Question 1  *Software Vulnerabilities*  
(20 min)

For the following code, assume an attacker can control the value of `basket` passed into `eval_basket`. The value of `n` is constrained to correctly reflect the number of elements in `basket`.

The code includes several security vulnerabilities. Circle three such vulnerabilities in the code and briefly explain each of the three.

```c
struct food {
    char name[1024];
    int calories;
};

/* Evaluate a shopping basket with at most 32 food items.
   Returns the number of low-calorie items, or -1 on a problem. */
int eval_basket(struct food basket[], size_t n)
{
    struct food good[32];
    char bad[1024], cmd[1024];
    int i, total = 0, ngood = 0, size_bad = 0;

    if (n > 32)
        return -1;

    for (i = 0; i <= n; ++i)
    { // snprintf(buf, len, fmt, ...) works like printf, but instead writes to buf, and won’t write more than len - 1 characters. It terminates the characters written with a '\0'. system runs the shell command given by its first argument.
        if (basket[i].calories < 100)
            good[ngood++] = basket[i];
        else if (basket[i].calories > 500)
        {
            size_t len = strlen(basket[i].name);
            snprintf(bad + size_bad, len, "%s", basket[i].name);
            size_bad += len;
        }
        
        total += basket[i].calories;
    }

    if (total > 2500)
    { // snprintf(buf, len, fmt, ...) works like printf, but instead writes to buf, and won’t write more than len - 1 characters. It terminates the characters written with a '\0'. system runs the shell command given by its first argument.
        const char *fmt = "health-factor --calories %d --bad-items %s";
        fprintf(stderr, "lots of calories!");
        snprintf(cmd, sizeof cmd, fmt, total, bad);
        system(cmd);
    }

    return ngood;
}
```

Reminder: `strlen` calculates the length of a string, not including the terminating `\0` character. `snprintf(buf, len, fmt, ...)` works like `printf`, but instead writes to `buf`, and won’t write more than `len - 1` characters. It terminates the characters written with a `\0`. `system` runs the shell command given by its first argument.