University of Maryland College Park
Dept of Computer Science
CMSC106 Fall 2013
Midterm III

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

- **Make sure you write your name now (we will not wait for you at the end).**
- This exam is a closed-book and closed-notes exam.
- Total point value is 100 points.
- The exam is a 50 minutes exam.
- Please use a pencil to complete the exam.
- WRITE NEATLY.
- You don't need to use meaningful variable names; however, we expect good indentation.

Grader Use Only

#1	Problem 1 (Miscellaneous)	(14)	
#2	Problem 2 (Memory Map)	(16)	
#3	Problem 3 (Arrays)	(45)	
#4	Problem 4 (Strings)	(25)	
Total	Total (100)	(100)	

Problem #1 (14 pts)

1. (2 pts) The function process has the prototype below.

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

   Does passing an array of a million elements to the process function take longer time than passing an array of two elements? Briefly explain. Answers without explanation receive no credit.

2. (3 pts) Write the output generated by the following C program (the program compiles). If there is access to a region of memory that is invalid or should not be accessed, write down INVALID.

```
#include <stdio.h>

#define MAX 3

int main() {
   int a[2 * MAX] = {5, 8, 14};

   printf("A: %d\n", a[2]);
   printf("C: %d\n", a[6]);
   printf("A: %d\n", *a);

   return 0;
}
```

3. (2 pts) Suppose you have an array with 10 characters. How can you tell whether the array is a string? Notice you don't need to write code; just describe what you need to do.

4. (2 pts) Would the following program compile? If it does not compile, indicate why. It is compiles provide the output.

```
#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;
}
```

5. (3 pts) There are several ways to initialize a string variable without using a loop. One is provided below ("Cat" is used to initialize choice). Provide another alternative (i.e., rewrite the right side of the =).

```
char choice[5] = "Cat";
```

6. (2 pts) Which of the following prototype declarations are equivalent? Circle those that are equivalent.

```
void manage_storage(double data[], int size)

void manage_storage(double *data, int size)

void manage_storage(double data[1234], int size)

int manage_storage(double data[], int size)

int manage_storage(double data[], int *size)
```

Problem #2 (16 pts)

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 3

void manage_data(int src[], int range, int *par) {
   src[1] = 10;
   range = -2;
   *(par + 1) = 200;
   par = NULL;
   /* HERE */
}

int main() {
   int orig[MAX] = {89, 4, 7};
   int size = MAX, i;

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

   return 0;
}
```

Problem #3 (45 pts)

1. Implement a function called **find** that has the prototype below. The function returns true if the value parameter exists in the array and false otherwise.

   `int find(int data[], int size, int value)`

2. Implement a function called **common** that has the prototype below. The function determines how many elements two arrays have in common. For this function, you need to use the **find** function you defined above. Feel free to use the **find** function even if you did not implement it. The following is an example of the function you are expected to implement:

```
int main() {
   int a[3] = {10, 70, 8}, b[3] = {10, 70, 8}, c[4] = {3, 8, 6, 70};
   int d[3] = {4, 10, 20}, e[3] = {1000, 2000, 300};

   printf("First: %d, ", common(a, b, 3, 3));
   printf("Second: %d, ", common(b, c, 3, 4));
   printf("Third: %d, ", common(a, d, 3, 3));
   printf("Fourth: %d\n", common(a, e, 3, 3));

   return 0;
}
```

   Output:

   First: 3, Second: 2, Third: 1, Fourth: 0

   `int common(int a[], int b[], int a_size, int b_size)`

PAGE FOR ANSWERS

Problem #4 (25 pts)

Implement a function called **verify_password_choice** that has the prototype below. The function verifies whether a string provided by the user represents a valid password choice. A string is valid if it has at least 8 characters and it is not the word "password". Notice that the function will keep asking the user for a string value as long as an invalid one is provided. The message "Invalid password choice" will be printed each time the user provides an invalid string. The message "Correct password choice" will be printed once a correct value is provided (at this point the function will end).

```
void verify_password_choice()
```

PAGE FOR ANSWERS

ADDITIONAL PAGE IN CASE YOU NEED IT