ENEE 140, Fall 2016
Final Exam

Date: Thursday, December 15, 2016, 8–10 am

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:	15	9	10	9	10	15	7	15	10	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. (15 points) This problem tests your understanding of C types and casts and of C operators. Assume that variables `a`, `b`, `c`, `d` and `e` are defined as follows:

```
int         a = -2;
unsigned    b = -1;
unsigned    c = 0;
float       d = 1;
float       e = 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`–`r11` variables. If an expression results in a compilation error, write ERROR.

Assignment		**Value**
float	r1 = d / e;	0.5
int	r2 = d / e;	______________
int	r3 = a / (**int**)b;	______________
int	r4 = (c−1) < 0;	______________
unsigned	r5 = b % 2;	______________
unsigned	r6 = d % 2;	______________
int	r7 = (FLT_MAX + 1) > FLT_MAX;	______________
float	r8 = d / a;	______________
float	r9 = ((**float**)a / 4) + (**float**)(a / 4);	______________
unsigned	r10 = (−(a+b) & 2) >> 1;	______________
unsigned	r11 = (−(a+b) && 2) >> 1;	______________

2. (9 points) The following question tests your knowledge of integer operations. Please write out the output of the program.

```
#include <stdio.h>

int main()
{
  int x = 0;
  while( ++x <= 13)
    printf("%d␣",x++);
  x = x << 1;
  printf("\n%d\n",x);
  printf("%x\n",x);
  return 0;
}
```

3. (10 points) This question tests your understanding of loops and characters. The program below reads an uppercase character and prints a half pyramid with the characters from `'A'` to the character provided by the user. For example, if the character provided is `'E'`, the output should be:

```
A
B B
C C C
D D D D
E E E E E
```

You may assume that the user will input an uppercase character.

```
#include <stdio.h>

int main()
{
    int i, j;
    char input, alphabet = 'A';

    printf("Enter␣the␣uppercase␣character␣you␣want␣to␣print␣in␣last␣row:␣");
    scanf("%c",&input);

    for(i=1; i <= ____________; ++i) {
        for(____________; ____________; ++j) {
            printf(____________, alphabet);
        }
        ____________;

        printf("\n");
    }

    return 0;
}
```

4. (9 points) This question tests your understanding of random number generation in C. Write a statement that will generate a number in the following ranges:

```
a) [0, 45]
b) [-40, 80]
c) Any even number (including zero) in the range [0, 8]
```

5. (10 points) This problem tests your understanding of composite types. The following program should prompt the user to enter in a student's ID number (an integer), GPA (a floating point number, and name (a string), which are then stored in one element of the array of students. The student information is stored in a `struct`, which is given the name `student_t` using `typedef`.

   Fill in the blanks to complete the program.

```
#include <stdio.h>

#define SIZE 5

int main(int argc, char *argv[])
{
        ____________ student {
                ____________
                ____________
                ____________
        } student_t;

        ____________ class[SIZE];

        int i;

        for( int i = 0; i < SIZE; i++) {
                printf("Please␣enter␣your␣student␣id\n");
                scanf("%____________", ____________);
                printf("Please␣enter␣your␣gpa\n");
                scanf("%____________", ____________);
                printf("Please␣enter␣your␣name\n");
                scanf("%29s", ____________);
        }

        return 0;
}
```

6. (15 points) This problem tests your understanding of arrays and aggregate values. Given a 5x3 matrix, the following function finds the row whose elements sum up to the largest value. The program outputs largest sum and the row index in which it was found. For example, given the following matrix,

```
1 2 3
6 1 2
9 8 1
3 3 4
5 6 8
```

   the output will be:

```
Largest Sum: 19
Row index: 4
```

Fill in the blanks to correctly implement this function. You may assume all numbers in the matrix are greater than 0.

```
_____________ largest_sum(int M[5][3]) {
  int sum = 0, max_val = _____________, max_index;
  int row, col;

  for(row = 0; _____________; row++) {
    for(col = 0; _____________; col++) {
      sum += _____________;
      if (_____________ > _____________) {
        max_val = _____________;
        max_index = _____________;
      }
    }
    _____________;
  }

  printf("Largest␣sum:␣%d\nRow␣index:␣%d",max_val,max_index);
}
```

7. (7 points) This question tests your knowledge of insertion sort. Given an 8 element array `A`:

`15 2 3 16 20 12 1 5`

Show the array `A` after each iteration of the outer loop of insertion sort. You may assume that the numbers are being sorted in ascending order (from low to high).

Iteration 1: ______________________________

Iteration 2: ______________________________

Iteration 3: ______________________________

Iteration 4: ______________________________

Iteration 5: ______________________________

Iteration 6: ______________________________

Iteration 7: ______________________________

8. (15 points) This question tests your knowledge of 2D arrays and random number generation. Please fill in the blanks in the following program to complete the following function: The program must generate an 8x8 chessboard, where the first square (0,0) is black. Black squares are represented by 'b' and white squares are represented by 'w'. Next, it must randomly generate a coordinate that can only be a black square, and replace this with a piece represented by '*'.

   Sample output:

```
b w b w b w b w
w b w b w b w b
b w b w b w b w
w b w b w * w b
b w b w b w b w
w b w b w b w b
b w b w b w b w
w b w b w b w b
```

   More sample output:

```
b w b w b w b w
w b w b w b w b
b w b w b w b w
w * w b w b w b
b w b w b w b w
w b w b w b w b
b w b w b w b w
w b w b w b w b
```

   Fill in the blanks below to implement this functionality (note: the program listing continues on the next page).

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

int main() {

  char a[____________________________][____________________________];
  int r,c;

  for (r = 0; r < ____________________________; r++) {
    for (c = 0; c < ____________________________; c++) {
      if ((____________________________)||(____________________________))
        // determines if the square is black
        a[r][c] = 'b';
      else
        a[r][c] = 'w';
    }
  }

  srand(time(NULL)); // seeds random number gen
  do {               // loops until r and c correspond to a black square
    r = ____________________________;  // generates random number from 0 to 7
    c = ____________________________;  // generates random number from 0 to 7
  } while (____________________________);
  a[r][c] = '*';

  for (r = 0; r < 8; r++) {
    for (c = 0; c < 8; c++) {
      printf("%c␣",a[r][c]);
    }
    printf("\n");
  }

  return 0;
}
```

9. (10 points) This problem tests your understanding of multidimensional arrays. What will be the output of the following program?

```
#include <stdio.h>

#define SIZE 3

void print_cube( int cube[SIZE][SIZE][SIZE] );

int main(void)
{
        int cube[SIZE][SIZE][SIZE];
        int i, j, k;

        for(i = 0; i < SIZE; i++)
        {
                for(j = 0; j < SIZE; j++)
                {
                        for(k = 0; k < SIZE; k++)
                        {
                                cube[i][j][k] = i*j+k;
                        }
                }
        }

        print_cube(cube);
}

void print_cube( int cube[SIZE][SIZE][SIZE] )
{
        int i, j, k;

        for(i = 0; i < SIZE; i++)
        {
                for(j = 0; j < SIZE; j++)
                {
                        for(k = 0; k < SIZE; k++)
                        {
                                printf("%d␣", cube[i][j][k]);
                        }
                        printf("\n");
                }
                printf("\n");
        }
}
```