**Background**

My project was a sound absorption and reverb time calculator. This program takes input from the user as room dimensions of the rectangular room and the dimensions of each wall. The user also picks the material of each surface from a list. The code outputs the total sound absorption, the estimated reverb time of the room, and displays a pie chart of each surface’s effective absorptive area.

**Explanation of the Code**

**User Input**

The program prompts the user for the length, width, and height of the rectangular room, and then uses those values to calculate the volume of the room (stored in ‘V’), and the surface area of the floor and ceiling; as seen below.

LF=input("Please enter the length of the floor in feet.\n");

WF=input("Please enter the width of the floor in feet.\n");

H=input("Please enter the height of the room in feet.\n");

V=LF\*WF\*H;

AF=LF\*WF;

The program then prompts the user to pick a material out of the given list, using a switch statement; it then calculates the effective absorption area by multiplying the absorption coefficient (a metric of how absorptive a surface is, ranging from 0 to 1) by the surface area. Additionally, each surface’s effective absorption area is put into a matrix labeled ‘TotalEA’,

This switch statement occurs repeatedly, using different variable names, for all 6 surfaces of the room.

materials=input("Please enter the material of the floor. Please type one of the following:\n" + ...

" Carpet\n Concrete\n Wood\n Tile\n Brick\n\n ", "s");

switch materials

case "Carpet"

EAF=0.15\*AF;

case "Concrete"

EAF=0.06\*AF;

case "Wood"

EAF=0.06\*AF;

case "Tile"

EAF=0.01\*AF;

case "Brick"

EAF=0.04\*AF;

otherwise

disp("Invalid input. Please try again.");

end

TotalEA=EAF;

**Calculations and Output**

The program then puts each surface area into a matrix, sums both the total surface area and the total effective absorptive area (stored in ‘S’ and ‘A’ respectively), and calculates the total absorption of the room by dividing the effective absorptive area by the total surface area.

SurfaceAreas=[AF,AC,A1,A2,A3,A4];

S=sum(SurfaceAreas);

A=sum(TotalEA);

TotalAbsorbtion=(A/S)

Next, the program runs an if-else statement, due to there being two equations for calculating the reverberation time of the room. If the total absorption is above 0.2, the program runs the first equation, if the total absorption is below 0.2 (the ‘else’ in this case), the program will run the second equation.

if TotalAbsorbtion >= 0.2

RT60=-0.049\*(V/(S\*log(1-TotalAbsorbtion)))

else

RT60=0.049\*(V/A)

end

**Display**

Displaying the pie chart in a neat fashion is a bit of a task. The function below takes in the matrix of each surface’s absorptive area, and then returns the maximum value of the matrix (‘c’) and the index of that maximum value (‘ind’).

The matrix called ‘explode’ contains six zeros, corresponding to the six surfaces of the room. The program then takes ‘ind’ and sets the value of ‘explode’ to one, at that same index. This will make the maximum absorptive area on the pie chart pop out from the rest.

[c,ind]=max(TotalEA);

explode=zeros(1,6);

explode(ind)=1;

Then, there is a matrix of strings (‘labels’) which will be the labels for the legend. The ‘legend’ function takes in labels, and other parameters about location and orientation, and then displays

a legend for the pie chart when the program runs. The ‘title’ function sets the title of the graph, and the ‘pie’ function finally displays the pie chart of the desired matrix, and explodes the maximum value of the matrix.

labels={'Floor','Ceiling','Wall 1','Wall 2','Wall 3','Wall 4'};

legend(labels,'Location','northeastoutside','Orientation','vertical');

title("Effective Absorbtion Area of Each Surface");

pie(TotalEA,explode)

**Bugs and Troubleshooting**

The only major flaw in the program is the lack of a ‘fail safe’; If there is user error when selecting a material from a list (the user input not matching the given strings exactly), the code then throws an error and stops running. This is something I tried to overcome with ‘While’ loops, but then ran into scope issues later down the road. I’m sure there is a solution to this that I’m not seeing, and is a simple fix.

Also, having a check to make sure each wall dimension (Wall 1, 2, 3, etc) matches a value given in the initial prompt for the room dimensions (length, width, height) would be very beneficial, but is something that I am unsure how to implement.