University of Maryland College Park
Dept of Computer Science
CMSC106 Fall 2015
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

- **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 200 points.
- The exam is a 50 minutes exam.
- Please use a pencil to complete the exam.
- WRITE NEATLY.

Grader Use Only

#1	Problem 1 (Miscellaneous)	(28)	
#2	Problem 2 (Memory Map)	(32)	
#3	Problem 3 (Arrays)	(90)	
#4	Problem 4 (Strings)	(50)	
Total	Total	(200)	

Problem #1, Miscellaneous

1. (3 pts) Write an equivalent prototype to the following prototype.

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

Answer:

int process(int *data, int size); or int process(int *, int); or int process(int data[ANYNUMBER], int size) or any other valid one.

2. (3 pts) When will the following program generate a segmentation fault?

```
#include <stdio.h>

int main() {
   int x, *p = NULL;

   scanf("%d", &x);
   if (x) {
      p = &x;
   }
   printf("%d\n", *p);

   return 0;
}
```

Answer: When 0 is entered

3. (3 pts) Would the following program compile? If it does not compile indicate why. If it compiles provide the output.

```
#include <stdio.h>

int main() {
   int a = 10, b = 20;
   int *m, p;

   m = &a;
   p = &b;

   printf("%d %d\n", *m, *p);

   return 0;
}
```

Answer: No. We cannot assign the address of b to p.

4. (3 pts) Would the following program compile? If it does not compile indicate why. If it 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;
}
```

Answer: No. We cannot change a.

5. (3 pts) Would the following program compile? If it does not compile indicate why. If it compiles provide the output.

```
#include <stdio.h>

int main() {
   int max;
   int *q = &max;

   printf("%d\n", *q);

   return 0;
}
```

Answer: It compiles. A trash / garbage value will be printed.

6. (3 pts) Using typedef create a type called **cmsc106int** that aliases the **int** type C provides.

Answer: typedef int cmsc106int

7. (3 pts) When a character array is a string? Briefly explain.

Answer: When it is null terminated (null character is part of the array).

8. (3 pts) How many null characters are present in the array **data** defined below?

```
char data[300] = "a";
```

Answer: 299

9. (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.

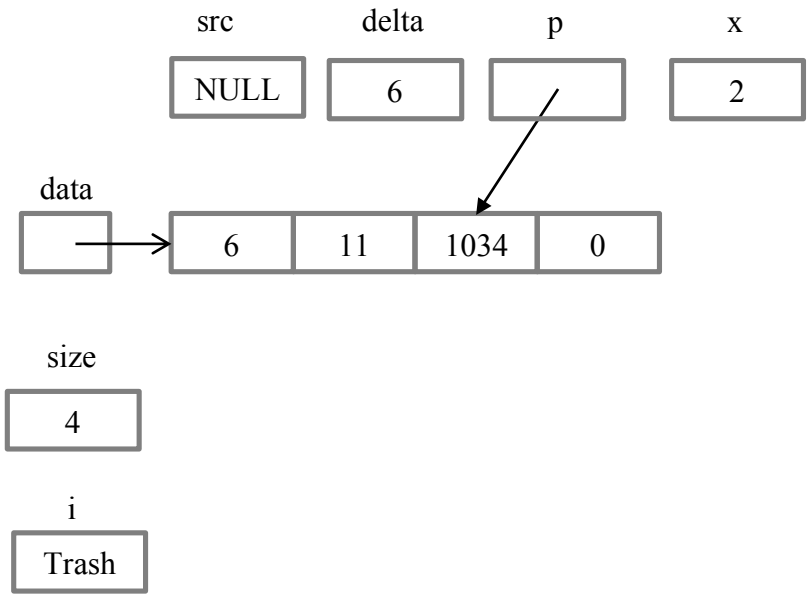
    Answer: b.

Problem #2, Memory Map

Draw a memory map for 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_data(int src[], int delta, int *p) {
   int x = p - src;
   delta += x;
   *p += 1000;

   src = NULL;
   /* HERE */
}

int main() {
   int data[MAX] = {6, 11, 34};
   int size = MAX, i;

   process_data(data, size, data + 2);
   for (i = 0; i < size; i++) {
      printf("%d\n", data[i]);
   }

   return 0;
}
```

Answer:



Problem #3, Arrays

Implement a function called **sub_array_evens** that has the prototype below. The function will initialize the array **result** with even the values that exist (if any) in the **src** array between **start_index** (inclusive) and **end_index** (inclusive). The function will not perform any computation and will return -1 if **src** is null, **src_length** is 0 or **start_index** is greater than **end_index**. Otherwise, the function will return the number of even values found. The driver and expected output below illustrates the functionality of the function you are expected to write. Remember that your function must work for arrays and values other than the ones presented by the example. Notice we are using a **print_array** function you do not need to implement.

Answer:

```
int sub_array_evens(int *src, int src_length, int result[],
                    int start_index, int end_index) {
    int i, j;

    if (src == NULL || src_length == 0 || start_index > end_index) {
       return -1;
    } else {
       j = 0;
       for (i = start_index; i <= end_index; i++) {
           if (src[i] % 2 == 0) {
              result[j++] = src[i];
           }
       }
    }

    return j;
}
```

Problem #4, Strings

Implement a function called **create_password** that has the prototype below. The function initializes the **new_password** parameter with a string created by adding a character (**special_char** parameter) after each character of the **word** parameter. For this problem:

- The **word** parameter may not be modified.
- The function will return -1 if **word** is null, the length of **word** is less than 6 or **word** has the value "password". Otherwise, the function will return the length of the new password.
- You may not use scanf.

The driver and expected output below illustrates the functionality of the function you are expected to write. Remember that your function must work for values other than the ones presented by the example.

Answer:

```
int create_password(const char word[], char special_char, char *new_password) {
   if ((word == NULL) || (strlen(word) < 6) || (strcmp(word, "password") == 0)) {
      return -1;
   } else {
      int i, j = 0;

      for (i = 0; i < strlen(word); i++) {
         new_password[j++] = word[i];
         new_password[j++] = special_char;
      }
      new_password[j] = '\0';
      return j;
   }
}
```