ENEE 140, Spring 2025
Final 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	9	Total
Points:	16	8	10	16	10	18	10	12	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:

```
int      a = 10;
unsigned b = 255;
char     c = 'A';
float    d = 3.14;
```

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. For float, assume 2 decimal points.

If an expression results in a compilation error, write ERROR.

Assignment		Value
float	r0 = d * 2;	6.28
int	r1 = b / 5;	**51**
char	r2 = c + 'e' − 'B';	**'d'**
float	r3 = d % 2;	**ERROR**
unsigned	r4 = (**unsigned**)((b − b) − 1);	**UINT_MAX**
float	r5 = (**int**)(d + 2) + a;	**15.0**
float	r6 = d − (**int**)(d + 1);	**-0.86**
unsigned	r7 = (a > −−a) ? a + 2 : a − 2;	**11**
int	r8 = b != b−−;	**0**

2. (8 points) This is a four-part multiple choice, each worth roughly 2 points per part. These questions test various concepts of C. Some questions may have more than one answer.

   (a) In C, which of the following statements execute their body more than once?

   A. **for**

   B. **if**

   C. **do–while**

   D. **while**

(a) **A,C,D**

Solution: A,C,D

(b) In C, scanf() and printf() belong to which library?

A. <stdlib.h>

B. <string.h>

C. <stdio.h>

D. <ctype.h>

(b) **C**

Solution: C

(c) In C, does the % operator work with variables of type **double**?

A. No

B. Yes

(c) **A**

Solution: A

(d) In selection sort, what does the algorithm do in each iteration of the outer loop?

A. Compares adjacent elements and swaps them **if** they are out of order

B. Locates smallest element and swaps it with the element at the current index

C. Divides the array into smaller subarrays and merges them back

D. Recursively sorts smaller portions of the array

(d) **B**

Solution: B

3. (10 points) This problem will test your understanding of control flow. The intent of this snippet of code is to determine which combination of coins was entered. You can assume this code will only be reached when one whole coin is entered, represented by the integer variable coin.

Convert this snippet of code from if/else-if statements into a switch statement. Write your solution in the space on the next page.

```
if (coin == 5){
    printf("A nickel was entered.");
} else if (coin == 10){
    printf("A dime was entered.");
} else if (coin == 25){
    printf("A quarter was entered.");
} else if (coin == 1){
    printf("A penny was entered.")
} else {
    printf("What coin is that?");
}
```

Solution:

```
switch(coin){
    case 1: printf("A penny was entered."); break;
    case 5: printf("A nickel was entered."); break;
    case 10: printf("A dime was entered."); break;
    case 25: printf("A quarter was entered."); break;
    default: printf("What coin is that?"); break;
}
```

4. (16 points) This question tests your understanding of `arrays` of `structs`. The following is the definition of a `struct rectangle` with integer components `length` and `width`.

```
typedef struct _rectangle{
    int length;
    int width;
} Rectangle;
```

The `average_area` function takes an `array of Rectangles` and the size of the array, computes the area of each rectangle, adds it to a sum of areas dynamically, and then returns the `floating-point` average of those areas. Fill in the blanks to make `average_area` work properly.

```
float average_area(Rectangle rects[], int size) {

    int sum = 0, i, area;

    for ( i = 0 ; i < size ; i++ ) {
        area = rects[i].length * rects[i].width;
```

```
        sum  += area  ;
    }

    return (float) sum / size;
}
```

5. (10 points) This problem tests your understanding of structs, file I/O, and your ability to debug code. Emma wants to find one more course for next semester with a rating of at least 4.5. She needs the course to be at least three credits to meet the minimum requirements for her scholarship. Help her debug this code to sort through a database of up to 100 courses. Put your lines in order starting from the beginning (on the next page). You can expect input from a file titled coursedatabase.txt to be of the format:

```
ENEE140 2 5.0
```

```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  #define MAX_STRING 100
5  #define MAX_COURSE 100
6
7  struct course {
8      char code[MAX_STRING];
9      int credits;
10     float rating;
11 };
12
13 int main() {
14     int i = 0;
15     struct course course_directory[MAX_COURSE];
16
17     FILE *file;
18     file = fopen("coursedatabase.txt", "r");
19
20     if (file = NULL){
21         printf("Could not open file.");
22         exit(-1);
23     }
24
25     while (fscanf(file,"%s %d %f",course_directory[i].code,
26         course_directory[i].credits,
27         &course_directory[i].rating) == 3 && i < MAX_COURSE) {
28         i++;
29     }
30     //Display 3 and 4 credit courses with a rating over 4.5
31     for (i = 0; i < MAX_COURSE; i++){
32         if (course_directory[i].rating >= 4.5 || course_directory[i].credits > 3){
```

```
33                printf("Course %s has a %.2f rating",
34                course_directory[i].code,course_directory[i].rating);
35            }
36        }
37        fclose(FILE);
38        return 0;
39    }
```

Line '_____: Bug is _____.

Correct line is _____;

Line '_____: Bug is '_____

Correct line is '_____;

Line '_____: Bug is '_____

Correct line is '_____;

Line '_____: Bug is '_____

Correct line is '_____;

Solution:

Line 20: Bug is = instead of ==.
Correct line is `if (file == NULL){`

Line 26: Bug is no & `for` an integer with fscanf

Correct line is &course_directory[i].credits,

Line 32: Bug is || instead of &&

Correct line is `if (course_directory[i].rating >= 4.5 ||`
`course_directory[i].credits > 3){`

Line 37: Bug is FILE instead of file

Correct line is fclose(file);

6. (18 points) This question tests your understanding of `arrays`, `for` loops, and the `%` and `?:` operators. What is the output of the following program?

```
1 #include <stdio.h>
2
```

```
3  int main() {
4
5      int i, a[9];
6
7      a[0] = 3;
8      for (i = 1; i < 9; i++) {
9          a[i] = a[i-1] * 2 - 2;
10     }
11
12     for (i = 0; i < 9; i++) {
13         a[i] = (a[i] % 3 == 0) ? a[i] + 1 : a[i] - 1;
14     }
15
16     i = 0;
17     printf("%d-", a[i]);
18     for (i = 2; i != 0; i = (i + 2) % 9) {
19         printf("%d-", a[i]);
20     }
21
22     return 0;
23 }
```

Solution: 4 7 19 67 259 3 9 33 129

7. (10 points) This question tests your ability to debug. The following program opens two files using command-line arguments, and writes the contents of file 1 into file 2. There are five errors in the code that prevent it from running properly. On the next page, list the lines where the errors occur and correct them.

```
1  #include stdio.h
2
3  #define MAXLINE 100
4
5  int main(int argc, char* argv) {
6
7      if (argc == 3) {
8          printf("Error: Wrong number of arguments\n");
9          return 1;
10     }
11
12     FILE* fp1;
13     FILE* fp2;
14     char line[MAXLINE];
15
16     fp1 = fopen(argv[1], "w");
17
18     if (fp1 == NULL) {
19         printf("File error\n");
20         return 1;
21     }
22
23     fp2 = fopen(argv[2], "w");
24
25     if (fp2 == NULL) {
26         printf("File error\n");
27         return 1;
28     }
29
30     while (fgets(line, MAXLINE, fp1) != NULL) {
31         fputs(line, stdout);
32     }
33
34     fclose(fp1);
35     fclose(fp2);
36
37     return 0;
38 }
```

Solution:

```
Line 1: #include <stdio.h>
Line 5: int main(int argc, char* argv[])
Line 7: if (argc != 3)
Line 16: fp1 = fopen(argv[1], "r");
Line 31: fputs(line, fp2);
```

8. (12 points) This code tests your understanding of variable scope. What is the value of a at each printf statement? Write your answer on the next page.

```c
#include <stdio.h>
int foo(int a);
void foo2(int arr[5]);
int main(void) {
    int arr[5] = {1,2,3,4,5};
    int a = arr[1];

    printf("a is:\n %d\n", a);

    a += 3.14;
    printf(" %d\n", a);

    foo(a);
    printf(" %d\n", a);

    a = foo(a);
    printf(" %d\n", a);

    a = a & 3;
    printf(" %d\n", a);

    foo2(arr);
    a = arr[2];
    printf(" %d\n", a);

    return 0;
}
int foo(int a){
    int b = 317;
    a = a % 20;
    return b;
}
void foo2(int arr[5]){
    int i;

    for (i = 0; i < 5; i++){
        if (i % 2 == 0){
            arr[i]++;
        }
    }
}
```

Solution:

```
a is:
 2
 5
 5
 317
 1
 4
```

9. (12 points) (BONUS) This question tests your understanding of C variables and operations. Your goal is to track the variable known as POLANDIUM 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 earth(int a);
4  void climate(int c);
5  int siberia(int b);
6
7  int main() {
8      int POLANDIUM = 304;
9      printf("POLANDIUM = %d\n", POLANDIUM);
10
11     POLANDIUM = earth(POLANDIUM);
12     printf("POLANDIUM = %d\n", POLANDIUM);
13
14     POLANDIUM += 8 * 1;
15     printf("POLANDIUM = %d\n", POLANDIUM);
16
17     climate(POLANDIUM);
18     printf("POLANDIUM = %d\n", POLANDIUM);
19
20     POLANDIUM += ((2 & 3) / 1) + (0 & 7);
21     printf("POLANDIUM = %d\n", POLANDIUM);
22
23     POLANDIUM = siberia(POLANDIUM);
24     printf("POLANDIUM = %d\n", POLANDIUM);
25
26     return 0;
27 }
28
29 int earth(int a) {
30     int continent = 0;
31     for (int ocean = 0; ocean < 3; ocean++) {
32         continent += (ocean + 1);
33     }
34     if (continent == 6) {
35         return a + 1;
36     } else {
37         return a + 2;
38     }
39 }
40
41 void climate(int c) {
42     int system = 1;
```

```
43      int elevation = c - 3;
44
45      do {
46          system += elevation + system;
47          elevation = (system % 20) + 1;
48      } while (system % elevation != 5 && elevation != 9);
49  }
50
51  int siberia(int b) {
52      int freeze = 2;
53      int wind = 0;
54
55      for (; freeze > 0; freeze--) {
56          wind += (freeze * 1);
57      }
58
59      if (wind == 3) {
60          return 140;
61      } else {
62          return 205;
63      }
64  }
```

Solution:

```
POLANDIUM = 304
POLANDIUM = 305
POLANDIUM = 313
POLANDIUM = 313
POLANDIUM = 315
POLANDIUM = 140
```