ME333 Introduction to Mechatronics Name ___Numair Ahmed_____ 2021 Quiz 1: C

Turn in your solutions on Canvas. You can answer digitally in a separate pdf or text document, or scan / photo handwritten notes.

1. The following code will overflow. Explain what that means and rewrite the code to prevent it:

```
unsigned char a = 150; unsigned char b = 125;

unsigned char c = a + b;
```

A char uses 1 byte of memory which means memory used is 2^8 = 256. Therefor a char can store values 0-256.

```
unsigned int a = 150; unsigned int b = 125;

unsigned int c = a + b;
```

2. In a sentence or two, describe the difference in using

```
int numStates = 3;
```

and

```
#define NUMSTATES 3
```

Using int allows numStates to be changed whereas #define will make NUMSTATES accessible throughout the code but cannot be changed. #define declares a constant value.

3. Describe what happens during the compilation process when you build a project with multiple .c files. Running gcc on multiple files generates multiple object .o files. Once generated, gcc will combine the object files into a single executable file to run.

4. Use a `while` loop to calculate and print the next ten prime numbers starting with 2:

```
   int count = 0, n = 0, check_prime = 2, j;

   while(count <10) {
       j = 1; //this will be the diviser

       while(j <= check_prime){ //check_prime is the number we're checking for
prime
           if (check_prime%j == 0 ){
               n++;
               j++;
           }
       }
       if (j == check_prime){
           printf("%d is a prime number\r\n", check_prime);
           count++;
       }
       check_prime++;
   }
```

5. Declare a variable using the following structure and initialize every `sampleValues` to 0 and the `serialNumber` to 15:

```
typedef struct {
 unsigned int sampleValues[5];
 unsigned int serialNumber;
} myData;
int main() {
```

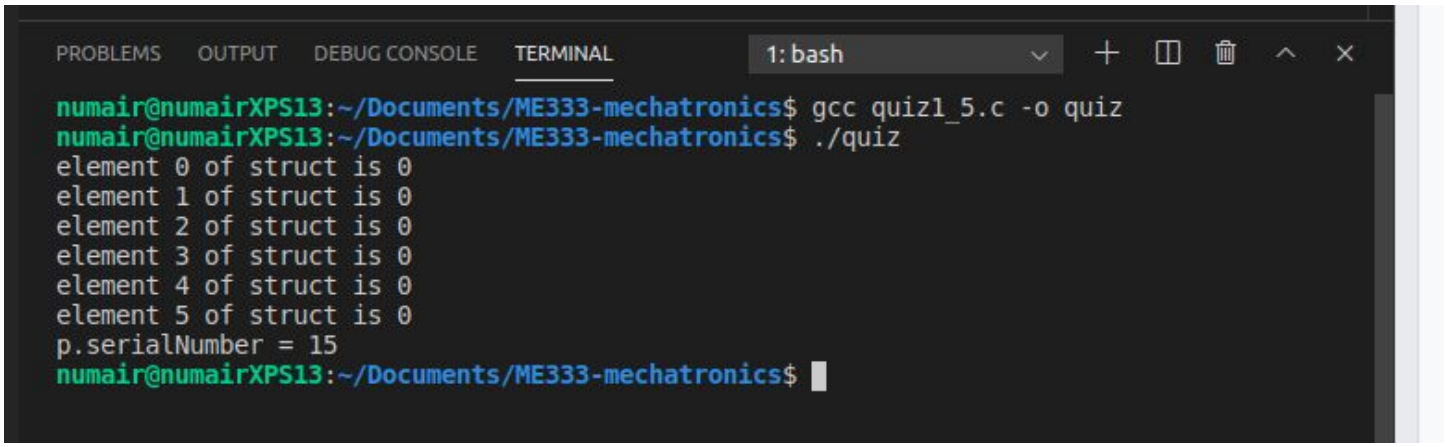
```
#include <stdio.h>
```

```c
typedef struct {
   unsigned int sampleValues[5];
   unsigned int serialNumber;
} myData;

int main() {
   myData p;
   int i;

   for (i = 0; i<= 5; i++){
       p.sampleValues[i] = 0;
       printf("element %d of struct is %d\r\n", i, p.sampleValues[i]);
   }

   p.serialNumber = 15;
   printf("p.serialNumber = %d\r\n", p.serialNumber);
}
```



6. Using your answer to #5, declare a pointer to your variable, and use the pointer to set the value of the `sampleValues` to [15,16,17,18,19]:

```c
#include <stdio.h>

typedef struct {
   unsigned int sampleValues[5];
   unsigned int serialNumber;
} myData;

int main() {
```

```
    myData p;
    myData * pp;
    pp = &p;



    int i;

    pp->sampleValues[0] = 15;
    pp->sampleValues[1] = 16;
    pp->sampleValues[2] = 17;
    pp->sampleValues[3] = 18;
    pp->sampleValues[4] = 19;

    for (i = 0; i<= 4; i++){

        printf("element %d of struct is %d\r\n", i, pp->sampleValues[i]);
    }

    p.serialNumber = 15;
    printf("p.serialNumber = %d\r\n", p.serialNumber);
}
```

7. Write two functions, one that uses type `myData` by value and one that uses `myData` by reference. In the function, ask the user to enter a new `serialNumber` and store it in your variable, then return to `main`:

8. Write a complete program (here on the page, you don't have to write and test the code with gcc) that asks the user to enter a string and a number, and then prints the string back with each character shifted over in the ASCII table by the number they entered. For example, if the user enters "ABCD 4", the result is "EFGH". Limit the user to entering a shift value between 1 and 8, keep asking for a shift value until it is in the correct range.

```
#include <stdio.h>

int main(void) {
    float num;
    char str;
    int i;

    printf("enter a string and a number: \r\n");
```

```
    scanf("%s %d", &str, &num);

    if (num >= 1 && num <= 4)
    {
        /* code */
        printf("%s:  %c\n", str, str + 4);
    }
    else
    {
        printf("please enter a number between 1 and 4\r\n");
    }
}
```