and lian. So its 9, 10 and 11 respectively. so those are the inputs in the 2nd column.

After that we press the calculate button and it now calculated how many USB sticks each group will receive based on hamilton’s apportionment.

Click reset button to reset all fields.

Click the edit button to edit values after calculating.

Same process for the other methods.

Sample input:

A screenshot of a computer

Description automatically generated

Sample output:

A screenshot of a computer

Description automatically generated