CSSE 332 -- Operating Systems

C Review

Name: _____

Question 1. (5 points) Microcontrollers are often much more resource constrained than general-purpose devices, and thus run on a 16-bit CPU architecture. What is the pointer size (in bytes) on such devices?

Question 2. (5 points) Consider a piece code in which a static array `A` is declared as `int A[5];`. What is the initial value of `A[0]`?

Question 3. Consider the following definition of the `element` and `container` structures:

```
1  struct element {
2    int id;
3    int cost;
4    char *name;
5  };
6
7  struct container {
8    int num_elements;           // the number of elements
9    struct element *elements;   // the elements array
10 };
```

(a) (5 points) Write down the syntax used to allocate an array (call it `arr`) of 20 container structures.

(b) (5 points) Assume that the array above has already been created, what is the outcome of executing the following statement:

```
int c = arr->elements[0].cost;
printf("%d", c);
```

Question 4. (5 points) Consider an array of integers created on the heap using

```
int *array = malloc(10 * sizeof(int));
```

Which of the following expressions can be used to access the **sixth** element of the array?

A. `*array + 5`

B. `*array + 6`

C. `*(array + 5)`

D. `*(array + 6)`

E. `*(array + 6*sizeof(int))`

F. `*(array + 5*sizeof(int))`

Question 5. Consider a pointer to a custom structure (defined elsewhere) declared as

```
struct cool_struct *p;
```

(a) (5 points) If we add `5` to p (i.e., do something like `q = p + 5;`), by how many bytes will `q` be away from `p`? _____

(b) (10 points) We would like to move `p` exactly **16** bytes forward and then read the following 4 bytes as an integer. Suggest a way to achieve that using pointer arithmetic.

Note: You do not have access to the code of `struct cool_struct` and so it cannot be changed.

Question 6. (5 points) Consider the following snippet of code:

```c
int add(int x, int y) { return x + y; }

int sub(int x, int y) { return x - y; }

int main(int argc, char **argv) {
  int a = read_int_from_user(); // assume this is implemented elsewhere
  int b = read_int_from_user();
  char op = read_op_from_user();

  int (*op_fn)(int, int) = (op == '+') ? sub : add;
  printf("%d %c %d = %d\n", a, op, b, op_fn(a,b));

  return 0;
}
```

What would be the output on the screen if the user inputs `1`, `3`, and `'-'` when prompted by this program?