University of Maryland College Park

Dept of Computer Science

CMSC106 Fall 2015

Midterm I 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.
- You do not need to write pseudocode.
- **You don't need to use meaningful variable names; however, we expect good indentation.**

Grader Use Only

#1	Problem 1 (Miscellaneous)	(80)	
#2	Problem 2 (Conditionals)	(40)	
#3	Problem 3 (Loops)	(40)	
#4	Problem 4 (Loops)	(40)	
Total	Total	(200)	

Problem #1 (40 pts)

1. (4 pts) Circle those identifiers considered VALID in C. We are not asking whether they are good identifiers names. We are asking whether they will compile.

   a. **advising**
   b. **large-box**
   c. **coldWeather**
   d. **the@house**
   e. **System##**
   f. **Strong!sounds**

   Answer: a., c.

2. (4 pts) What would happen when the following code fragment is executed? Briefly explain.

```
int i;
for (i = 0; ; i++) {
   printf("%d\n", i);
}
```

   Answer: Infinite loop

3. (4 pts) How many values can you represent with 6 bits?

   Answer: 64

4. (4 pts) Why the following pseudocode is not considered valid pseudocode for code that multiplies a value by 2? Briefly explain.

```
scanf("%d", &x)
Multiply x by 2
Display results
```

   Answer: Valid pseudocode does not rely on language construct like scanf

5. (4 pts) Complete the following printf statement so we can print the following message. Notice that the printf must print the quotes.

Michael "Mike"

```
printf(                                            );
```

   Answer: "Michael \"Mike\""

6. (4 pts) Complete the following printf statement so we can print a 0 (number zero) before any value for x that is between 0 (inclusive) and 9 (inclusive). Any value larger than 9 will not have a preceding 0. You may not add any additional statement. The scanf reads the value we want to print.

```
int x;
scanf("%d", &x);
printf(                                    x);
```

Answer: "%02d"

7. (4 pts) Complete the following printf statement so it generates the output: **11.24**

```
float y = 11.2375;
printf(                                    y);
```

Answer: "%.2f"

8. (4 pts) Write the output generated by the following code fragment.

```
int x = 400, y = 3;
if (x == 400 || --y)
   printf("%d %d", x, y);
else
   printf("700\n");
```

Answer: 400 3

9. (4 pts) Rewrite the following code using a single **if** statement.

```
if (salary <= 50000)
   if (salary > 40000)
      printf("LevelOne\n");
```

Answer:

```
if (salary <= 50000 && salary > 40000)
   printf("LevelOne\n");
```

10. (4 pts) Write the output generated by the following code fragment if the value of x is 90.

```
if (x < 100)
   if (x < 80)
      printf("Dog\n");
else
   printf("Cat\n");
printf("Done");
```

Answer: Done

11. (4 pts) What is the output of the following program?

```
int main() {
   int i;

   printf("%d", i);

   return 0;
}
```

Answer: A garbage/trash value as i has not been initialized.

12. (8 pts) Write a Unix command that will copy the file **public01.in** present in the current directory to the 106 folder that is in your home directory.

Answer:

```
cp public01.in ~/106
```

13. (16 pts) Rewrite the following code using a **switch** statement. Your code must use the minimum number of lines of code possible otherwise you will lose credit.

```
char a = 't';

if (a == 'm') {
   printf("Mon\n");
} else if ((a == 't') || (a == 'T')) {
   printf("Tuesday\n");
} else {
   printf("Other\n");
}
```

Answer:

```
switch(a) {
   case 'm':
      printf("Mon\n");
      break;
   case 't':
   case 'T':
      printf("Tuesday\n");
      break;
   default:
      printf("Other\n");
}
```

14. (12 pts) Rewrite the following code using a for loop where the loop body only has the printf statement. You will lose credit if you have any other additional statements in the body. Variables i and j are integer variables.

```
i = 1;
j = 2;
while (i <= 10) {
   printf("%d %d\n", i, j);
   i++;
   j += 3;
}
```

Answer:

```
for (i = 1, j = 2; i <= 10; i++, j+=3) {
    printf("%d %d\n", i, j);
}
```

Problem #2 (40 pts)

Write a **C program** that decides the week a student can register. Students with a gpa of 3.5 (inclusive) to 4.0 (inclusive) and that have more than 50 (inclusive) credits will register during the first week. Students with a gpa between 3 (inclusive) and less than 3.5 and with less than 50 credits will register during the second week. Any other students will register during the third week. For this program:

- Use the message "Enter gpa and credits:" to read the gpa and number of credits.
- You must have only one scanf statement.
- The program will display as result "First Week", "Second Week" or "Third Week".
- You can assume users will provide a gpa value between 0 (inclusive) and 4.0 (inclusive).
- You can assume users will provide a number of credits between 0 (inclusive) and 200 (inclusive).

Answer:

```
#include <stdio.h>

int main() {
   int credits;
   float gpa;

   printf("Enter gpa and credits: ");
   scanf("%f%d", &gpa, &credits);

   if ((gpa >= 3.5 && gpa <= 4.0) && (credits >= 50)) {
      printf("First Week\n");
   } else if ((gpa >= 3.0 && gpa < 3.5) && credits < 50) {
      printf("Second Week\n");
   } else {
      printf("Third Week\n");
   }

   return 0;
}
```

Problem #3 (40 pts)

Write a **C program** that prints the powers of values that are divisible by 3 that exist between two integer values provided by the user. If the first value is larger than the second the message “Invalid limits” will be displayed and the program will not perform any computation. Use the message “Enter limits:” to read the limit values.

The following is an example that illustrates the functionality of your program when the user enters the values 4 and 17. Notice that your program must work for other scenarios and not just the example below. Underlined text represents input provided by the user. **For this problem you must use a while loop otherwise you will lose significant credit. In addition you can only use one return statement.**

Answer:

```
#include <stdio.h>

int main() {
   int lowerLimit, upperLimit, i;

   printf("Enter limits: ");
   scanf("%d%d", &lowerLimit, &upperLimit);
   if (lowerLimit > upperLimit) {
      printf("Invalid limits\n");
   } else {
      i = lowerLimit;

      while(i <= upperLimit) {
         if (i % 3 == 0) {
            printf("%d %d\n", i, i * i);
         }

         i++;
      }
   }

   return 0;
}
```

Problem #4 (40 pts)

Write a program that reads a number of days and computes the number of minutes associated with those days. The program expects a value between 1 (inclusive) and 31 (inclusive). The program will generate the error message “Invalid value” and keep asking for a value as long as the user provides an invalid one. Once a correct value has been provided the program will print the number of minutes associated with the number of days. Use the message “Total minutes: “ followed by the number of minutes in order to display your answer. **For this problem you must use a do while otherwise you will lose significant credit.**

Answer:

```
#include <stdio.h>

int main() {
   int days, valid;

   do {
      printf("Enter days: ");
      scanf("%d", &days);
      valid = (days >= 1 && days <= 31);
      if (!valid) {
         printf("Invalid value\n");
      }
   } while(!valid);

   printf("Total minutes: %d\n", days * 24 * 60);

   return 0;
}
```