This document contains solutions to the CS107 midterm given in Winter 2018 by instructor Chris Gregg. This was a 120-minute exam.

Solutions

1a)
```
16
64
0
```

1b) A non-zero value is return for any n such that `1 <= n <= 127`

1c) The value returned is `n` rounded down to the nearest exact power of two. Alternatively: the return value retains the most significant bit of the original number, and zeros all bits less significant.

2a)
```
char *substr(const char *s, char start, char stop, char result[]) {
    result[0] = '\0'; // initialize result to empty string
    char *first = strchr(s, start);
    if (!first) {
        return result;
    }
    char *last = strchr(first + 1, stop);
    if (!last) {
        return result;
    }
    int len = last - first + 1;
    strncpy(result, first, len);
    result[len] = '\0';
    return result;
}
```

2b) `malloc(strlen(result))` allocates 1 fewer byte than needed (no space for null terminator) and `result` may not have been allocated on the heap, so freeing it could cause a runtime error.

3a)
```
bool queue_dequeue(queue *q, void *addr) {
    if (q->front == NULL) {
        return false;
    }
    node *to_remove = q->front;
    q->front = to_remove->next;
    if (!q->front) {
        q->back = NULL;
    }
    memcpy(addr, to_remove->data, q->width);
    free(to_remove->data);
    free(to_remove);
    return true;
}
```

3b)
```
int main(int argc, char **argv) {
        char buffer[1024];
        int nlines = atoi(argv[1]);
        FILE *fp = fopen(argv[2], "r");
        queue *q = queue_create(sizeof(char *)); // line 1
        int lines_read = 0;
        char *line;
        while (fgets(buffer, sizeof(buffer), fp)) {
            buffer[strlen(buffer) - 1] = '\0';
            line = strdup(buffer);   // line 2
            queue_enqueue(q, &line); // line 3
            if (++lines_read > nlines) {
                queue_dequeue(q, &line); // line 4
                free(line); // line 5
            }
        }
        fclose(fp);
        while (queue_dequeue(q, &line)) { // line 6
            printf("%s\n", line);
            free(line); // line 7
        }
        free(q); // line 8
        return 0;
    }
```

4)
```
int cmp_date(const void *a, const void *b) {
       const struct date *one = (const struct date *)a;
       const struct date *two = (const struct date *)b;
       if (one->year == two->year) {
           return one->month - two->month;
       }
       return one->year - two->year;
  }
```

5)
```
void map(void *arr, int n, size_t width, void (*fn)(void *)) {
       for (int i = 0; i < n; i++) {
           fn((char *)arr + i * width);
       }
}
```