xinruiy2 - HW0

Comments

Ch1 Q4 Didn't close stdout Ch2 Q4. 0x7fbd9d50 Ch3 Q3. At the top of the stack. Q4. No explanation given Ch4 Q10. Person **friends Ch6 Q2. The makefile only looks at the timestamp

Timestamp

1/22/2018 20:59:23

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

```
#include <unistd.h>
int main(){
      write(1,"Hi! My name is Ray Ying\n", 24);
      return 0;
}
```

Grade: **100.0%**

2. Write a program that uses write() to print out a triangle of height n to Standard Error

```
#include <unistd.h>
int main(){
        int count = 1;
        for(;count <n+1; count++){
             while(int i = 1; i <=count; i++){
                     write(1, "*", 1);
             }
             write(1,"\n",1);
         }
        return 0;
}
```

Grade: **100.0%**

3. Take your program from "Hello World" and have it write to a file called "hello world.txt" (without the quotes).

```
#include <unistd.h>
#include<sys/types.h>
#include<sys/stat.h>
#include<fcntl.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, "Hello, World!", 13);
      close(fildes);
      return 0;
}
```

Grade: **100.0%**

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

```
#include<unistd.h>
#include<sys/types.h>
#include<sys/stat.h>
#include<fcntl.h>
#include<stdio.h>
int main(){
    mode_t mode = S_IRUSR | S_IWUSR;
    int fildes = open("hello_world.txt", O_CREAT | O_TRUNC | O_RDWR, mode);
    printf("Hello, World!\n");
    close(fildes);
    return 0;
}
```

Grade: **0.0%**

5. Name some differences from write() and printf()

```
write() is a system call and printf() has a buffer. It only calls when it is
    specific told to, finish a line or the buffer is full.
```

Grade: **100.0%**

1. How many bits are there in a byte?

```
At least 8 bits.
```

Grade: **100.0%**

2. How many bytes is a char?

```
1
```

Grade: **100.0%**

3. Tell me how many bytes the following are on your machine: int, double, float, long, long long

```
4 8 4 8 8
```

Grade: **100.0%**

4. On a machine with 8 byte integers (refer to code snippet below)

```
0x7fbd9d48
```

Grade: **0.0%**

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

```
*(data+3) or 3[data]
```

Grade: **100.0%**

6. Why does this segfault (refer to code snippet below)?

```
hello is a constant memory so it cannot be changed.
```

Grade: **100.0%**

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

```
12
```

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

```
5
```

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

```
"Ab"
```

Grade: **100.0%**

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

```
char*
```

Grade: **100.0%**

1. Name me two ways to find the length of argv

```
1. argc
2. loop
```

Grade: **100.0%**

2. What is argv[0]

```
The program name
```

Grade: **100.0%**

3. Where are the pointers to environment variables stored?

```
 They are stored in the process's own memory.
```

Grade: **0.0%**

4. On a machine where pointers are 8 bytes (refer to the code snippet)

```
8 bytes and 6 bytes
```

Grade: **50.0%**

5. What datastucture is managing the lifetime of automatic variables?

```
enters and leaves the variable's scope. The data structure is Stack
```

Grade: **100.0%**

1. If I want to use data after the lifetime of the function it was created in, then where should I put it and how do I put it there?

```
make the data static or use malloc to allocate a space on heap
```

Grade: **100.0%**

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

```
Stack is used for static memory allocation and Heap for dynamic memory allocation
```

Grade: **100.0%**

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

```
yes I think there are other kinds of memory e.g. global, shared, text segment
```

Grade: **100.0%**

4. Fill in the blank. In a good C program: "For every malloc there is a ___".

```
call of free()
```

5. Name one reason malloc can fail.

```
The program use up the heap memory
```

Grade: **100.0%**

6. Name some differences between time() and ctime()

```
ctime() is human readable and time() is the variable in machine
```

Grade: **100.0%**

7. What is wrong with this code snippet?

```
double free the same pointer cause undefined behavior
```

Grade: **100.0%**

8. What is wrong with this code snippet?

```
After free the pointer memory, the pointer cannot be used
```

Grade: **100.0%**

9. How can one avoid the previous 2 mistakes?

```
Once we freed a pointer, set it to NULL
```

Grade: **100.0%**

10. Create a struct that represents a Person and typedef, so that "struct Person" can be replaced with a single word. A person should contain the following information: name, age, friends (pointer to an array of pointers to People).

```
struct Person{
    string name;
    int age;
    struct Person* friends;
};
typedef struct Person person_t;
```

Grade: **0.0%**

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

```
int main(){
person_t* ptr1 = (person_t*) malloc(sizeof(person_t));
person_t* ptr2 = (person_t*) malloc(sizeof(person_t));
ptr1->name = "Agent Smith";
ptr2->name = "Sonny Moore";
ptr1->age = 128;
ptr2->age = 156;
ptr1->friends = ptr2;
ptr2->friends = ptr1;
free(ptr1);
free(ptr2);
return 0;
}
```

Grade: **100.0%**

12. 'create()' should take a name and age. The name should be copied onto the heap. Use malloc to reserve sufficient memory for everyone having up to ten friends. Be sure initialize all fields (why?).

```
person_t* create(string name, int age){
        person_t* ptr = (person_t*) malloc(sizeof(person_t));
        ptr->name = strdup(n);
        ptr->age = strdup(a);
        ptr->(*friend) = (person_t*) malloc(10 * sizeof(person_t));

        return ptr;
}
```

Grade: **100.0%**

13. 'destroy()' should free up not only the memory of the person struct, but also free all of its attributes that are stored on the heap. Destroying one person should not destroy any others.

```
void destroy(person_p* p){
    free(p->name);
    free(p->age);
    memset(p, 0, sizeof(person_p));
    free(p);
}
```

Grade: **100.0%**

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

```
putchar(getchar()) gets(buffer); puts(buffer);
```

Grade: **100.0%**

2. Name one issue with gets()

```
overflow
```

Grade: **100.0%**

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

```
int main(){
    char* data = "Hello 5 World";
    char buffer1[20];
    char buffer2[20];
    int score = -1;
    int result = sscanf(data, "%s %d %s", buffer1, &score, buffer2);
    printf("answer: %s: %d: %s\n", buffer1, score, buffer2);
    return 0;
}
```

Grade: **100.0%**

4. What does one need to define before using getline()?

```
define _GNU_SOURCE
```

Grade: **100.0%**

5. Write a C program to print out the content of a file line by line using getline()

```
#define _GNU_SOURCE
#include <stdlib.h>
#include <stdio.h>
```

```
int main() {
    File* fl;
    fl = fopen("filename.txt", "r");
    char* buffer = NULL;
    size_t capacity = 0;
    if(fl==NULL)
        return EXIT_FAILURE;
    ssize_t result;
    while((result = getline(&buffer, &capacity, fl))!= -1 ){
        printf("%s", buffer);
    }
    fclose(fl);
    if(buffer){
          free(buffer);
   }
    return EXIT_SUCCESS;
}
```

Grade: **100.0%**

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

```
-g
```

Grade: **100.0%**

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

```
Make will not adopt the new flag we put into the makefile so the debugging mode
    will not run.
```

Grade: **50.0%**

3. Are tabs or spaces used in Makefiles?

```
Tabs
```

Grade: **100.0%**

4. What does 'svn commit' do? What's a revision number?

```
Revision Numbers. When you create a new Subversion repository, it begins its life
    at revision zero and each successive commit increases the revision number by
    one. After your commit completes, the Subversion client informs you of the new
    revision number: $ svn commit --message "Corrected number of cheese slices."
'svn commit'Send changes from your working copy to the repository
```

Grade: **100.0%**

5. What does ‘svn log‘ show you?

```
svn log shows the log messages for all files and directories inside (and including)
     the current working directory of your working copy.
```

Grade: **100.0%**

Final grade: 0.8837209302%