ENEE 140, Fall 2018
Final Exam

Date:

University of Maryland Honor Pledge: The University is committed to Academic Integrity, and has a nationally recognized Honor Code, administered by the Student Honor Council. In an effort to affirm a community of trust, the Student Honor Council proposed and the University Senate approved an Honor Pledge. The University of Maryland Honor Pledge Reads:

"*I pledge on my honor that I have not given or received any unauthorized assistance on this examination (or assignment)*"

Please write the exact wording of the Pledge, followed by your signature, in the space below:

Pledge: _____
Pledge: _____
Pledge: _____
Pledge: _____

Your signature: _____
Full name: _____ Course: _____ Directory ID: _____

List of Exam Questions:

Question:	1	2	3	4	5	6	7	8	Total
Points:	10	6	15	8	16	16	15	14	100
Score:									

Instructions:

- Make sure that your exam is not missing any sheets, then write your full name, your section and your Directory ID on the front.
- Write your answers in the space provided below the problem. If you make a mess, clearly indicate your final answer.
- The exam has a maximum score of 100 points.

- The problems are of varying difficulty. The point value of each problem is indicated. Pile up the easy points quickly and then come back to the harder problems.
- This exam is OPEN BOOK. You may use any books or notes you like. Calculators are allowed, but no other electronic devices. Good luck!

1. (10 points) This problem tests your understanding of C types and casts and of C operators. Assume that variables `a`, `b`, `c` and `d` are defined as follows:

```
int      a = -2;
float    b = -1;
unsigned c = 1;
double   d = 2;
```

Fill in all the empty cells in the table below. For each of the C assignment expressions in the left column, state the resulting value of the `r2`–`r9` variables. If an expression results in a compilation error, write ERROR.

Assignment		Value
double	r0 = a / 3;	0
double	r1 = (c + d) / 2;	
double	r2 = c / a;	
unsigned	r3 = UINT_MAX + c;	
int	r4 = (INT_MAX + INT_MIN) == 0;	
int	r5 = (a + c) < 0;	
double	r6 = a ? b : d;	
unsigned	r7 = c % 9;	
unsigned	r8 = c % 9.0;	
unsigned	r9 = c % '9';	
int	r10 = c | (**unsigned**)d;	

2. (6 points) This question tests your understanding of if-statement conditions. Recall that `'1'` represents the logical true and `'0'` is the logical false. Given this and the following code, write the output that the program will print.

```
int main () {
        int A = 0;
        int B = 0;

        if (A || B)
                printf("P1, ");
        if (!A || B)
                printf("P2, ");
        if (A && B)
                printf("P3, ");
        if (!(A && B))
                printf("P4, ");
        if (!A || !B)
                printf("P5, ");
        if (!A)
                printf("P6, ");
        return 0;
}
```

3. (15 points) This problem tests your understanding of C control flow and logical operations. Consider a function `three_way_compare()` that takes 3 integers as arguments and that returns:

   1 if the arguments are provided in a strictly increasing order

   0 if they are all equal

   -1 if the arguments are provided in a strictly decreasing order

   -2 otherwise

   Strictly increasing/decreasing means that no two arguments are equal. For example, `three_way_compare(1,2,3)` should return 1, but `three_way_compare(1,2,2)` should return -2. Fill in the blanks below to implement this functionality.

```
int
three_way_compare (int a, int b, int c)
{
        int result;

        if (a < b) {
                if (b < c) {
                        result = 1;
                } else {
                        result = _____;
                }


        _____

                result = (a > b || a == b) - _____;

                if (a != b) {
                        result = (b > c) - 2;
                } _____ if (!(b < c _____ b > c)) {
                        result += 2;
                }
        }

        return result;
}
```

4. (8 points) This question tests your understanding of random number generation. Given the following ranges of random numbers, write in a single line a statement that will generate random numbers within this range. You may assume that the random number generated was properly seeded and you may use `RAND_MAX` as the maximum possible number generated.

   (a) Integers 0 to 10, including 0 and 10.

   (b) Integers 5 to 15, including 5 and 15.

   (c) Random floats from 0.00 to 1.00, out to two decimal places, including 0.0 but not 1.0.

   (d) Random integer multiples of 5 in the range of 0 to 100, including 0 and 100.

5. (16 points) This question tests your knowledge of file I/O and of multi-dimensional arrays.

   The program below initializes a matrix `A` (by invoking a function called `initialize_matrix`) and prints its contents to a user-specified file. The program is compiled to `print_matrix.out`, which is executed as follows: `./print_matrix.out filename.txt` The matrix should be printed into the file row-by-row, which each row separated by a new line. The output file should be closed upon completion of the program. Fill in the blanks in the following main function to ensure that the application executes correctly.

```
int main(int argc, char *argv[]) {
  int A[SIZE][SIZE], i, j;
  FILE *file;
  file = fopen(_____, _____);

  if (file == _____) {
    printf("File does not exist.\n")
    _____(file);
    return 0;
  }

  initialize_matrix(A);

  for (i = 0; i < SIZE; i++) {
    for (j = 0; j < SIZE; j++) {
      fprintf(_____, "%3d", _____);
    }
    fprintf(file, _____);
  }

  _____(file);

  return 0;
}
```

6. (16 points) This problem tests your knowledge of multi-dimensional arrays.

   Assume you have a 2D array `a[4][4]` filled with `0`'s. The following code fragment will add values to a then print the array. What is the output?

```
int i,j;
    for(i = 0; i < 4; i ++){
        for(j = 0; j <= i; j ++){
            a[j][i] = j + i;
        }
    }

    for(i = 0; i <4; i ++){
        for(j = 0; j < 4; j ++){
            printf("%d ",a[i][j]);
        }
        printf("\n");
    }
```

7. (15 points) Using your knowledge of arrays, characters, and strings, what is the output of the following code?

```
#include<stdio.h>

int main(){
    int i;
    char a[] = {'f','i','n','a','l','e','x','a','m','\n','\0'};

    for(i = 1; i < 7; i ++){
        if(a[i]>'A' && a[i] < 'Z'){
            printf("H");
        }
        if(a[i]>'a' ){
            printf("L");
        }
        if(a[i] > 'j'){
            printf("M");
        }
        putchar(a[i]++);
        printf("\n");
    }
printf(a);
}
```

8. (14 points) This problem tests your understanding of structs and function calls. What is the output of this program?

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


typedef struct x{
  int num;
  float num2;
}Sol;

int add(Sol x,Sol y);
float add2(Sol x,Sol y);

int main(){

  Sol a;
  a.num=0;
  a.num2=0;
  Sol b;
  b.num=0;
  b.num2=0;
  int control=8;
  while(control!=0){
   a.num= add(a,b);
   b.num2= add2(a,b);
    control=control>>1;
  }
  printf("%d\n%.1f\n",a.num,b.num2);

  return 0;
}

int add(Sol x,Sol y){
  x.num+=3;
  y.num+=2;
  return x.num+y.num;
}

float add2(Sol x,Sol y){
  x.num2+=.5;
  y.num2+=1;
  return x.num2+y.num2;
}
```