UNCA CSCI 235
Exam 2 (Near Final) Fall 2017
28 November, 2017

This is a closed book and closed notes exam. Communication with anyone other than the instructor is not allowed during the exam. **Furthermore, calculators, cell phones, and any other electronic or communication devices may not be used during this exam.** Anyone needing a break during the exam must leave their exam with the instructor. Cell phones or computers may not be used during breaks.

For this exam, you are allowed a two-sided page with handwritten notes.

If you want partial credit for imperfect answers, explain the reason for your answer!

Name:_____

Problem 1 (20 points) C expressions
In the left column, there are ten tricky and not-so tricky C expressions. Write their values in the right column. Express your answers in base 10. You may assume that all of these numbers are stored in 16-bit two's complement representation.

`~17`	-18
`!17`	0
`17 | 5`	21
`17 || 5`	1
`17 >> 2`	4
`17 << 2`	68
`17 < 5`	0
`17 / 5`	3
`17 == 0x11`	1
`1 && 2017*235`	1

Problem 2 (10 points) Decimal to two's complement conversion
Convert the following two signed decimal numbers into eight-bit *two's complement* representation, the `byte` of Java. Some of these numbers may be outside the range of representation for eight-bit two's complement numbers. Write "out-of-range" for those cases.

130	**-1**
out-of-range	**11111111**

Problem 3 (10 points) Decimal to Q4.4 conversion
Convert the following two signed decimal numbers into Q4.4 *two's complement* numbers (four fixed and four fractional bits) into signed decimal representation. If you can't express the number exactly, give the nearest Q4.4 representation.

-3.75
11000100 think -4 + ¼ or even better -8 + 4 + 1/4
7.3
01110101

Problem 4 (10 points) C structure definition

Define a C data struct (`struct`) that would be appropriate for holding the following three values:

- Name of an automobile brand, *e.g.*, Rambler
- Name of an automobile model, *e.g.*, Palm Beach
- Number of cup holders in automobile, *e.g.*, 6

```
struct Automobile {
  char brand[50] ;
  char model[50] ;
  int cupHolders ;
} ;
// number doesn't have to be 50, but must be a constant
   char[] and String are not C types.
```

```
struct Automobile {
  char *brand ;
  char *model ;
  int cupHolders ;
} ;
```

Problem 5 (10 points) Declaring and initializing a structure
Suppose you had a file containing several lines of comma-separated input similar to the following:

```
Rambler, Palm Beach, 6
```

Further assume that the file pointer `carIn`, of type, `FILE *` (of course) has been opened for reading from this.
Complete a call to `fscanf` to read *one* of these lines into the data structure your wrote in Problem 4. I am giving you the first four tokens needed for the call. (You don't need to check return codes.)

```
// assuming carData is the appropriate structure
fscanf(carIn, " %[^,], %[^,], %d",
              carData.brand,
              carData.model,
              &carData.cupHolders)
```

This is the same format string used in the C programming assignment (and posted "near solution").
The brand and model fields are already addresses, so they don't need the &.

Problem 6 (20 points) Summing up an array
Write a C function called `sumHolders` that, when passed an array of your Problem 4 structures along with the number of elements in that array, returns the total number of cup holders in all the cars passed in the array.

```
// using the name of the structure from Problem 4
int sumHolders (struct automobile cars[], int size) {
  int holderCount = 0 ;
  for (int i = 0; i<size; ++i) {
    holderCount = holderCount + cars[i].cupHolders ;
  }
  return holderCount ;
}

// could also use this header – with no change in function body
int sumHolders (struct automobile *cars, int size)
```

Problem 7 (6 points)
In a recent lab you were introduced to the following structure:

```
struct noteInfo {
    int frequency ;            /* frequency in Hz */
    long duration ;            /* duration in mSec */
}
```

Write a procedure `numPeriods`, with the following signature

```
int numPeriods(struct noteInfo *note)
```

that returns the period of the note in microseconds, that is, 1000000 divided by the frequency of the note. (This is a one-liner.)

```
int numPeriods(struct noteInfo *note) {
   return 1000000 / note->frequency ;
}
// (*note).frequency also works
// In the lab we used
//    1000000l / note->frequency
// because the Arduino has 16-bit integers
```

Problem 8 (14 points)
Suppose your computer can retrieve a value from the cache in 10 µsec and from DRAM in 1 msec.

Using 1000 µsec for 1 msec in answer.

What is the average memory access time if the cache hit rate is 0.99?

0.99*(10 µsec) + 0.01*(1000 µsec)
= 9.9 µsec+ 10 µsec
= 19.9 µsec

What is the average memory access time if the cache hit rate is 0.9?

0.9*(10 µsec) + 0.1*(1000 µsec)
= 9 µsec+ 100 µsec
= 109 µsec