CS214 Fall 2017 Midterm Exam

Name: _____ **NetID:** _____

Just C things (Pick 10 out of 13 to answer. Circle the problem numbers you want graded):

4 points each

1. What happens in each of the stages of compilation: Preprocessing, compilation, assembly, and linking? 1 pt per correct answer

Preprocessing – search and replace for preprocessor macros (#include,#define, etc)

Compilation – translates code to assembly code

Assembly – translates assembly to machine code

Linking – rearranging and pointing to library code

2. The new version of Linux has a new system call called `groot`. How do you find out what `groot` does, and how to use it? Assume you do not have access to the internet.

man groot

3. What is a system call? When do you use system calls?

A request to the kernel for some managed resource. Use when requesting such a resource (sbrk for memory, fork for process, etc)

4. How are strings represented in C? Create a string called `foo` with the value "cs214". This string must be mutable.

Strings are represented as contiguous characters terminated by a null 0 in memory. (2)

Foo should be a char array or a char *. If the latter, cannot be a literal assignment (immutable). Strcpy allowed. (2)

5. What is the difference between a struct, an enum, and a union? Which of these constructs is best suited to represent the colors of the rainbow? Instantiate such a construct in C called `colors`. The colors of the rainbow are red, orange, yellow, green, blue, indigo, violet.

CS214 Fall 2017 Midterm Exam

Name: _____ NetID: _____

2 pts

Struct – collection of fields
Enum – enumerated types where every value is listed in the type
Union – multiple fields assigned to the same memory location biggest

enum colors { red; orange; yellow; green;blue;indigo;violet} (2 pts)

6.

a) The following code attempts to reimplement strcpy. What is wrong with it? Show an example of how it goes wrong. You may assume initial addresses for dst and src to be 0x5000 and 0x7000 respectively.

```
void strcpy( char * dst, const char * src){
    while(*src){
        dst = src;
        dst++;
        src++;
    }
}
```

The above copies memory addresses. It does not actually copy the contents of the characters. (2 pts)

b) Edit the code above to fix the error. You may do this inline by crossing out and updating the code.

To fix, derefernce on the assignment (2 pts)

7. What is a segmentation fault? Name 2 different causes of segmentation faults.

Segfaults occur when attempting to access invalid memory. (2)

Freeing already freed ptrs, dereferencing null pointers, accessing past valid array indices, accessing past string boundaries, etc(2)

CS214 Fall 2017 Midterm Exam

Name: _____ **NetID:** _____

8. Given this code:

```
unknown * thingy = (unknown*)malloc(4*sizeof(unknown));
int mystery = 0;

mystery = (thingy+1)-thingy;
```

What value does `mystery` hold?

sizeof(unknown) (4 pts)

9. Write a function pointer named "derp" for the following function:

```
int * oddFunction( int * values, struct stuff * storage, char
delimiter)
{...}
```

Int * (*derp)(int *, struct stuff *, char);

10. Why might the following code segfault? Add some code to make sure it returns -1 rather than segfaults.

Malloc can return null (2 pts)

```
int aValue = 12;

int * ptr = (int*)malloc(4*sizeof(int));

if ( !ptr) return -1; (2 pts)

*ptr = aValue;
```

11. What are the differences between strlen and sizeof a string in C? Why? Show an example.

Strlen returns the number of non null characters in the string. Sizeof on a char array returns the number bytes that the char array takes. With a char *, it returns the size of a memory address (2)

CS214 Fall 2017 Midterm Exam

Name: _____ **NetID:** _____

Any valid example of the differences: 2 pts

12. The code below is supposed to increment each value in an int array of length N by 1 and save the new value in a new array. What is actually printed out? Why? Fix the code so that the right thing happens. (Hint: the numbers in someArray are indeed incremented by 1 and stored somewhere)

```
while( i < N )
{
    incrementArray[i] = someArray[i]++;
    printf("%d  %d\n",        incrementArray[i], someArray[i]);
    i++;
}
```

The contents of the original array someArray is printed twice. (2 pts)
To fix: incrementArray[i] = someArray[i]+1; (2 pts)

13. You wish to write a function that encrypts text as numerical values. You know that in C, memory is an amorphous entity. You wish to take every 4 characters in a string, and output the integer equivalent of those 4 bytes. E.g. the string "jack" is encoded as a single integer 1784767339. You may output via printf. You may assume that strlen(str) % 4 == 0. Do NOT make assumptions about the length of a string. Hint: This solution requires fewer than 10 lines of code.

J	a	c	k
01101010	01100001	01100011	01101011
01101010011000010110001101101011			
1784767339			

```
void convert( char * str){


int i;

for( i=0; I < strlen(str) / 4; i+=4){

     printf("%d", *(int *)str[i]);

}
```

CS214 Fall 2017 Midterm Exam

Name: _____ **NetID:** _____

Memory Management (Answer all of these questions) (6 pts each)

1. Fill in the following memory map with the correct labels. Then describe the function, in one sentence, of each part of memory. Possible labels: heap, stack, text, data, bss

0xffffffff

Stack
Heap
Bss
Data
Text

0x00000

3pts for placing labels properly, 3 pts for descriptions

stack is for statically allocated memory/stack frames

heap for dynamic

bss uninitialized global var

data initialized global var

text instructions

2. What is malloc()? Why do C programs tend to have malloc() statements? What does malloc() return?

Malloc is a C library call that dynamically allocates memory by request. Malloc statements are required to have memory that is accessible from all method calls and to allow the programmer to manage the usage of memory as needed. Malloc returns a void pointer.

3. What is wrong with the following function?

```
int * sum(int a, int b){
    int c = a + b;
```

Name: _____ **NetID:** _____

```
    return &c;

}
```

The scope of c does not survive the method call

4

a) Imagine a 32-bit system's implementation of malloc as an implicit list (size + free boundary tags). Given a 4096 byte block of memory to manage, and 100 successful malloc operations within that block (an no free operations), calculate the metadata overhead (e.g. amount of memory for metadata/total amount of memory). Assume size and the free tag are both stored as shorts.

6 pts:

101 x (2 + 2 + 2 + 2)/4096

5 pts:

100 x (2 + 2 + 2 + 2)/4096

3 pts:

101 x (2+2)/4096

2 pts

100 x (2+2)/4096

b) On a 64 bit system, imagine that blocks are implemented in explicit lists (pointer + size + free boundary tags). What is the metadata overhead now? Assume the same 4096 initial block and 100 malloc operations. Assume size and free are stored as shorts.

6 pts

101 x (X)/4096 where X is dependent on their assumptions: 4 vs 8 byte pointers, doubly vs singly linked lists, whether or not they encoded free. Accept any reasonable answer

Take off 1 pt if they multiplied by 100 instead

CS214 Fall 2017 Midterm Exam

Name: _____ **NetID:** _____

5. What is the benefit of an explicit free list in malloc implementations vs. implicit lists via size? What is a drawback of an explicit free list?

(3) Benefit: faster malloc operations: don't have to search through all blocks

(3) Drawback: more metadata usage

6. Presume your malloc implementation never checked for adjacent free blocks (coalesce) when freeing malloced memory. Given enough memory allocations and frees, eventually the code:

```
char * anArray = (char*)malloc(2*sizeof(char));
```

would fail, no matter how much memory was being used. Why?

Fragmentation from metadata

7. A buddy system memory allocator can coalesce adjacent free blocks quickly and reduces metadata overhead. What ways might a buddy system allocator waste more memory than a first fit allocator with block splitting would? Max 4 sentences.

Internal fragmentation (need to use next power of 2 blocksize. 33 byte request requires 64 bytes!

8., What is a memory leak? How does it occur? How does one fix it?

Mallocing without freeing when done. Free all malloc'd requests

Project Redux (Answer all of these questions)

1. One of the issues faced by groups was to handle commas inside a movie title properly. How did you parse the string to ensure that movie titles with commas were parsed correctly?

There must be enough detail here for you to understand that it is a correct solution.

"searched for quotes" is not sufficient. They need to provide a little more.

CS214 Fall 2017 Midterm Exam

Name: _____ **NetID:** _____
4pts

2. Some groups used dynamically allocated arrays to store each movie record. Some groups used statically allocated arrays to store each movie record. Assume Record is a typdef'd struct representing all the fields of a movie record. Also assume sizeof(Record) returns 100.

```
Record arr[5000];
Record * arr2[5000];
```

a) How much memory is allocated for arr? _____5000 * 100____(2pts)_____

b) How much memory is allocated for arr2? _____5000 * 4 (or 8)____(2pts)_____

c) Which of the above declared arrays can data be copied into without further initialization? Why?

a) Because that is a static allocation of structs

(2 pts)

d) Suppose a sorting algorithm requires swap operations. Which of the above structures is more (time) efficient for swapping records? Why?

b) Because no need for deep copies. One can just switch pointers.

Scratch/Additional space: