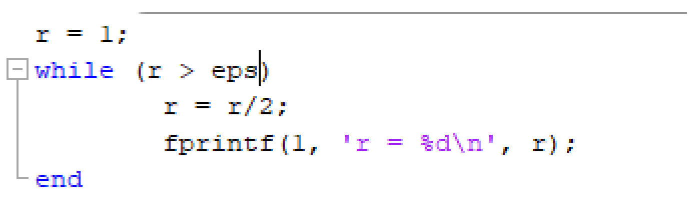
Section 3

2.1



```
r = 1;
while (r > eps)
        r = r/2;
        fprintf(1, 'r = %d\n', r);
end
```

A. Yes it is a more appropriate test for terminating the loop since it will stop after a smaller amount of decimal points where it becomes less important. In this case there were 52 numbers while using eps and 1075 while using 0.

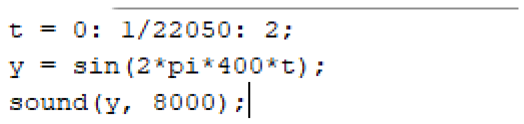
B.

  a. The value of eps is 2.220446e-16

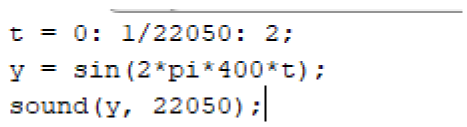
  b. The value of realmin is 2.225074e-308

  c. The value of realmax is 1.797693e+308

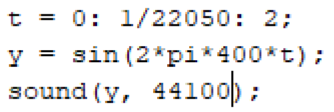
2.2



```
t = 0: 1/22050: 2;
y = sin(2*pi*400*t);
sound(y, 8000);
```



```
t = 0: 1/22050: 2;
y = sin(2*pi*400*t);
sound(y, 22050);
```



```
t = 0: 1/22050: 2;
y = sin(2*pi*400*t);
sound(y, 44100);
```

A. The number 22050 is the sample rate of playback in Hz

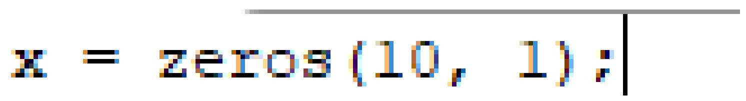
B. Sound will be 2 seconds and there will be 44101 samples

C.

  a. 8000Hz is a deeper longer tone with a longer play time, this can be explained since the sample rate is now about 2.5 times smaller but the number of samples hasn’t changed its effectively playing it in slow motion hence the deeper tone and the longer playtime.

  b. 44100Hz is a much louder and sharper tone with a shorter play time, this can be explained since the sample rate is 2 times larger with the same number of samples its effectively playing it in fast forward hence the sharper tone and the shorter tone.

2.6



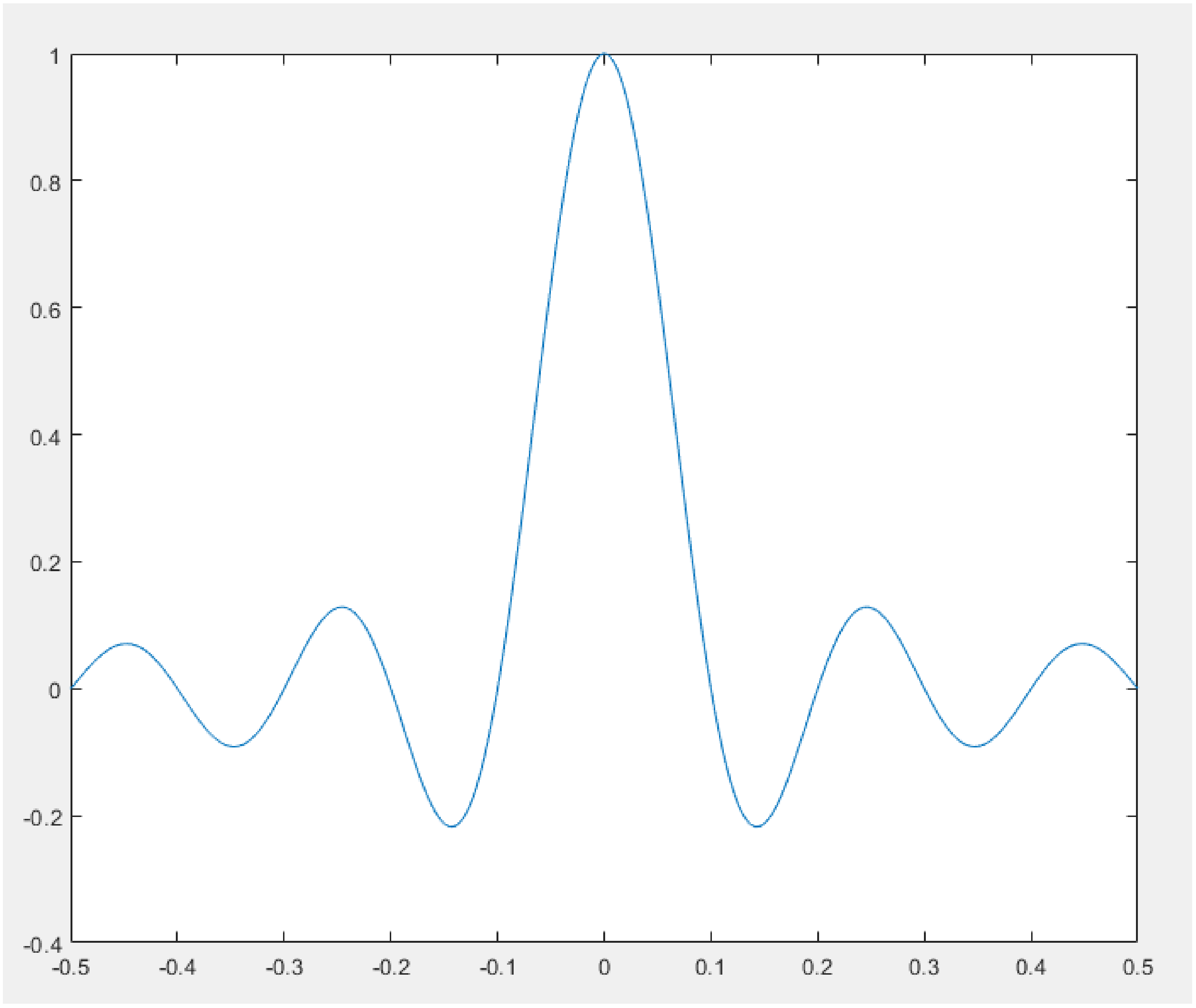
```
x = zeros(10, 1);
```

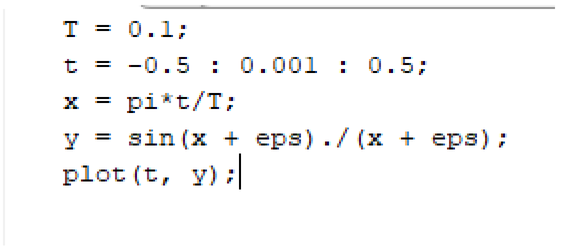
A. Since there was only 1 value given to the zeros function it will make a square matrix with the number given so instead of having 32768 zeros you'd get 1073741824 (32768 x 32768) zeros. To get the desired buffer you'd need to use x = zeros(32768, 1);

2.7

A. The second one was most likely the one that was intended. The second statement checks if any of the elements in 'h' are greater than 0.

2.8



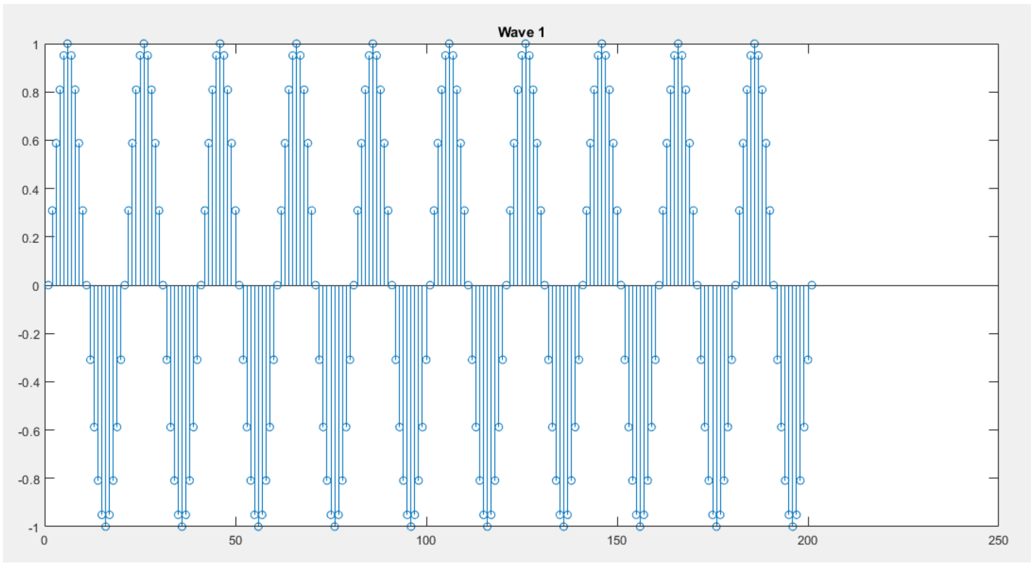


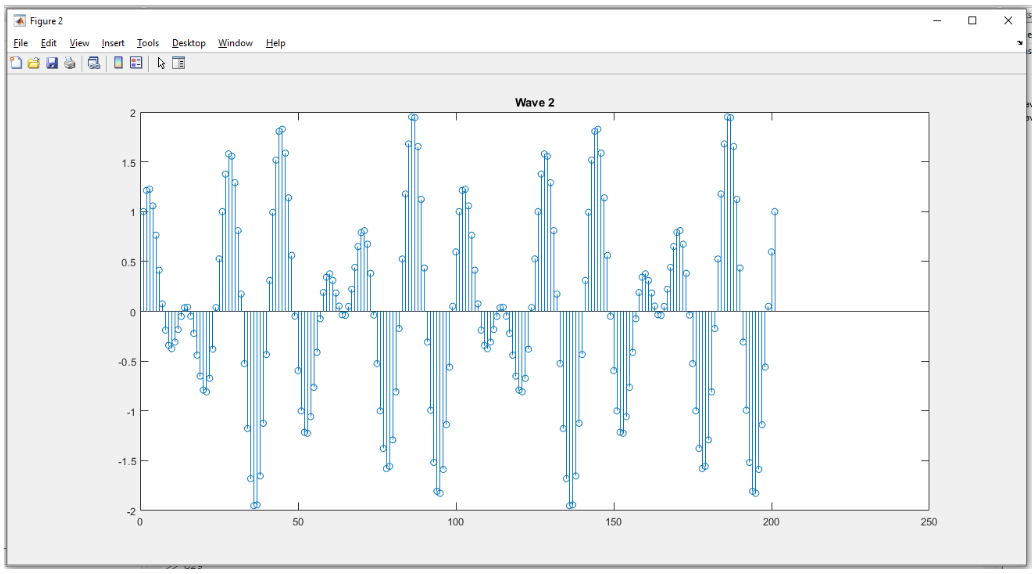
```
T = 0.1;
t = -0.5 : 0.001 : 0.5;
x = pi*t/T;
y = sin(x + eps)./(x + eps);
plot(t, y);
```

A. Its incorrect because the division that was done was not element division and since the graph needed each element then you would need to use element by element division (**./**). The purpose of the eps value is to protect the program from division by 0.

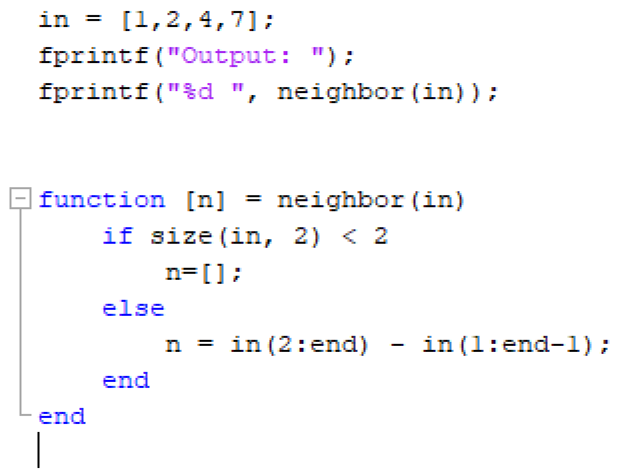
Section 4

```
x = 0: pi/100 : 2*pi;

wave1 = sin(x*10);
wave2 = sin(x*10) + cos(x*14);

figure(1);
stem(wave1);
title('Wave 1');

figure(2);
stem(wave2);
title('Wave 2');
```



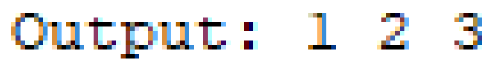


Section 5



```
in = [1,2,4,7];
fprintf("Output: ");
fprintf("%d ", neighbor(in));


function [n] = neighbor(in)
    if size(in, 2) < 2
        n=[];
    else
        n = in(2:end) - in(1:end-1);
    end
end
```

Output:



```
Output: 1 2 3
```