Spring 2014 COP 3223 Section 4 Final Exam Solutions

Note: You may declare extra variables for any of the following questions.

1) (10 pts) An ice cream cone costs $3 and a sundae costs $5. Complete the program below so that it prompts the user for how many ice cream cones they want to buy and how many sundaes they want to buy and prints out their final total. (Note: Tax is already included in the prices listed above.)

```
#include <stdio.h>
int main() {

    int cones, sundaes;
    printf("How many ice cream cones do you want to buy?\n");
    scanf("%d", &cones);
    printf("How many sundaes do you want to buy?\n");
    scanf("%d", &sundaes);

    // printf = 1, () = 1, "" = 1, %d = 1, comma = 1
    // formula = 5, 2 for each mult 1 for add
    printf("Your total cost is $%d.\n", 3*cones + 5*sundaes);
    return 0;
}
```

2) (10 pts) Paper costs $5 per ream. It's also sold in large boxes of 100 reams each for $300. Complete the program below so that it prompts the user to enter the number of reams of paper they need and prints out the minimum cost to buy at least than may reams of paper.

```
#include <stdio.h>

int main() {

    int reams;
    printf("How many reams of paper do you want to buy?\n");
    scanf("%d, &reams);

    // Grading: 2 pts correct expression for if
    // 4 pts for > 60 case, 3 pts boxes only, 1 pt correct # boxes
    // 4 pts for <= 60 case, 2 pts for each component

    if (reams%100 > 60)
        printf("You must spend $%d.\n", 300*(reams/100+1));
    else
        printf("You must spend $%d.\n", 300*(reams/100)+5*(reams%100));


    return 0;
}
```

3) (10 pts) Jacqueline runs 2 miles every weekday and 5 miles every weekend day. Assume that day 0 is a Monday. Complete the program below so that takes a starting day number and an ending day number from the user and prints out a report with a daily log of how far Jacqueline ran that day and the total number of miles she ran up to that day, counting from the start day. For example, if the user entered 4 for the starting day and 8 for the ending day, then the print out should look like:

```
Day       Miles     Total
4         2         2
5         5         7
6         5         12
7         2         14
8         2         16
```

```
#include <stdio.h>

int main() {

    int start, end, i, total = 0;
    printf("Enter the starting and ending day.\n");
    scanf("%d%d", &start, &end);

    printf("Day\tMiles\tTotal\n");           // 0 pts
    for (i=start; i<=end; i++) {             // 2 pts
        int day = 2;                      // These 3
        if (i%7 > 4)                      // lines are
            day = 5;                      // 4 pts
        total += day;                     // 1 pt
        printf("%d\t%d\t%d\n", i, day, total); // 3 pts - 1
    }                                            // for each

    return 0;
}
```

4) (8 pts) Describe what each of the following four string functions in string.h do.

strcmp: **Makes a lexicographical comparison between two strings. (2 pts) In laymen's terms, this is similar to an alphabetical comparison. It returns a negative integer if the first parameters "comes before" the second, 0 if they're equal and a positive integer if the first parameter "comes after" the second. (Only the first sentence is needed, and just comparison is good enough.)**

strlen: **Returns the length of a string. (2 pts)**

strcat: **Concatenates the second string to the end of the first. (2 pts)**

strcpy: **Copies the contents of the second string into the first. (2 pts)**

5) (10 pts) The next two questions concern manipulating grayscale picture images. These images can be stored in two dimensional integer arrays with each value in the array ranging from 0 (black) to 255 (white). Functions that edit pictures take in the picture as a two dimensional integer array and then make the appropriate changes to each pixel of the array. The function below "lightens" each pixel by adding 10 to its value, with a maximum of 255:

```
#define HEIGHT 200
#define WIDTH 300

void lighten(int pic[][WIDTH]) {

    int i, j;
    for (i=0; i<HEIGHT; i++) {
        for (j=0; j<WIDTH; j++) {
            pic[i][j] += 10;
            if (pic[i][j] > 255)
                pic[i][j] = 255;
        }
    }
}
```

Write a function that takes in an image like the function above and flips the grayscale value of each pixel. To flip pixel values, note that we want to replace a value of 0 with 255, a value of 1 with 254, a value of 2 with 253, etc. Basically, the darkest pixels become the lightest ones and vice versa, with everything mapping linearly in between.

```
void invertColors(int pic[][WIDTH]) {

    int i, j;                                   // 1 pt
    for (i=0; i<HEIGHT; i++)                    // 2 pts
        for (j=0; j<WIDTH; j++)                 // 2 pts
            pic[i][j] = 255 - pic[i][j];        // 5 pts -
                                                // LHS = 1 pt
                                                // RHS = 4 pts

}
```

6) (10 pts) Write a function that takes in an image and reflects the picture along the vertical line that divides the picture in two. Intuitively, the function will change the picture to be its own mirror image. Specifically, for each pixel of the form pic[i][0] should be swapped with each pixel of the form pic[i][WIDTH-1], each pixel of the form pic[i][1] should be swapped with the pixel of the form pic[i][WIDTH-2], each pixel of the form pic[i][2] should be swapped with the pixel of the form pic[i][WIDTH-3], etc. Note that the resulting picture has no new pixel values, but each pixel has moved to a different location in the picture than where it was previously.

```
void mirrorImage(int pic[][WIDTH]) {


    int i, j;                                   // 1 pt
    for (i=0; i<HEIGHT; i++) {                  // 2 pts
        for (j=0; j<WIDTH/2; j++) {             // 3 pts
            int tmp = pic[i][j];                // 1 pt
            pic[i][j] = pic[i][WIDTH-1-j];      // 2 pts
            pic[i][WIDTH-1-j] = tmp;            // 1 pt
        }
    }


}
```

7) (10 pts) Complete the function below so that it takes in an int array, the length of the array, and a minimum value and returns the number of values in the array that are greater than or equal to that minimum value. For example, if we gave the function the array [23, 12, 18, 16, 18, 22] and a minimum value of 18, it should return 4, since 23, 18, 18 and 22 are at least as big as 18.

```
int numMeetThreshold(int array[], int length, int minimum) {

    int cnt = 0, i;                         // 1 pt
    for (i=0; i<length; i++)                // 3 pts
        if (array[i] >= minimum)            // 3 pts
            cnt++;                          // 2 pts

    return cnt;                             // 1 pt
}
```

8) (10 pts) Complete the function below so that it returns 1 if the second string is a substring of the first string. A substring is a contiguous portion of a given string. For example "baseball" contains the substrings "base", "ball", "sebal" and "baseball". It does NOT contain the substrings "sb", "baseballs" or "sell".

```
int contains(char str[], char sub[]) {
    int i, j;
    int lenStr = strlen(str);
    int lenSub = strlen(sub);


    for (i=0; i <= lenStr-lenSub; i++) { // 2 pts

        int cnt = 0;
        for (j=0; j < lenSub; j++)      // 2 pts

            if ( sub[j] == str[i+j] )   // 3 pts
                cnt++;

        if ( cnt == lenSub )            // 2 pts
            return 1;
    }

    return 0 ;                          // 1 pt
}
```

9) (10 pts) The struct shown below stores a fraction. Write a method that takes in two struct fractions and returns a struct fraction representing the sum of the two input fractions. Do not reduce the resultant fraction (for ease of grading) and don't worry about overflow errors.

```
struct fraction {
    int num;
    int den;
};

struct fraction add(struct fraction op1, struct fraction op2) {

    struct fraction ans;                          // 1 pt
    ans.num = op1.num*op2.den + op2.num*op1.den;  // 5 pts
    ans.den = op1.den*op2.den;                    // 3 pts
    return ans;                                   // 1 pt



}
```

10) (10 pts) Using the same fraction struct from the previous question, write a fraccmp function that returns a positive integer if the first fraction is bigger than the second, 0 if the two are equivalent in value, and a negative integer if the first is smaller than the second. For example, if a = 3/5, b = 1/2 and c = 6/10, fraccmp(a,b) should return a positive integer, fraccmp(b,c) should return a negative integer, and fraccmp(a,c) shold return 0. The struct definition is given for you again below. **YOU MAY ASSUME THAT BOTH op1 and op2 represent positive fractions.**

```
struct fraction {
    int num;
    int den;
};

int fraccmp(struct fraction op1, struct fraction op2) {

    return op1.num*op2.den - op2.num*op1.den;

    // Grading: Most will write more than this.
    // 3 pts for numerator of first fraction
    // 3 pts for numerator of second fraction
    // 4 pts for returning accordingly

}
```

11) (2 pts) What is the first name of the author of I am Malala, the autobiographical account of a young human rights advocate?

Malala (2 pts give to all)