ENEE 140, Spring 2024
Midterm Exam — Answer Key
Do Not Make a Copy!!

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	4	9	10	19	24	18	10	110
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 10 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:

```
char        a = '0';
int         b = -1;
unsigned    c = 1;
float       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 `r1`–`r8` variables. If an expression results in a compilation error, write ERROR.

Assignment	**Value**
float r0 = d / 4;	-0.5
int r1 = c / 4;	**0**
float r2 = b / 4;	**0.0**
char r3 = a − b;	**'1'**
unsigned r4 = UINT_MAX / b;	**1**
int r5 = (c−d) / 2;	**1**
float r6 = −d % 2;	**ERROR**
unsigned r7 = d / ++c;	**UINT_MAX**
unsigned r8 = '9' − a + INT_MAX % 2;	**10**

2. (4 points) This question tests your understanding of `for` loops. How many times will the following loop execute?

```
for (i = 0; i <= 10; i = i-1) {}
```

A. 100 times

B. Forever

C. 11 times

D. It doesn't execute at all

2. **B**

3. (9 points) This problem has three parts and will test your understanding of `while` and `for` loops, as well as binary–decimal conversions. Provided below is a program utilizing a `for` loop that converts a base 10 number into a base 2 number.

```
#include <stdio.h>

int main() {
    unsigned numberBaseTen = 0;

    printf("Please enter your number: ");
    scanf("%u", &numberBaseTen);

    for (; numberBaseTen > 0; numberBaseTen = numberBaseTen / 2) {
        int remainder = numberBaseTen % 2;

        printf("%d", remainder);
    }
}
```

(a) Please convert the `for` loop into a `while` loop without changing its functionality. You do not need to rewrite the entire program - the while loop is sufficient.

Solution:

```
while (numberBaseTen > 0) {
    int remainder = numberBaseTen % 2;

    printf("%d", remainder);
    numberBaseTen = numberBaseTen / 2;
}
```

(b) If the user inputs `26` after being prompted to provide a number, what will the output of the `for` loop be (it will be a binary sequence)? `NOTE:` The output of the for loop will be on the same line (printf does not have a newline character).

Solution:

```
For Loop Output:    01011
```

(c) Now, reverse the order of the output you found. For example, if the sequence of numbers that was printed out was "1101001", then reversing it would result in "1001011". What is this number in base 10?

Solution:

```
Final Number:       11010 (Base 2) -> 26 (Base 10)
```

4. (10 points) This problem tests your understanding of `if-else` statements. What is the output of the following program? Write your answer on the next page.

```
#include <stdio.h>
int main() {
    int a = 0, b = 0, c = 0, d = 0, e = 0;
    if (b == 1) {
        b = 3;
        e = 5;
    }
    if (a == 0) {
        b = 2;
        c = 0;
        if (d == 1) {
            a = 5;
        }
        d = 4;
    } else {
        a = 7;
        b = 9;
    }
    if (b == 2) {
        if(c == 1) {
            d = 2;
        } else {
            d = 3;
        }
    } else {
        if(d == 1) {
            a = 4;
        }
        c = 3;
    }
    if (d == 0) {
        a = 4;
        b = 6;
        if (c == 5) {
            d = 7;
            e = 8;
        }
    } else {
        e = 7;
        c = 6;
        a = 4;
    }
    printf("a is %d, b is %d, c is %d, d is %d, e is %d", a,b,c,d,e);
}
```

Solution: a is 4, b is 2, c is 6, d is 3, e is 7

5. (19 points) This question will test your knowledge of logical statements, `getchar()`, and `char` conversions. Determine the output of the program given the following input and print it on the given lines below. Assume at that there is a EOF character at the end of the input. *Note: the '_' on line 22 refers to an underscore character, NOT a space.*

   Input:

   !uBBn_#_tBsQRdo_#qt

   Program:

```
1
2      #include <stdio.h>
3
4      int main() {
5          int c;
6          while ((c = getchar()) != EOF) {
7              if (c == '#') {
8
9                  putchar('X');
10
11             }
12             else if (c >= 'A' && c <= 'Z') {
13
14                 putchar(c + ('D' - 'A'));
15
16             }
17             else if (c >= 'a' && c <= 'z') {
18
19                 putchar(c + ('A' - 'a'));
20
21             }
22             else if (c == '_') {
23
24                 putchar(c);
25
26             }
27
28             else {
29
30                 putchar('Q');
31
32             }
33     }
```

Solution:

QUEEN_X_TESTUDO_XQT

6. (24 points) This question will test your understanding of functions and arithmetic operations.

   The `diff_of_factors` function below takes in an integer number `n` (assume it is **positive, non-prime, and greater than 1**) and uses the remainder in a `for loop` to find all of `n`'s factors (not including `n` itself). Once it finds a factor, it checks to see if that factor is even or odd. All of the even factors are summed together in `even_sum` and all the odd factors (including 1) are summed together in `odd_sum`. The function then returns the difference between `even_sum` and `odd_sum`. Fill in the blanks to make `diff_of_factors` work properly.

```
      int      diff_of_factors(      int      n) {

    int i, even_sum =       0      , odd_sum =       0      ;

    for (     i = 1     ;       i       < n;      i++      ) {

        if ((       n       % i) == 0) {

            if ((i %       2       ) == 0) {

                even_sum += i;

            }

            else {

                odd_sum += i;

            }
        }
    }

return even_sum - odd_sum;

}
```

Solution:

```
    int diff_of_factors(int n) {
```

```
    int i, even_sum = 0, odd_sum = 0;

    for (i = 1; i < n; i++) {

        if ((n % i) == 0) {

            if ((i % 2) == 0) {

                even_sum += i; /* or even_sum = even_sum + i; */

            }

            else {

                odd_sum += i;

            }
        }
    }

return even_sum - odd_sum;

}
```

7. (18 points) This question will test your ability to debug code. The following program should declare an integer array of **size 7**. It should then store the integers 64, 32, 16, 8, 4, 2, 1, in the array (**in the order listed**). Finally, the program should print out the **first two elements** in the array. Circle the lines that contain bugs and correct them on the given lines/space below the program. *Note: there are no errors involving the mathematical equation.* (Hint: There are **6 lines with errors**)

```
1
2     #include stdio.h
3     #include <math.h>
4
5     int main() {
6         int i;
7         int array[];
8
9         for (i = 0; i <= 7; i++) {
10            pow(2, (6 - i)) = array[i];
11        }
12
13        printf("First␣two␣elements␣are␣%d␣and␣%d"; array[1], array[2]);
14
15        return 0
16    }
```

Solution:

```
Line 2: #include <stdio.h>

Line 7: int array[7]

Line 9: for (i=0; i < 7; i++) {

Line 10: array[i] = pow(2, (6 - i));


Line 13: printf("First two elements are %d and %d", array[0], array[1]);

Line 15: return 0;
```

8. (10 points) EXTRA CREDIT: This question tests your understanding of C variables and operations. In the program below, the variable CLONEWARS will be having the ride of its life, jumping into many different operations. Write down the output of the program. To do this, you should track the progress of CLONEWARS across the program, keep in mind all changes to its value, and write down what what each of the printf statements will print out.

```c
#include <stdio.h>

void REPUBLIC (int x);
int SEPARATIST (int y);

int main() {
    int CLONEWARS = 0;
    int i;
    for (i=0; i<=7; i++) {
        CLONEWARS += i*i;
    }
    printf("CLONEWARS = %d\n", CLONEWARS);

    REPUBLIC(CLONEWARS);
    printf("CLONEWARS = %d\n", CLONEWARS);

    CLONEWARS += ((10/4.0)*5.0)-2.0;
    printf("CLONEWARS = %d\n", CLONEWARS);

    CLONEWARS += 0x32;
    printf("CLONEWARS = %d\n", CLONEWARS);

    CLONEWARS = SEPARATIST(CLONEWARS);
    printf("CLONEWARS = %d\n", CLONEWARS);

    return 0;
}

void REPUBLIC (int x) {
    int venator = 0;
    while (venator < 10) {
        x = x - venator;
        venator += 1;
    }
    if (x % 2 == 0) {
        x += 25;
    } else {
        x +=4;
    }
}

```

```c
int SEPARATIST (int y) {
    int lucrehulk;
    for (lucrehulk = 1; lucrehulk < 20; lucrehulk++) {
        y = y*2/lucrehulk;
    }
    if (y/2 == 0) {
        y = 205;
    } else {
        y = 200;
    }
    return y;
}
```

Solution:

```
CLONEWARS = 140
CLONEWARS = 140
CLONEWARS = 150
CLONEWARS = 200
CLONEWARS = 205
```