ENEE 140, Spring 2019
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	Total
Points:	16	8	16	24	10	12	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. (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 = 5;
int b = 4;
char c = 'F';
unsigned d = 6;
```

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
float	r1 = d / a;	1.2
int	r2 = d / b;	**1**
float	r3 = (**float**) d / b;	**1.5**
char	r4 = c − b;	**'B'**
unsigned	r5 = UINT_MAX + b;	**3**
float	r6 = a % d;	**ERROR**
int	r7 = (**int**) a & d;	**4**
int	r8 = b == d;	**0**
char	r9 = c + ++d;	**'M'**

2. (8 points) This question tests understanding of loops. Write the output for the following code:

```
int main () {
    int y=0;
    int b = 3;
    while (y < b) {
        printf("%d -> %d\n",y,y+1);
        y += 2;
    }
    return 0;
}
```

Solution:

```
0 -> 1
2 -> 3
```

3. (16 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 = 0, b = 0, c = 0, d = 0;
    if(b == 1){
        b = 2;
    }
    if(a == 0){
        b = 1;
        c = 1;
        if(d == 1)
            a = 2;
        d = 2;
    } else{
        a = 3;
        b = 2;
    }
    if(b == 2){
        if(c == 1){
            d = 2;
        }else{
            d = 3;
        }
    }else{
        if(d == 1){
            a = 4;
        }
        c = 3;
    }
    printf("a is %d, b is %d, c is %d, d is %d", a,b,c,d);
}
```

Solution:

a is 0, b is 1, c is 3, d is 2

4. (24 points) This question tests your ability to debug code. The follow program should create an array of 10 characters. It should then store the letters A, B, C, D, E, F, G, H, I, and J in the array. Finally, the program should print out the first element in the array (`'A'`). Circle and correct the lines that contain bugs. (Hint: There are 6)

```
1 #include stdio.h
2
3 char get_letter (int i);
```

```
4
5  int main () {
6    int i;
7    char array[];
8
9    for (i=0;i <= 10; i++) {
10     array[i] = get_letter(i);
11   }
12
13   printf("%c\n", array[1]);
14   return 0;
15 }
16
17 void get_letter (int i) {
18   return A + i;
19 }
```

Solution:

```
Line 1: #include <stdio.h>

Line 7: char array[10]

Line 9: for (i=0; i < 10; i++) {

Line 13: printf("%c\n", array[0]);

Line 17: char get_letter (int i) {

Line 18: return 'A' + i;
```

5. (10 points) This problem will test your understanding of functions and character manipulation. The intent of this program is to enter any letter and to print out its upper case version shifted one letter over.

   For example, an entry of `'a'` should print out `'B'`. Additionally, an entry of `'C'` should print out `'D'`. Fill in the blanks within `main()` and the function itself to make this occur. The purpose of the function is to convert lower case letters into uppercase. You can assume that the user will enter a letter.

```
#include <stdio.h>

#include <stdlib.h>

int func(char x);

int main(){

char c;

printf("Enter a letter: ");

scanf("%c",&c);

if(c>='a' && c<='z'){

        c=func(      c      );

}

if(    c=='Z'     ){

        c='A'-1;

}

printf("The letter is %c",     c+1      );


return 0;

}

int func(char x){

        return  x-('a'-'A')  ;

}
```

Solution:

```
#include <stdio.h>

#include <stdlib.h>


int func(char x);


int main(){

char c;

printf("Enter a letter: ");


scanf("%c",&c);

if(c>='a' && c<='z'){
          //Other answers could include c>='a'
          //due to the definition that a letter will be entered.
        c=func(c);

}

if(c=='Z'){

        c='A'-1;

}

printf("The letter is %c",c+1);



return 0;

}


int func(char x){


        return x-('a'-'A');
```

```
}
```

6. (12 points) This question tests your understanding of debugging a program, binary, and bitwise manipulation. The purpose of this program is to take in any number and report back to the user whether the inputted number is a power of 2 or not. Identify the errors within the program that prevent this functionality. There are 4 errors in total. Assume that the computer uses 32 bit integers.

   Remember that, on computers where unsigned integers are represented using 32 bits (b_0, b_1, ... b_{31}), the value of an unsigned integer `U` is computed as a sum of powers of 2:

$$U = \sum_{i=0}^{31} = b_i \cdot 2^i$$

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


int main(){

int i;
unsigned num;
int sum;

printf("Please enter a number: ");
scanf("%u",num);

for(i=31;i>=0;i--){
        sum=sum+((num>>i)|1);
}

if(sum>=0){
        printf("Power of 2");
}

else{
        printf("Not a Power of 2");
}

return 0;
}
```

Solution:

```
#include <stdio.h>

#include <stdlib.h>


int main(){


int i;

unsigned num;

int sum=0;//Sum needs to initialized to 0

printf("Please enter a number: ");

scanf("%u",&num);//Forgot the & sign in scanf


for(i=31;i>=0;i--){

        sum=sum+((num>>i)&1);//Should be & instead of |

}

if(sum==1){//Should be if sum==1

        printf("Power of 2");

}

else{

        printf("Not a Power of 2");

}


return 0;

}
```

7. (14 points) This question will test your ability to write a loop that solves a given problem, and your ability to correctly use arrays in a loop. The following program prompts the user for a cost of an item, and then it determines the number of each type of coin required to make up the cost for that item to use the smallest number of total coins. For example, if the user enters 1.37 for the cost of the item, then the program would print out:

```
Quarters: 5
Dimes: 1
Nickels: 0
Pennies: 2
```

Fill in the blanks in the program.

```
#include <stdio.h>

#define QUART .25
#define DIME .10
#define NICKEL .05
#define PENNY .01

int main() {

    float cost = 0;
    int i = 0;
    int num_coins[4] = ____0,0,0,0____;
    float coin_vals[4] = {QUART, DIME, NICKEL, PENNY};

    printf("How much does the item cost?: ");
    scanf("%f", &cost);

    for(i = ____0____; ____i < 4____; ____i++____) {
        while (cost >= ____coin_vals[i]____) {

            num_coins[i] = ____num_coins[i]____ + 1;
            cost = cost - ____coin_vals[i]____;
        }
    }

    printf("Quarters: %d\n", num_coins[0]);
    printf("Dimes: %d\n", num_coins[1]);
    printf("Nickels: %d\n", num_coins[2]);
    printf("Pennies: %d\n", num_coins[3]);

    return 0;
}
```

Solution:

```
#include <stdio.h>

#define QUART .25
#define DIME .10
#define NICKEL .05
#define PENNY .01

int main() {

    float cost = 0;
    int i = 0;
    int num_coins[4] = {0,0,0,0};
    float coin_vals[4] = {QUART, DIME, NICKEL, PENNY};

    printf("How much does the item cost?: ");
    scanf("%f", &cost);

    for(i = 0; i < 4; i++) {
        while (cost > coin_vals[i]) {

            num_coins[i] = num_coins[i] + 1;
            cost  = cost - coin_vals[i];
        }
    }

    printf("Quarters: %d\n", num_coins[0]);
    printf("Dimes: %d\n", num_coins[1]);
    printf("Nickels: %d\n", num_coins[2]);
    printf("Pennies: %d\n", num_coins[3]);

    return 0;
}
```