hxie13

1. Write a program that uses write() to print out "Hi! My name is <Your Name>".

```
#include <unistd.h>
int main() {
	write(1, "Hi! My name is Handi Xie", 20);
	return 0;}
```

Grade: 50%

2. Write a function to print out a triangle of height n to standard error.

```
#include <unistd.h>

void write_triangle(int n){
	if (n <= 0){
		exit(1);
	}
	int len, star;
	for(len = 1; len <= n ; len++) {
		for (star = 0; star < len; star++){
			write(STDERR_FILENO, "*", 1);
		}
		write(STDERR_FILENO, "\n", 1);
	}
}

int main() {
	//testing
	write_triangle(0);

	return 0;}
```

Grade: 100%

3. Take your program from "Hello, World!" and modify it write to a file called "hello_world.txt".

```
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <unistd.h>

int main() {
	mode_t mode = S_IRUSR | S_IWUSR;
	int fildes = open("hello_world.txt", O_CREAT | O_TRUNC | O_RDWR, mode);
	write(fildes, "Hi! My name is Handi Xie", 20);
	close(fildes);
	return 0;}
```

Grade: 100%

4. Take your program from "Writing to files" and replace write() with printf().

```
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <unistd.h>

int main() {
       mode_t mode = S_IRUSR | S_IWUSR;
       close(1);
       int fildes = open("hello_world1.txt", O_CREAT | O_TRUNC | O_RDWR, mode);
       printf("%s", "Hi! My name is Handi Xie");
       close(fildes);
       return 42;}
```

Grade: 100%

5. What are some differences between write() and printf()?

```
Write() is designed to only write a sequence of bytes and is considered too basic.
Printf() is a function that convert your data into a formatted sequence of bytes
   and that calls write() to write those bytes onto the output.
-------- Quoted from Stackoverflow
```

Grade: 100%

6. How many bits are there in a byte?

```
8
```

Grade: 100%

7. How many bytes are there in a char?

```
1
```

Grade: 100%

8. How many bytes are each of the following on your machine?

```
4 8 4 4 8
```

Grade: 100%

9. Refer to code snippet below. If the address of data is 0x7fbd9d40, then what is the address of data+2?

```
0x7fbd9d48
```

Grade: 0%

10. What is data[3] equivalent to in C?

```
0x7fbd9d4c
```

Grade: 0%

11. Why does the code snippet below segfault?

```
String is immutable
```

Grade: 100%

12. What does sizeof("Hello\0World") return?

```
12
```

Grade: 100%

13. What does strlen("Hello\0World") return?

```
5
```

Grade: 100%

14. Give an example of X such that sizeof(X) is 3.

```
char* temp = "241"; size of temp is 3
```

Grade: 0%

15. Give an example of Y such at sizeof(Y) might be 4 or 8 depending on the machine.

```
Pointers will be 4 on a 32-bit system, and 8 on a 64-bit system
```

Grade: 100%

16. What are two ways to find the length of argv?

```
argc is the length of argv; or loop over until argv[index] points to NULL
```

Grade: 100%

17. What does argv[0] represent?

```
The program name
```

Grade: 50%

18. Where are the pointers to environment variables stored?

```
 top of the process memory layout, above the stack | they are not stored in files
     but in the process' own memory
```

Grade: 100%

19. Refer to the code snippet below. What are the values of sizeof(ptr) and sizeof(array)?

```
sizeof(ptr) is 4 because that is the size of a pointer, and size of array is 6
   because the size of the char array represents the 5 bytes it uses for "Hello"
   and also another for the "\0" in the end
```

Grade: 75%

Feedback: The question says the size of pointers is 8 bytes.

20. What data structure manages the lifetime of automatic variables?

```
Stack
```

Grade: 100%

21. If I want to use data after the lifetime of the function it was created in ends, where should I put it? How do I put it there?

```
Heap by using malloc, realloc, and calloc
```

Grade: 100%

22. Fill in the blank: "In a good C program, for every malloc, there is a ___".

```
free
```

Grade: 100%

23. What is one reason malloc can fail?

```
Not enough space
```

Grade: 100%

24. What are some differences between time() and ctime()?

```
Time() returns the seconds after 1970 and as a time_t object
Ctime() interprets the value pointed by timer as a calendar time and converts it to
    a C-string containing a human-readable version of the corresponding time and
   date, in terms of local time, and it returns in a C-type string.
quoted from cplusplus.com
```

Grade: 100%

25. What is wrong with this code snippet?

```
Double free
```

Grade: 100.0%

Submitted: 9/1/2017 23:35:01

26. **What is wrong with this code snippet?.1**

```
Make use of freed pointers.
```

Grade: 100%

27. **How can one avoid the previous two mistakes?**

```
Set them to NULL to avoid dangling pointers.
```

Grade: 100%

28. **Create a struct that represents a Person, and then make a typedef, so that "struct Person" can be replaced with a single word.**

```
#include <stdio.h>

struct Person{
       char* name;
       int* age;
       struct Person* friends;
}

typedef struct Person person_t;

int main() {
       return 0;
}
```

Grade: 100%

29. **Now make two persons on the heap, "Agent Smith" and "Sonny Moore", who are 128 and 256 years old respectively and are friends with each other.**

```
#include <stdio.h>

struct Person{
       char* name;
       int* age;
       struct Person* friends;
}

typedef struct Person person_t;

int main() {
       person_t* p1 = (person_t*) malloc(sizeof(person_t));
       person_t* p2 = (person_t*) malloc(sizeof(person_t));
       p1 -> name = "Agent Smith";
       p1 -> age = 128;
       p2 -> name = "Sonny Moore";
       p2 -> age = 256;
       return 0;
}
```

Grade: 70%

Feedback: friend definition wrong

30. Create functions to create and destroy a Person on the heap.

```
#include <stdio.h>

struct Person{
        char* name;
        int* age;
        struct Person* friends;
}

typedef struct Person person_t;

person_t* person_create(char* aname, int* aage) {
        person_t * prn = (person_t) malloc(sizeof(person_t));
        if (aname == NULL){
                ret -> name = "Average Joe";
        }
        if (aage == NULL){
                ret -> age = 18;
        }
        prn -> name = strdup(aname);
        prn -> age = strdup(aage);
        person_t * afriends = (person_t) malloc(sizeof(person_t)*10);
        prn -> friends = afriends;
        return prn;
}

person_t* person_destroy(person_t* prn){
        free(prn->name);
        free(prn->age);
        memset(prn->friends, 0, sizeof(person_t));
        free(prn);
}

int main() {

        return 0;
}
```

Grade: 100%

31. What functions can be used for getting characters from stdin and writing them to stdout?

```
gets() and puts()
```

Grade: 100%

32. Name one issue with gets().

```
It needs to have a buffer declared and it could have been overflown and you can't
   tell whether the input is too long for it.
```

Grade: 100%

33. Write code that parses the string "Hello 5 World" and initializes 3 variables to "Hello", 5, and "World").

```
#include <stdio.h>

int main() {
        char * data = "Hello 5 World";

        char buffer[20];
        int score = -42;
        char buffer2[20];

        sscanf(data, "%s %d %s", buffer, & score, buffer2);
        return 0;
}
```

Grade: 100%

34. What does one need to define before including getline()?

```
#define _GNU_SOURCE
```

Grade: 100%

35. Write a C program to print out the contents of a file line-by-line using getline().

```
int main() {
        FILE * fp; // to be initialized
        char *buffer = NULL;
        size_t capacity = 0;
        ssize_t result = getline(&buffer, &capacity, fp);

        while (result!=-1){
                printf("%s\n", buffer);
                result = getline(&buffer, &capacity, fp);
        }
}
```

Grade: 90%

Feedback: Had to use fopen to actually open any random file.

36. What compiler flag is used to generate a debug build?

```
-g
```

Grade: 100%

37. You modify the makefile to generate debug builds and type make again. Explain why this is insufficient to generate a new build.

```
Because a better way to make build is to use option flags in make file to generate
    a new build.
```

Grade: 0%

38. Are tabs or spaces used to indent the commands after the rule in a Makefile?

```
tab
```

Grade: 100%

39. What are the differences between heap and stack memory?

```
Heap stays and stacks get zeroed out by stack pointer after used, and heap needs to
     be freed.
```

Grade: 100%

40. Are there other kinds of memory in a process?

```
Data Segment
```

Grade: 100%

41. Convert your a song lyrics into System Programming and C code covered in this wiki book and share on Piazza.

[No response]

42. Find, in your opinion, the best and worst C code on the web and post the link to Piazza.

[No response]

43. Write a short C program with a deliberate subtle C bug and post it on Piazza to see if others can spot your bug.

[No response]

Final grade: 85.875%