Name:______**SOLUTIONS**_________
GTID:____________________________

Quiz 2

No Books, No Computers, No Phones, No smart watches, No Calculators, No GBAs! Use your brain, your hand, and a writing instrument!

1. What's the difference between NULL and '\0'? **(10 points)**

'\0' is the char 00000000, ascii 0. NULL is a pointer value from <stddef.h> that specifies that a variable does not point to any address space (though it commonly is implemented *as* a pointer with all bits set to zero)

2. What is the output of the following code at runtime? **(9 points)**

main.c:

```
#include <stdio.h>
#include "mylib.h"
static int time;
int main(){
  for(int i=0; i<3; i++){
    time += timestep();
    printf("time:%d\n", time);
  }
}
```

mylib.c

```
#include "mylib.h"
int time;
int timestep(){
  return time++;
}
```

mylib.h

```
int timestep();
```

time: 0
time: 1
time: 3

The `static` on `int time` in main.c makes that time a completely different time from the one in mylib.c Then you just needed to remember that global variables have static *storage* duration, so they are preallocated with their default value (in this case, zero).

3. Write two different ways to dereference a character pointer named c (to the same character that c points at). **(10 points)**

*c OR c[0]

4. Given the following typedef, write a function to swap two Student instances in memory: **(20 points)**

```
typedef struct {
  unsigned short age;
  char name[128];
} Student;
```

```
void swap(Student *a,
Student *b) {
        Student c = *a;
        *a = *b;
        *b = c;
}
```

5. What is the value of `sizeof("foo");`? **(6 points)**

4 ('f' 'o' 'o' '\0', in the constant string literal storage area)

6. Name two ways structs are not "objects" in the Object Oriented programming sense. **(10 points)**

No encapsulation, no inheritance, no abstraction, no polymorphism, many other differences I won't list all of here.

7. What does the resulting memory look like inside of the given struct, provided it is at memory address 0x1000 (draw boxes for s1 and s2, and say what is in them, to the extent possible)? Be sure to clearly show the sizes of the memory involved! **(15 points)**

```
typedef struct {
  char *s1;
  char s2[12];
} Example;

Example e = { "hello", "world" };
```

S1 is just the size of a pointer, 4 bytes on the GBA (that points to a string literal, "hello" somewhere else in memory)
S2 comes *right after* s1 in memory and has 6 extra bytes of space after 'w' 'o' 'r' 'l' 'd' '\0' that either is 6 more '\0's or whatever random chars were there before `Example e` used the space (depending on static storage duration or dynamic allocation (on the static))

8. Name one way a preprocessor function macro is better than a C function. **(5 points)**

Macros are automatically inline, no call stack
Generally Faster
Easy to mess up

9. Name one way a C function is better than a preprocessor function macro. **(5 points)**

It takes up less space in the compiled project (but could take up more at runtime)
It can be assigned as a function pointer
Less prone to unexpected side effects (at least when passing in x++, etc.) because it evaluated the argument once.
They can be recursive (that could be a weakness if you dislike recursion :D)

10. How is your code / system backed up (or how will you be backing it up very soon)? **(5 points)**

Do it NOW!

11. Favorite Pokémon? **(7 points)**
Squirtle is the Best, but I suppose Pikachu is the most popular.