University of Maryland College Park
Dept of Computer Science
CMSC106 Fall 2016
Midterm II Key

Last Name (PRINT): _____

First Name (PRINT): _____

University Directory ID (e.g., umcpturtle)_____

I pledge on my honor that I have not given or received any unauthorized assistance on this examination.

Your signature: _____

Instructions

- This exam is a closed-book and closed-notes exam.
- Total point value is 200 points.
- The exam is a 50 minutes exam.
- Please use a pencil to complete the exam.
- WRITE NEATLY.
- Write your name now.
- **Your code must be efficient.**
- **You don't need to use meaningful variable names; however, we expect good indentation.**

Grader Use Only

#1	Problem #1 (Miscellaneous)	40	
#2	Problem #2 (Memory Map)	30	
#3	Problem #3 (Arrays)	40	
#4	Problem #4 (Loops)	90	
Total	Total	200	

Problem #1 (Miscellaneous)

1. (4 pts) The function process has the prototype below.

```
int process(int data[], int size);
```

Why does passing an array of a million elements to the process function takes the same amount of time as passing an array of two elements? Briefly explain.

Answer: Because we only pass the address of the first element of the array.

2. (4 pts) The following program does not compile. Why?

```
#include <stdio.h>

int main() {
   double a[3] = {2.0, 3.0, 4.0};
   double b[3];

   a = b;
   printf("%d\n", a[0]);

   return 0;
}
```

Answer: Assigning of arrays is not allowed.

3. (4 pts) How can you tell that a character array is a string?

Answer: If there is null character.

4. (4 pts) Write an equivalent prototype to the prototype below.

```
int process(int data[], int size);
```

Answer: int process(int *data, int size), in process(int *, int), etc.

5. (4 pts) Does the following code compile? If so, what is the output.

```
#include <stdio.h>

int main() {
   int x;
   int *p = &x;

   printf("%d", *p);

   return 0;
}
```

Answer: Yes it compiles. Garbage / trash is the output.

6. (4 pts) What will happen when the following code fragment is executed?

```
int *y = NULL;
*y = 5;
printf("%d\n", *y);
```

Answer: Segmentation fault, core dump.

7. (4 pts) Would the following be considered valid pseudocode? Yes/No answer without explanation will receive no credit.

   a. Read two values
   b. Add two values and store result in x
   c. printf("%d", x);

Answer: No, it makes reference to printf that is unique to C.

8. (4 pts) When we read a string using scanf:

   a. scanf makes sure it can fit the string in the provided string variable.
   b. Adds a null character at the end of the string variable.
   c. Adds a newline character at the end of the string variable.
   d. None of the above.

Answer: b.

9. (8 pts) What is the output of the following program?

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

#define MAX 80

int main() {
   char n1[MAX + 1] = "Terps", n2[MAX + 1] = "Terps";


   printf("Val %d\n", strcmp(n1, n2));
   printf("Val2 %d\n", strlen(n1));

   return 0;
}
```

Answer:

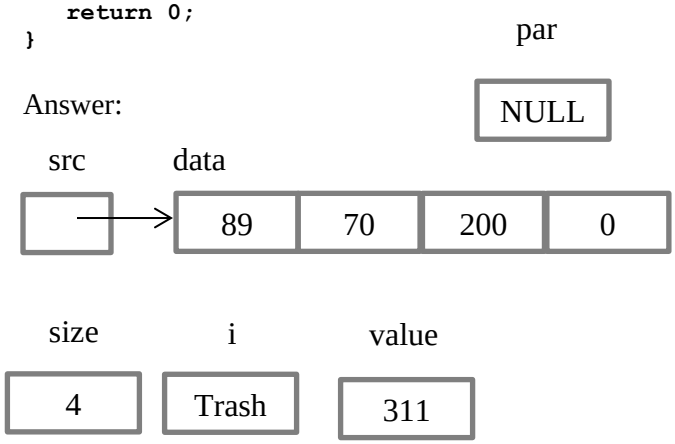
Val 0
Val2 5

Problem #2 (Memory Map)

Draw a memory map to the right of the following program that shows the values of variables when execution has reached the point indicated by /* HERE */.

```
#include <stdio.h>

#define MAX 4

void process(int src[], int value, int *par) {
   src[1] = 70;
   value = 311;
   *(par + 1) = 200;
   par = NULL;
   /* HERE */
}

int main() {
   int data[MAX] = { 89, 4, 5 };
   int size = MAX, i;

   process(data, size, data + 1);
   for (i = 0; i < size; i++) {
      printf("%d\n", data[i]);
   }

   return 0;
}
```

Answer:



Problem #3 (Arrays)

Implement a function named **rotate_right_once** that rotates the elements of an array one position to the right moving the rightmost element to the first array position. For this problem:

- **You will lose significant credit if you declare a new array in the function.**
- The function will not perform any computation if **src** is NULL or **src_length** is less than 1.
- Below we have provided an example of using the function you are expected to implement. Notice we rely on the function print_array which you do not need to implement.

```
int main() {
   int src[] = {7, 3, 10, 12, 19}, length = 5;

   print_array(src, length);
   rotate_right_once(src, length);
   print_array(src, length);

   return 0;
}
```

```
% a.out
Array: 7 3 10 12 19
Array: 19 7 3 10 12
%
```

Answer:

```
void rotate_right_once(int src[], int src_length) {
   if (src != NULL && src_length >= 1) {
      int temp = src[src_length - 1], i;

      for (i = src_length - 1; i >= 1; i--) {
         src[i] = src[i - 1];
      }

      src[0] = temp;
   }
}
```

Problem #4 (Loops)

Implement a function named **draw_diagram** (see prototype on the next page) that creates a triangle of a specified size using two characters. For this problem:

- The triangle to display is left-justified, that is, printing of characters starts on the leftmost column.
- The **size** parameter represents how many lines will be associated with the rectangle.
- The character to use for a particular line must be randomly selected and must be either **first_char** or **second_char** (those are parameters). Use the rand() function to randomly select a character to use.
- The function will use the **count** parameter to return the total number of **first_char** characters that were displayed.
- The function will return 0 and perform no computation if size is less than 1 or if **first_char** is equal to **second_char**; otherwise the function will return 1.
- Below we have provided an example of using the function you are expected to implement.

```
  int main() {
   int count;

   draw_diagram(5, '*', '%', &count);
   printf("First Count: %d\n", count);

   draw_diagram(6, '@', '9', &count);
   printf("Second Count: %d\n", count);

   return 0;
}
```

```
% a.out
*
%%
***
****
*****
First Count: 13
@
99
999
@@@@
@@@@@
999999
Second Count: 10
%
```

Answer:

```
int draw_diagram(int size, char first_char, char second_char, int *count) {
   int row, col;
   char to_print;

   if (size < 1 || (first_char == second_char))  {
      return 0;
   }

   *count = 0;
   for(row = 1; row <= size; row++) {
      to_print = rand() % 2 ? first_char : second_char;
      for (col = 1; col <= row; col++) {
         printf("%c", to_print);
         if (to_print == first_char) {
            (*count)++;
         }
      }
      printf("\n");
   }

   return 1;
}
```