Com S 327
Fall 2017
Midterm Exam

DO NOT OPEN THIS EXAM UNTIL INSTRUCTED TO DO SO

Name: ______________________________

ISU NetID (username): ______________________________

Closed book and notes, no electronic devices, no headphones. Time limit 45 minutes. Partial credit may be given for partially correct solutions.

- Use correct C syntax for writing code.
- You are not required to write comments for your code; however, brief comments may help make your intention clear in case your code is incorrect.

If you have questions, please ask!

Question	Points	Your Score
1	30	
2	40	
3	30	
EC	1	
Total	100	

1. (30 pts; 3 each) For each code snippet, either give its output, indicate that a runtime error occurs (which does not necessarily imply that the program crashes), or indicate that it runs cleanly but produces no output. Invoking undefined behavior should be considered a runtime error. Exactly one of these four cases occurs for each problem.

   Be careful! These problems test more than just your knowledge and understanding of how the I/O functions work. In particular, if two of them look essentially the same, you should pay close attention to the differences.

   `char *strcpy(char *dest, const char *src)` copies the string pointed to by `src`, including the terminating null byte (`'\0'`), to the buffer pointed to by `dest`. You should have extensive experience with the rest of the functions used below, but please ask if you need explanations.

   You do not need to write newlines in your answers. They are included in the code only for the sake of completeness.

   Problem parts are fully independent.

```
enum {
  troy,   gabriella, ryan,   sharpay,
  kelsi,  chad,      zeke,   taylor
};

char *song[] = {
  "Breaking␣Free",              "Start␣of␣Something␣New",
  "Stick␣to␣the␣Status␣Quo",    "Bop␣to␣the␣Top",
  "What␣I've␣Been␣Looking␣For", "Get'cha␣Head␣In␣The␣Game",
  NULL,                          NULL
};

char character[][16] = {
  "Troy␣Bolton",   "Gabriella␣Montez", "Ryan␣Evans",   "Sharpay␣Evans",
  "Kelsi␣Nielsen", "Chad␣Danforth",    "Zeke␣Baylor",  "Taylor␣McKessie"
};
```

(a)
```
char *s = "Breaking␣Free";
printf("%s\n", s);
```

(b)
```
printf("%s:␣%s\n", character[sharpay], song[sharpay]);
```

(c)

```
strcpy(character[troy], "Jack");
printf("%s\n", character[troy]);
```

(d)

```
strcpy(character[troy], "Jack");
character[troy][4] = '␣';
printf("%s\n", character[troy]);
```

(e)

```
strcpy(song[sharpay], song[kelsi]);
printf("%s\n", song[sharpay]);
```

(f)

```
/* Careful! */
printf("%s:␣%s\n", character[gabriella], song[gabriella]);
```

(g)

```
strcpy(character[gabriella] + 9, character[troy] + 5);
printf("%s:␣%s\n", character[gabriella], song[gabriella]);
```

(h)

```
printf("%s\n", song[zeke]);
```

(i)

```
printf("%s\n", ((char *) character) + 64);
```

(j)

```
printf("%s\n", *((char **) song) + 50);
```

2. (40 pts; 20 each) Complete the following functions according to the given specifications. You may not use any other functions (e.g., from the standard library or otherwise assumed) except, if necessary, `malloc()` and `free()`. You may not use any non-local variables. You may not write and use any "helper" functions. You may not leak memory; however, if the function is defined to return the address of dynamically allocated storage, it is the responsibility of the user to free that storage, so returning that address without freeing it is not considered a leak. In all cases, you may assume that all arguments are non-NULL.

   (a) The `strcasecmp()` function performs a byte-by-byte comparison of the strings `s1` and `s2`, ignoring the case of the characters. It returns an integer less than, equal to, or greater than zero if `s1` is found, respectively, to be less than, to match, or be greater than `s2`.

   Hint: You can convert uppercase to lowercase by bitwise or'ing it with `0x20`, but this conversion is only valid if the character is alphabetical. You can check if a character is alphabetical with the predicate function `isalpha()`; **however, to simplify this exercise, you may assume that all characters are alphabetical and forgo tests for such!**

```
int strcasecmp(const char *s1, const char *s2);
{










}
```

(b) The strcspn() function calculates the length of the initial segment of s which consists entirely of bytes not in reject.

size_t is an integer type which is guaranteed to be large enough to hold any indexable size on the systems (i.e., 32 bits on a 32-bit system and 64 bits on a 64-bit system).

```
size_t strcspn(const char *s, const char *reject);
{

}
```

3. (30 pts; 3 each) Circle the correct statement for each problem that follows. You may assume that all parts are independent and that all code appears in a valid context.

   (a)
   ```
   int *a;
   free(a);
   ```

   This code will: (a) run correctly; (b) invoke undefined behavior or crash; (c) not compile

   (b)
   ```
   int i[5];
   free(i);
   ```

   This code will: (a) run correctly; (b) invoke undefined behavior or crash; (c) not compile

   (c)
   ```
   int *i = malloc(15 * sizeof (*i));
   free(i);
   ```

   This code will: (a) run correctly; (b) invoke undefined behavior or crash; (c) not compile

   (d)
   ```
   int *i = malloc(15 * sizeof (*i));
   int **p = &i;
   free(*p);
   ```

   This code will: (a) run correctly; (b) invoke undefined behavior or crash; (c) not compile

   (e)
   ```
   int *i, *j;
   i = j = malloc(15 * sizeof (*i));
   free(i);
   free(j);
   ```

   This code will: (a) run correctly; (b) invoke undefined behavior or crash; (c) not compile

(f)

```
int *i = malloc(15 * sizeof (*i));
free(*i);
```

This code will: (a) run correctly; (b) invoke undefined behavior or crash; (c) not compile

(g)

```
char s[10];
s = "Mrs.␣Darbus";
```

This code will: (a) run correctly; (b) invoke undefined behavior or crash; (c) not compile

(h)

```
char *s = "Taylor";
strcat(s, "␣McKessie");
```

This code will: (a) run correctly; (b) invoke undefined behavior or crash; (c) not compile

(i)

```
char *s = strndup("Sharpay␣Evans", 5);
free(s);
```

This code will: (a) run correctly; (b) invoke undefined behavior or crash; (c) not compile

(j)

```
char *s = strndup("Zeke␣Baylor", 5);
s++;
free(s - 1);
```

This code will: (a) run correctly; (b) invoke undefined behavior or crash; (c) not compile

Extra Credit. (1 pt) This is mostly just for fun, and it's only one point, so don't even waste time looking at it unless you're done with everything else. One of the TAs said that there needs to be a "hard question".

Give the output of the following program (Note: this will give different output on big- and little-endian hardware; assume that it is running on a little endian machine):

```
#include <stdio.h>

int main(int argc, char *argv[])
{
  unsigned M = 1;
  unsigned A = 3;
  unsigned T = 79;
  unsigned H = 221;
  unsigned E = 15263;
  unsigned W = 26683;

  unsigned prod[] = {
    W * H * A * T,
    T * E * A * M
  };

  printf("%s\n", (char *) prod);

  return 0;
}
```