ENEE 140, Spring 2026
Midterm 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:	16	15	8	10	15	18	18	12	112
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 name and section at the bottom of each page as well.
- Write your answers in the space provided below the problem. If you make a mess, clearly indicate your final answer.

- The last question is for EXTRA CREDIT and is worth 12 points. You may receive full credit without answering it, assuming that you answered correctly all the other questions (the exam will be scored out 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. No electronic devices (e.g. laptops, tablets, smartphones, calculators) are allowed. Good luck!

1. (16 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:

```
float a = 2;
int b = 4;
char c = 'E';
unsigned d = 5;
```

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 `v2`–`v9` variables. If an expression results in a compilation error, write ERROR.

Assignment		Value
float	`q1 = d / a;`	2.5
int	`q2 = d / b;`	
float	`q3 = a % d;`	
float	`q4 = d / (float) b;`	
char	`q5 = c - a;`	
int	`q6 = b % d;`	
unsigned	`q7 = UINT_MAX + b;`	
float	`q8 = (int) a * d;`	
char	`q9 = c + ++b;`	

2. (15 points) This question test your understanding of various concepts in the class. Fill in the blanks with the most appropriate answer.
   (a) _____ is a special value that indicates input has ended.
   (b) To write a function that does not return anything, we use the _____ keyword.
   (c) The result of dividing two integers is a/an _____.
   (d) When less space is allocated to a float by a printf format specifier than there are digits after the decimal point, the value will be _____.
   (e) Characters in C are stored as _____.

3. (8 points) This is a four-part multiple choice, each worth roughly 5 points per part. These questions tests functions in C.

a. How is a function declared in C?

A. `returnType(parameters) functionName;`

B. `functionName(parameters) returnType;`

C. `functionName: returnType(parameters)`

D. `returnType functionName(parameters);`

3. _____

b. What is the purpose of a function prototype in C?

A. To call a function

B. To create a reference

C. To declare it before defining

D. To write the body of the function

3. _____

c. Which of the following function calls have no **return** statement in C?

A. **void**

B. **int**

C. **char**

D. **float**

3. _____

d. Is **int** main() a function?

A. No

B. Yes

C. Maybe

3. _____

4. (10 points) This problem has two parts and will test your understanding of integer overflow, `for` loops, and `if` statements. Provided below is a program that tests when integer overflow occurs as the values are incremented by one.

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

int main() {
    int x = INT_MIN + 2*INT_MAX;
    int previous_x = x;
    unsigned int y = UINT_MAX - 2;
    unsigned int previous_y = y;

    for (int i = 0; i < 3; i++) {
        printf("Before increment: x = %d\n", x);
        x++;
        printf("After increment:  x = %d\n", x);

        printf("Before increment: y = %u\n", y);
        y++;
        printf("After increment:  y = %u\n", y);

        if (x < previous_x) {
            printf("Signed overflow occurred at i = %d\n", i);
        }
        if (y < previous_y) {
            printf("Unsigned overflow occurred at i = %d\n", i);
        }

        previous_x = x;
        previous_y = y;
        printf("\n");
    }

    return 0;
}
```

(a) At what value `i` does x first reach integer overflow?

A. `0`

B. `1`

C. `2`

D. `x does not reach integer overflow`

(a) _____

(b) At what value `i` does y first reach integer overflow?

A. `0`

B. `1`

C. `2`

D. `y does not reach integer overflow`

(b) _____

5. (15 points) This question tests your understanding of loops and character I/O. The program below should convert the uppercase alphabet (A-Z) to lowercase and print the result for each letter on a new line. Fill in the blanks to achieve this functionality. Your answers should not contain any numbers.

```
#include <stdio.h>

int main(void) {

    char c;
    for (_____; _____; _____) {
        printf("_____\n", _____);
    }
    return 0;
}
```

6. (18 points) This problem tests your understanding of `if-else` statements. What is the output of the following function?

```
#include <stdio.h>

int main() {
    int a = 2, b = 5, c = 4, d = 3, e = 6, f = 1;

    if (a % 2 == 0) {
        b = a + 4;
        c = b * 3;
    } else {
        d = c + 5;
        e = d - 1;
    }

    if (b % 3 == 0) {
        f = c - d;
        a = e + 3;
    } else {
        c = a - b;
        d = f + 2;
    }

    e = a + b + c;
    f = d + e;

    printf("a is %d, b is %d, c is %d, d is %d, e is %d, f is %d\n",
    a, b, c, d, e, f);

    return 0;
}
```

7. (18 points) This question tests your ability to debug code. The following program should print the following characters: A, B, C, D, E, F, G, H, I, and J. Circle and correct the lines that contain bugs. (Hint: There are 6 lines)

```
1  #include stdio.h
2
3  char get_letter (int i);
4
5  int main () {
6    int i;
7    float a;
8
9    for (i=1;i < 10; i++) {
10     a = get_letter(i);
11
12     printf("%d\n", a);
13   }
14   return 0;
15 }
16
17 void get_letter (int i) {
18   return A + i;
19 }
```

8. (12 points) (BONUS) This question tests your understanding of C variables and operations. Your goal is track the variable known as GAMING as it progresses through the code. Note the changes to its value, and write down what each of the `printf` statements will print out. Don't lose it, keep watch of it!

```
1  #include <stdio.h>
2
3  int minecraft(int a);
4  int gta(int b);
5  void counterstrike(int c);
6
7  int main() {
8      int GAMING = 408;
9      printf("GAMING = %d\n", GAMING);
10
11     GAMING = minecraft(GAMING);
12     printf("GAMING = %d\n", GAMING);
13
14     GAMING = GAMING + (4 / 2);
15     printf("GAMING = %d\n", GAMING);
16
17     counterstrike(GAMING);
18     printf("GAMING = %d\n", GAMING);
19
20     GAMING = GAMING + (7 * 3) + 6;
21     printf("GAMING = %d\n", GAMING);
22
23     GAMING = gta(GAMING);
24     printf("GAMING = %d\n", GAMING + 3);
25
26     return 0;
27 }
28
29 int minecraft(int a) {
30     int x = 9;
31     int y = 6;
32     int z = 0;
33
34     for (int i = 0; i < 3; i++) {
35         z += (x - y) / 3;
36     }
37
38     return a + z;
39 }
40
41 void counterstrike(int c) {
42     int temp = c % 5;
43     int shadow = temp * 2;
44
45     if ((shadow - temp) % 2 == 0) {
46         shadow += 1;
```

```
47      }
48  }
49
50  int gta(int b) {
51      int bonus = 0;
52
53      for (int i = 1; i <= 3; i++) {
54          bonus += i * 4;
55      }
56
57      return b + bonus;
58  }
```