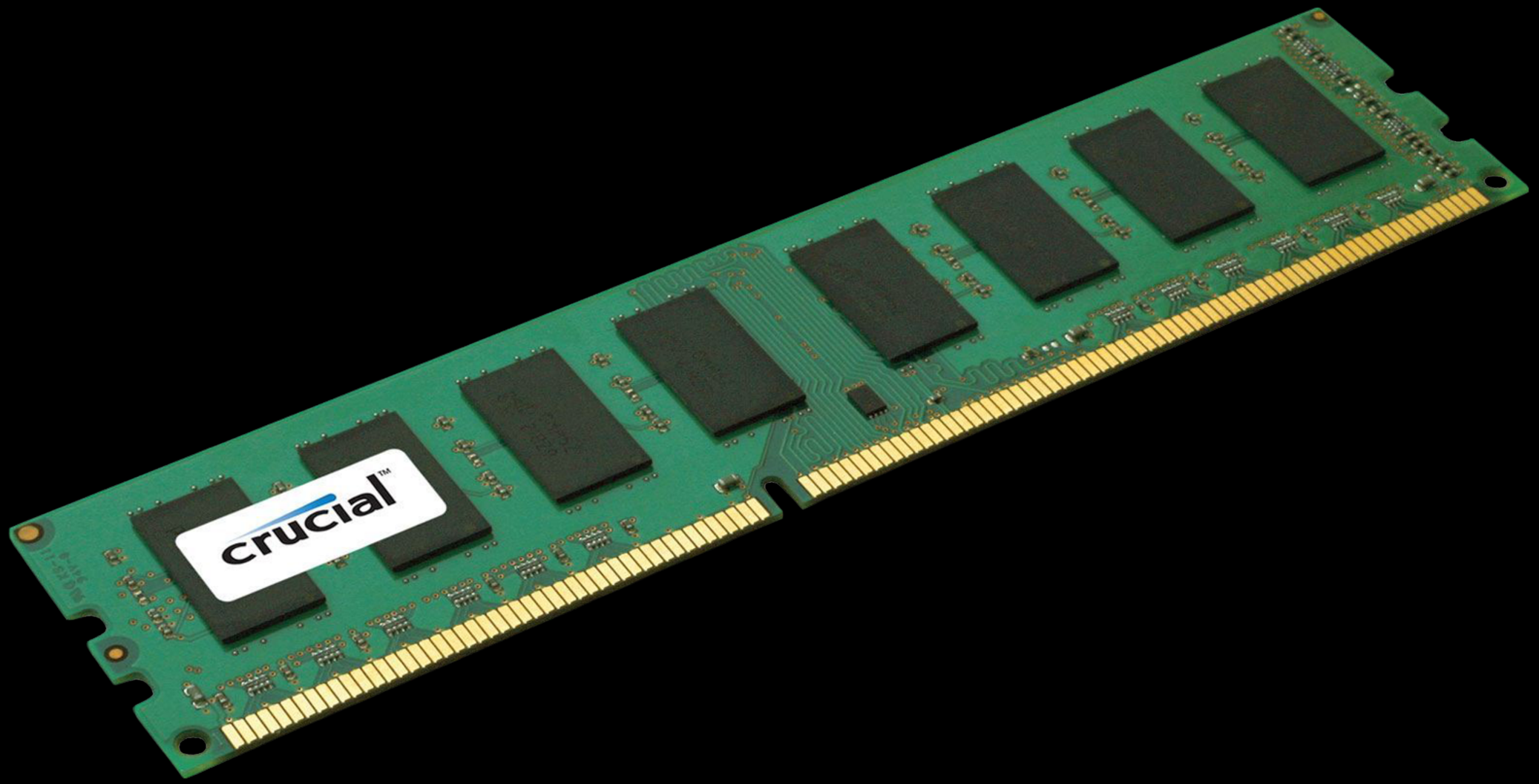
math teacher vs computer


crucial™

cough0.c | cough1.c | animal.c | float.c | xyfloat.c

```c
#include <cs50.h>
#include <stdio.h>

int main(void)
{
    // 사용자에게 x 값 받기
    float x = get_float("x: ");

    // 사용자에게 y 값 받기
    float y = get_float("y: ");

    // 나눗셈 후 출력
    printf("x / y = %f\n", x / y);
}
```

Terminal

```
$ ./xyfloat
x: 1
y: 10
x / y = 0.100000
$
```

cough0.c | cough1.c | animal.c | float.c | xyfloat.c

```c
#include <cs50.h>
#include <stdio.h>

int main(void)
{
    // 사용자에게 x 값 받기
    float x = get_float("x: ");

    // 사용자에게 y 값 받기
    float y = get_float("y: ");

    // 나눗셈 후 출력
    printf("x / y = %.50f\n", x / y);
}

```

Terminal

```
$ ./xyfloat
x: 1
y: 10
x / y = 0.100000
$ make xyfloat
clang -fsanitize=signed-integer-overflow -fsanitize=undefined -ggdb3 -O0 -std=c11 -Wall -Werror -Wextra -Wno-sign-compare -Wno-unused-parameter -Wno-unused-va
riable -Wshadow    xyfloat.c  -lcrypt -lcs50 -lm -o xyfloat
$ ./xyfloat
x: 1
y: 10
x / y = 0.10000000149011611938476562500000000000000000000000
$
```

cough0.c | cough1.c | animal.c | float.c | xyfloat.c | overflowint.c

```c
#include <stdio.h>
#include <unistd.h>

int main(void)
{
    for (int i = 1; ; i *= 2)
    {
        printf("%i\n", i);
        sleep(1);
    }
}

```

Terminal

```
$ make overflowint
clang -fsanitize=signed-integer-overflow -fsanitize=undefined -ggdb3 -O0 -std=c11 -Wall -Werror -Wextra -Wno-sign-compare -Wno-unused-parameter -Wno-unused-va
riable -Wshadow    overflowint.c  -lcrypt -lcs50 -lm -o overflowint
$ ./overflowint
1
2
4
8
16
32
64
128
256
512
1024
2048
```

cough0.c | cough1.c | animal.c | float.c | xyfloat.c | overflowint.c

```c
#include <stdio.h>
#include <unistd.h>

int main(void)
{
    for (int i = 1; ; i *= 2)
    {
        printf("%i\n", i);
        sleep(1);
    }
}

```

Terminal

```
16777216
33554432
67108864
134217728
268435456
536870912
1073741824
overflowint.c:6:25: runtime error: signed integer overflow: 1073741824 * 2 cannot be represented in type 'int'
-2147483648
0
0
0
0
0
^C
$
```

floating-point imprecision

integer overflow

123

124

125

126

127

128

129

1

120

130

999

1

990

1

900

111

1

110

1

100

1

000

000

integer overflow

1999

1999

1900

NOW HEAR THIS —

Boeing 787 Dreamliners contain a potentially catastrophic software bug

Beware of integer overflow-like bug in aircraft's electrical system, FAA warns.

DAN GOODIN - 5/2/2015, 2:55 AM

A software vulnerability in Boeing's new 787 Dreamliner jet has the potential to cause pilots to lose control of the aircraft, possibly in mid-flight, Federal Aviation Administration officials warned airlines recently.

The bug—which is either a classic integer overflow or one very much resembling it—resides in one of the electrical systems responsible for generating power, according to memo the FAA issued last week. The vulnerability, which Boeing reported to the FAA, is triggered when a generator has been running continuously for a little more than eight months. As a result, FAA officials have adopted a new airworthiness directive (AD) that airlines will be required to follow, at least until the underlying flaw is fixed.

"보잉787, 248일마다 재부팅해라"

FAA는 248일 안에 시스템 재시작을 해야 하는 이유에 대해 소프트웨어 문제라고만 설명했다. 하지만 248일은 컴퓨터나 전자 제품에서 가끔씩 보이는 숫자다. 오랫동안 연속 구동하는 서버나 라우터 같은 장비에서도 마찬가지. 연속가동시간 등을 기록하는 내부 카운터에 32비트 정수를 사용하는데 이 과정에서 예기치 못한 동작이 발생하는 것. 이 숫자의 2배인 497일 문제의 경우 윈도 서버 2008이나 윈도 비스타, 7 등에서도 발생한 바 있다.

보잉 787의 경우 248일 이상 계속되면 발전기를 제어하는 시스템이 안전 모드로 들어가 일제히 AC 전원이 꺼져 비행 중이라면 통제 불능이 될 우려가 있다는 것이다. 보잉은 이 문제를 해결하기 위한 패치를 작업 중이지만 적용 시기는 밝혀지지 않았다. 관련 내용은 이곳에서 확인할 수 있다.

This is Kyumin Lee