ENEE 140, Spring 2014

Midterm Exam

Tuesday, April 8, 2014, 12:30 – 1:45 pm

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: _____Section: _____ Directory ID: _____

1 (12):
2 (15):
3 (15):
4 (8):
5 (30):
6 (20):
TOTAL (100):

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!

Problem 1. (12 points)

This problem tests your understanding of C types and casts. For the following lines of code, explain whether the variables are implicitly or explicitly cast and what type they are casted into. Assume `NUM_A` is an integer, `NUM_B` is a float, and `ANS` is a float.

```
1. ANS = NUM_A * NUM_B;
2. ANS = NUM_A * (int) NUM_B;
3. ANS = (float) NUM_A * NUM_B;
4. ANS = 1.0 * NUM_A
```

1.	
2.	
3.	
4.	

Problem 2. (15 points)

This problem tests your understanding of C characters. Fill in the blanks to complete the following function:

```
// This function returns:
//        1 if the char input is upper case
//       -1 if the char is lowercase
//        0 if it is neither.
int
character_case(char input) {
   if (____)
         return ____;
   else if (____)
         return ____;
   else
         return ____;
}
```

Problem 3. (15 points)

This problem tests your understanding of C loop statements. Edit the following code to convert the `for` loop into a `while` loop:

```
// The following code asks the user for 10 integers
// and prints out their sum.
int i, num, sum = 0;

for (i=0; i<10; i++) {
   printf("Please enter a number:");
   scanf("%d", &num);
   sum = sum + num;
}

printf("%d", sum);
```

Problem 4. (8 points)

This problem tests your understanding of C strings. Consider the following code fragment, which produces a formatted string using the sprintf() function:

```
char s[SIZE_STRING];
sprintf(s, "ENEE %d is the best!\n", 140);
```

`SIZE_STRING` is a constant that you define. How large should `SIZE_STRING` be?

Problem 5. (30 points)

This problem tests your understanding of C control flow and functions. Write the output of the following program (note that the program listing continues on the next page):

```
int
functionOne (int n)
{
  int x;
  int answer=1;
  for (x = 1; x<=n; x++) {
        answer = answer*x;
  }
  return answer;
}

void
functionTwo(int i, int j)
{
  int x;
  int y;
  for (x=0; x<i; x++) {
        for (y=0; y<j; y++) {
              printf("*");
        }
        printf("\n");
  }
}

int
functionThree(int n)
{
  if (n < 10) {
        n = 10;
        return n;
  } else if (n > 10) {
```

```
        n = 20;
        return n;
    } else {
        n = 15;
        return n;
    }
}

int
main()
{
    int x;
    int y[4] = {5,1,15,20};
    int z = 0;

    for (x=0; x<3; x++) {
        printf("x=%5d\n", x);
        if (x == z) {
            printf("f1=%4d\n",functionOne(y[x]));
        } else if (x == y[x]) {
            functionTwo(x, y[x]);
        } else if (x < y[x]) {
            printf("f3=%4d\n",functionThree(z));
        } else {
            printf("None of the above\n");
        }
    }

    return 0;
}
```

_____

_____

_____

_____

_____

_____

_____

_____

Problem 6. (20 points)

This problem tests your understanding of C variables, strings and functions. Consider a function `reverse_string()` that takes a string as argument and that must reverse the order of characters in the string. For example, if the string `s` is "abc", after invoking `reverse_string(s)`, `s` should become "cba". The implementation below contains several bugs. List all the bugs that you can find.

```
int
reverse_string(char s[])
{
      int i, length;
      char c;

      // Determine the string length
      i = 0;
      c = s[0];
      while (c != '\0') {
            c = s[i++];
      }

      length = i;

      for (i=0; i <= length/2; i++) {
            // Swap characters at opposing ends of the string
            s[length-i-1] = s[i];
            s[i] = s[length-i-1];
      }
}
```
