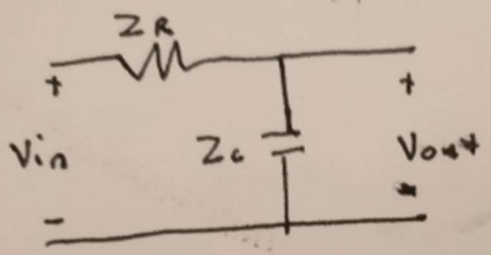
Johnson Le

CMPE 167

Homework 0

① Low-pass filter.

$$f_c = \frac{1}{2\pi RC} = 10.61\ \text{Hz}$$



$$Z_R = R_1$$

$$Z_c = \frac{1}{sC_1}$$

$$\frac{V_{out}}{V_{in}}(s) = \frac{1}{sC_1R_1 + 1}$$

② Inverting Amplifier

$$\text{Gain} = -\frac{R_f}{R_i} = -\frac{R_2}{R_3} = -\frac{10K}{20K} = -0.5$$

I expect this inverting amplifier to work.

③ $$A_v = \frac{V_{out}}{V_{in}} = \frac{A_p\left(f/f_c\right)}{\sqrt{1+\left(f/f_c\right)^2}}$$

$$= \frac{2\cdot\left(\frac{2k}{2k}\right)}{\sqrt{1+\left(\frac{2k}{2k}\right)^2}} = \frac{2}{\sqrt{2}}$$

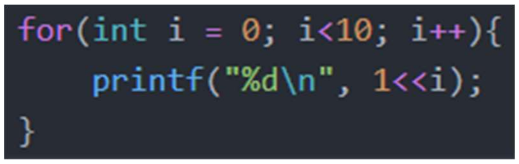
$$= 1.414$$

$$V_{out} = V_{in}\cdot A_v$$

$$= 1V\cdot 1.414 = 1.414V$$

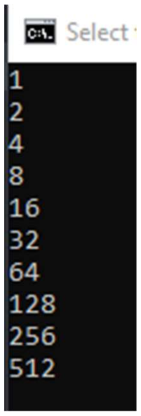
4)

CODE:



```c
for(int i = 0; i<10; i++){
    printf("%d\n", 1<<i);
}
```

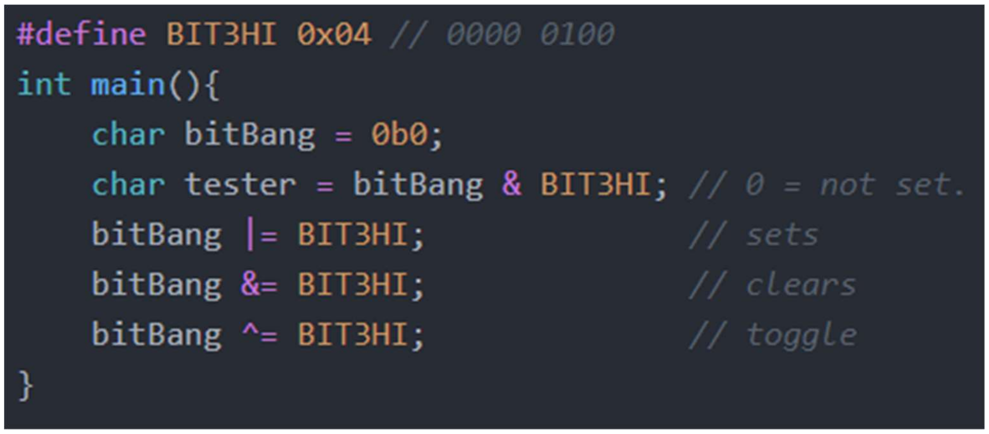
OUTPUT:



```
1
2
4
8
16
32
64
128
256
512
```

5)

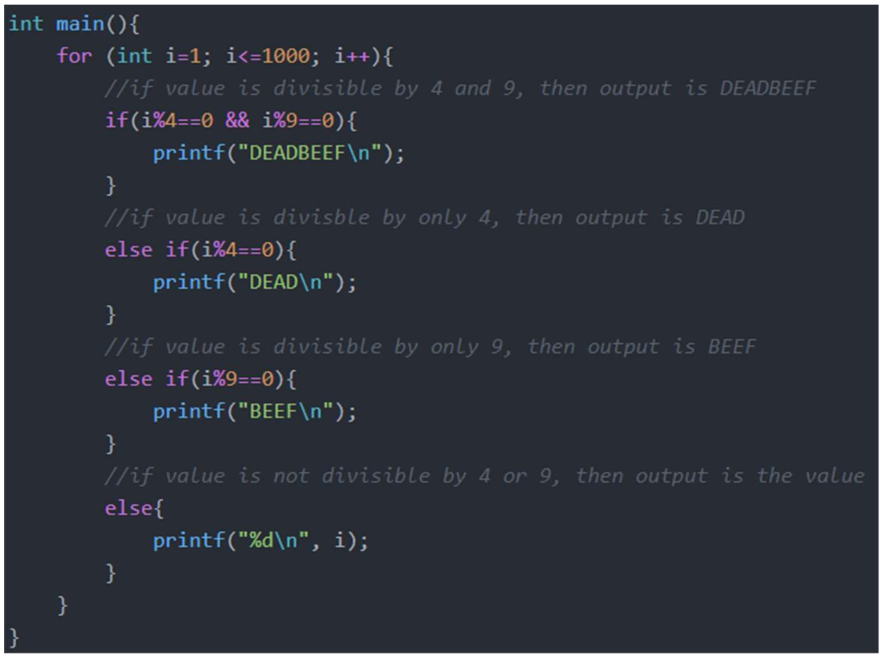
code:



```c
#define BIT3HI 0x04 // 0000 0100
int main(){
    char bitBang = 0b0;
    char tester = bitBang & BIT3HI; // 0 = not set.
    bitBang |= BIT3HI;              // sets
    bitBang &= BIT3HI;              // clears
    bitBang ^= BIT3HI;              // toggle
}
```

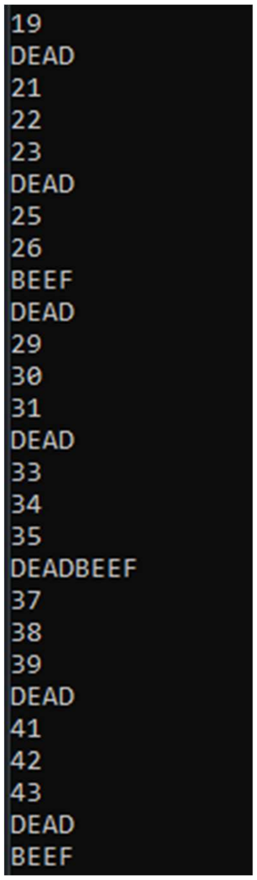
6)

Code:



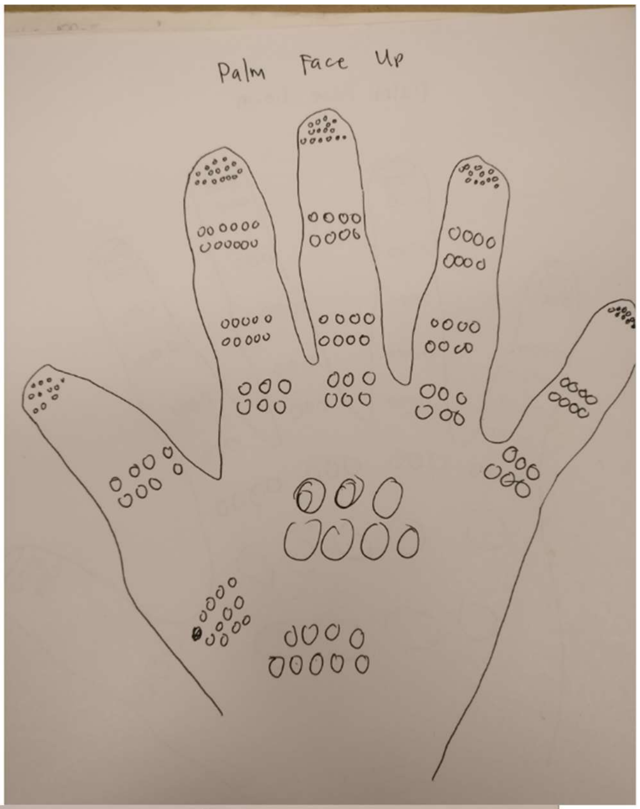
```c
int main(){
    for (int i=1; i<=1000; i++){
        //if value is divisible by 4 and 9, then output is DEADBEEF
        if(i%4==0 && i%9==0){
            printf("DEADBEEF\n");
        }
        //if value is divisble by only 4, then output is DEAD
        else if(i%4==0){
            printf("DEAD\n");
        }
        //if value is divisible by only 9, then output is BEEF
        else if(i%9==0){
            printf("BEEF\n");
        }
        //if value is not divisible by 4 or 9, then output is the value
        else{
            printf("%d\n", i);
        }
    }
}
```

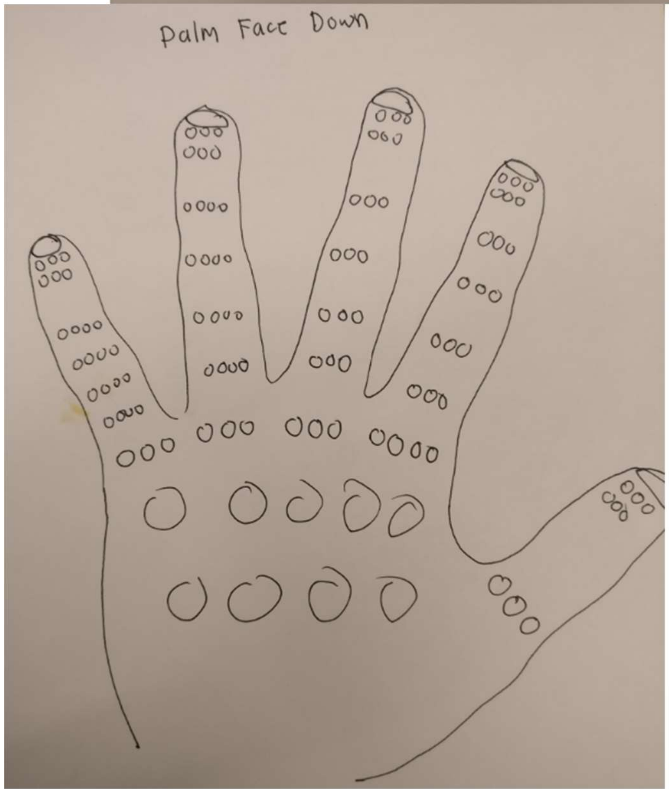
Output: Only a small section because 1000 lines is too much.



```
19
DEAD
21
22
23
DEAD
25
26
BEEF
DEAD
29
30
31
DEAD
33
34
35
DEADBEEF
37
38
39
DEAD
41
42
43
DEAD
BEEF
```

7)





In the drawings above, the smaller circles indicate that I can distinguish between two individual points closer. For this experiment, I used a pair of scissors to poke myself. This allowed me to control the distance between the blades with ease. The palm side is more sensitive in general.

The fingertips are the most sensitive part for sure as there was a big enough difference. Going down the fingertips on the palm side, the gap between the blades increased slowly. The backside however had the blade distance kept the same. For both palm and backside, towards the outside of the palm is more sensitive. Traveling inwards, the sensitivity decreases.

8)

Blank. Question hurts my brain.

9)

I began by printing out the provided blind spot image on a decently large scale. I couldn't get the dot or mouse to disappear by trying to move the piece of paper closer. When I looked at the image from the computer screen however, I did notice the dot and mouse disappear as I moved closer. Covering my left eye and focusing on the crosses made the dot and mouse disappear. Covering my right eye and focusing on the dot and mouse made the crosses disappear. I believe this occurs because each eye has a blind spot so when both eyes are open, they accommodate each other's blinds spots. When covering one eye and entering a distance where one of the objects in the image disappear, the brain tries to fill in the gap with the surrounding background. When the dot and crosses disappeared, all I saw was white. When the mouse disappears, the lines were still there.

From personal experience, I've noticed that sensors can have blind spots as well. For example, an IR receiver trying to detect a transmitter may not work if it is too close.