

CS240 Midterm Solution, summer 2024

P1(a) 15 pts

A segmentation fault is likely to arise when executing statement `*y = 6`.
5 pts

Since the content of `y` has not been properly initialized, `y` will likely contain an address that does not belong to the running program.
5 pts

`printf()` would have successfully completed, returning to `main()`, but the output would likely not be visible on `stdout` (i.e., display).
5 pts

// Since output is temporarily held in main memory without flushing to
// `stdout` when the program abnormally terminated.

P1(b) 15 pts

`x = ~(~0 << 16)`
// Expressions that do not utilize bit processing techniques such as
// `x = 0x00FF` receive minimal partial credit.
15 pts

P1(c) 15 pts

`scanf("%c", &h[3]);`
8 pts

`scanf("%c", h+3);`
7 pts

P2(a) 15 pts

The code will likely generate a stack smashing error (and terminate abnormally).
The reason is that `z` being local resides in `main()`'s stack frame which contains `main()`'s return address.
// The canary guarding the return address is likely to be overwritten by the
// loop exceeding its valid bound.
5 pts

If `z` is global then stack smashing will not occur since `z` is not in a region that contains a return address.
// It may, or may not, generate segmentation fault. In the example discussed in class, it did not.
5 pts

If `z` is a pointer and `malloc()` is used to allocate heap memory, then it is similar to the global case.
// Heap memory resides above `a.out` as do globals.
5 pts

P2(b) 15 pts

```

int countnum = 0;
int countascii = 0;
4 pts

while((y = fgetc(fp)) != EOF) {
// 5 pts
    countnum++;
    if(0 <= y <= 127) countascii++;
// 6 pts
}
// Instead of range checking bit processing may utilized to check if the 8'th
bit
// is 0.

```

P3 25 pts

To determine how many lines: read byte-by-byte checking how many bytes have value '\n'.
5 pts

To determine maximum line length: use a counter initialized to 0 that is incremented when a byte is read until newline is reached (count includes newline). Compare current count to previous maximum count. If strictly bigger update maximum count.
5 pts

essayptr needs to point to a memory region of N contiguous pointers. Each of the N pointers points to a region M contiguous memory locations (total M bytes).
// This is wasteful since M is memory required for maximum line length,
7 pts

```

essayptr = malloc(N * sizeof(char *));
Allocates N contiguous memory locations each of size to hold a char pointer,
i.e., char *.
8 pts

```

Bonus 10 pts

fgetc() needs to indicate that the end of file has been reached which it does by returning EOF (-1).
5 pts

The return type of fgetc() cannot be char since all 8 bits are needed to specify the content of the byte read. Hence setting the return type to int (4 bytes) allows the 8-bit content to be stored in the first byte of int, and the additional 3 bytes allows -1 to be represented.
5 pts