CIS 2107. Quiz 3b Solutions. Name:

The quiz is 2 pages and is out of 20 points.

1. **1 point** 5 minutes = ? milliseconds

   1. $5 \cdot 60 \cdot 1000$

2. **1 point** Using the approximation trick we spoke about in class, about how much is $2^{25}$

   2. **32 million**

3. **2 points** What is $B50DDD798_{16} + 58AA23_{16}$ in base 16?

   $\begin{array}{cccccccc}
   B & 5 & 0 & D & D & D & 7 & 9 & S_{16} \\
   + & & 5 & 8 & A & A & 2 & 3 & B_{16} \\
   \hline
   B & 5 & 1 & 3 & 6 & 8 & 1 & B & B_{16}
   \end{array}$

4. **8 points** Write a function which is passed a string. The function returns the number of double letters (i.e., two consecutive letters that are the same) it contains. For example, if the string is shampoo, the function returns 1. If it’s subcommittee, it returns 3. Do not use any functions in <string.h>. No credit given for solutions that use functions in <string.h>.

   **Solution:** One possibility:

   ```c
   int num_double_letters(char s[]) {
       int len=my_strlen(s);
       int i, count=0;

       for (i=1; i<len; i++) {
           if (s[i]==s[i-1])
               count++;
       }

       return count;
   }

   int my_strlen(char s[]) {
       int i=0;

       while (s[i]!='$\0$')
           i++;

       return i;
   }
   ```
5. 8 points A `color_t`, which is defined as `typedef uint32_t color_t;`, is used to store how red, green, and blue a particular color is. Each of the three components (i.e., the red, green, and blue values) is stored as an `unsigned char`. Write a function which is passed a `color_t` and an integer `c` (for component). If `c==0`, return the blue component. If `c==1`, return the green component, if it’s 2, return the red component. For any other value of `c` return the integer 0. For example, if the function is passed a `color_t` with value 0x0027A3FF and `c` with value 1, the function returns the integer 0xA3.

**Solution:** One possibility:

```c
unsigned int pack_color(color_t color, int c) {
    if (c>2 || c<0)
        return 0;
    return color & (0xFF << (c*8));
}
```