

[1]

```
p[2] // 104

sizeof(p) // 8

m->msg + 1 // char *

"0"[1]==0 && "0"[1]=='\0' // 1

*argv++ // char *


m->msg - (char *)m // 16

&m[0] // struct MdbRec *

sizeof(m[0].msg) // 24

&p // int **

x + y // -2


c[2] // 255
```

```
&c                                // unsigned char **

(~x) & y                          // 0

sizeof(a) / sizeof(a[0])          // 10

*&*a[5]                            // 105
```

2 points each, no partial credit, except:

(1.7) MdbRec * is acceptable and gets full credit.

(1.12) char ** is acceptable and gets full credit.

[2]

(a)

ABCabc123

- ABCabc123\n is acceptable and gets full credit.

(b)

7 bytes

- no partial credit

(c)

NONE

- reduction of -1 point for each line falsely identified up to -5.

(d)

```
32: free(p);
```

[3]

(a)

```
{Romney} said {binders full of women}
```

```
{Romney} said {binders full of women}
```

(b)

40 bytes.

- no partial credit

(c)

22,23: `p = (struct MdbRec *)malloc(sizeof(struct MdbRec));`

23,24: `free(p);`

26,27: `free(p);`

- 5 points for each line correctly identified, with BOTH of the existing

line numbers enclosing the new line.

- casting to `(struct MdbRec *)` can be omitted.
- "MdbRec" instead of "struct MdbRec" is fine.
- `malloc(40)` is fine.

A solution can be crafted in a few different ways, but you need to use a

static variable in all cases. Two different solutions are shown below:

```
int f()
{
    static int x = 0;
    return ++x;
}
```

```
int f()
{
    static int i = 0;
    i++;
    if (i == 10)
        return 55;
    else
        return 0;
}
```

- No credit if static variable is not declared inside f.