CMSC 106 Quiz 5 Worksheet

The next quiz for the course will be on Wed, Oct 26. The following list provides additional information about the quiz:

- Do not post any solutions to this worksheet in Piazza. That represents an academic integrity violation.
- The quiz will be a written quiz (no computer).
- The quiz will be in lab session.
- Closed book, closed notes quiz.
- Answers must be neat and legible.
- Quiz instructions can be found at http://www.cs.umd.edu/~nelson/classes/utilities/examRules.html

The following exercises cover the material to be included in this quiz. Solutions to these exercises will not be provided, but you are welcome to discuss your solutions with the TA or instructor during office hours. If is recommended that you try this exercises on paper first (without using the computer).

Exercises

1. What is the mechanism used to pass arguments in C?

2. What is casting?

3. Which of the following pointer variables uses the largest number of bytes?

```
int *p;
char *q;
```

4. What will happen when the following code is executed? Explain briefly.

```
int *p = NULL;
*p = 20;
```

5. What will happen when the following code is executed? Explain briefly.

```
int *p;
*p = 20;
```

6. What will happen when the following code is executed? Notice that **a** has not been initialized. Explain briefly.

```
int a;
int *a_ptr = &a;

printf("%p\n", (void *)a_ptr);
```

7. What is the difference between a void pointer variable and a non-void pointer variable (e.g., an integer pointer variable)?

8. Can you compare two pointer variables?

9. What will happen when the following code fragment is executed?

```
int main() {
   int x = 20, y = 40, *p = &x;

   if (x >= 4) {
      p = NULL;
   } else {
      p = &y;
   }
   printf("%d", *p);

   return 0;
}
```

10. Write a memory map and the output for the program below. In cases where you are asked to print an address, write **NULL** or **MEMORY_ADDRESS** (for any other value). To help you understand pointers better you may assume some memory addresses while drawing the memory map (as we did in lecture).

```
#include <stdio.h>

void process(float *passed_card);

void process(float *passed_card) {
   *passed_card += 200;
   printf("In process_one %.2f\n", *passed_card);
   passed_card = NULL;
}

int main() {
   float bank_account = 500.00;
   float *card_one, *card_two;

   card_one = card_two = &bank_account;
   printf("V1: %.2f\n", bank_account);
   printf("V2: %.2f\n", *card_one);
   printf("V3: %.2f\n", *card_two);
   printf("V4: %p\n", (void *)card_two);
   printf("V5: %p\n", (void *)&bank_account);

   *card_two += 20.0;
   printf("V6: %.2f\n", *card_two);
   card_two = NULL;
   printf("V7: %.2f\n", *card_one);

   process(card_one);
   printf("V8: %p\n", (void *)card_one);
   printf("V9: %.2f\n", bank_account);

   return 0;
}
```