CSc 3320: Systems Programming

Spring 2021

Homework

4: Total points 100

Submission instructions:

1. Create a Google doc for each homework assignment submission.
2. Start your responses from page 2 of the document and copy these instructions on page 1.
3. Fill in your name, campus ID and panther # in the fields provided. If this information is missing in your document TWO POINTS WILL BE DEDUCTED per submission.
4. Keep this page 1 intact on all your submissions. If this *submissions instructions* page is missing in your submission TWO POINTS WILL BE DEDUCTED per submission.
5. Each homework will typically have 2-3 PARTS, where each PART focuses on specific topic(s).
6. Start your responses to each PART on a new page.
7. If you are being asked to write code copy the code into a separate txt file and submit that as well.
8. If you are being asked to test code or run specific commands or scripts, provide the evidence of your outputs through a screenshot and copy the same into the document.
9. Upon completion, download a .PDF version of the document and submit the same.

Full Name: Tracy Michaels

Campus ID: tmichaels1

Panther #: 002430918

ALL PROGRAMS MUST BE COMMENTED. YOUR SOLUTION WILL NOT BE ACCEPTED IF THERE ARE NO COMMENTS IN YOUR SCRIPT. Also note that the comments MUST be useful and not be random.

PART 1: 40pts
Must incorporate use of Functions and Pointers

1. Write a C program `checkPasswd.c` to check if the length of a given password string is 10 characters or not. If not, deduct 5 points per missing character. If the total deduction is greater than 30 points, print out the deduction and message "The password is unsafe! Please reset."; otherwise, print out "The password is safe."

hw4 > C checkPasswd.c

```c
#include<stdio.h>
#include<string.h>

//Tracy Michaels
//System Level Programming Homework 4

//this program checks for the safty of a password
//password is unsafe if deductions are greater than 30 points
// - deduct 5 points for each character under length of 10

void checkLength(char *, int *);
const char* checkSafty(int);
int stringLength(char);

int main(){

    char inputPasswd[20];
    int deductions = 0;

    printf("Enter Password: ");
    scanf("%s", &inputPasswd);

    checkLength(inputPasswd, &deductions);
    printf("%s", checkSafty(deductions));

    return 0;
}

//check if length of password is less than 10 characters long.
//deduct 5 points for every missing character
void checkLength(char *passwd, int *deduct){
    if((int) strlen(passwd) < 10){
        *deduct += (10 - ((int) strlen(passwd))) * 5;
    }
}

//returns if the password is safe or not
//safe if no more than 30 points have been deducted
const char* checkSafty(int deduction){
    printf("total deductions: %d\n", deduction);
    return (deduction > 30) ? "The password is unsafe! Please reset.\n" : "The password i
}

```

```
[tmichaels1@gsuad.gsu.edu@snowball hw4]$ ./checkPasswd
Enter Password: thisisapassword
total deductions: 0
The password is safe
[tmichaels1@gsuad.gsu.edu@snowball hw4]$ ./checkPasswd
Enter Password: pass
total deductions: 30
The password is safe
[tmichaels1@gsuad.gsu.edu@snowball hw4]$ ./checkPasswd
Enter Password: pas
total deductions: 35
The password is unsafe! Please reset.
[tmichaels1@gsuad.gsu.edu@snowball hw4]$
```

2. Similar to above question, update the C program `checkPasswd.c` to check if a password is safe or by not by checking only the evaluation criteria below. It will still print out the final score, and "safe" or "unsafe" when deduction is more than 30 points.

- Missing lower case -20 points
- Lack of capital letters -20 points
- Missing numbers -20 points
- More than 2 consecutive characters (e.g. 123 or abc) -20 points

hw4 > C checkPasswd.c

```c
#include<stdio.h>
#include<string.h>

//Tracy Michaels
//System Level Programming Homework 4

//this program checks for the safty of a password
//password is unsafe if deductions are greater than 30 points
// - deduct 5 points for each character under length of 10
// - deduct 20 points for missing a lowercase letters
// - deduct 20 points for missing capital letters
// - deduct 20 points for missing letters
// - deduct 20 points for having consecutive characters

void checkLength(char *, int *);
void checkCriteria(char *, int *);
const char* checkSafty(int);

int main(){

    char inputPasswd[40];
    int deductions = 0;

    printf("Enter Password: ");
    scanf("%s", &inputPasswd);

    checkLength(inputPasswd, &deductions);
    checkCriteria(inputPasswd, &deductions);
    printf("%s", checkSafty(deductions));

    return 0;
}

//check if length of password is less than 10 characters long.
//deduct 5 points for every missing character
void checkLength(char *passwd, int *deduct){
    if((int) strlen(passwd) < 10){
        *deduct += (10 - ((int) strlen(passwd))) * 5;
    }
}

//returns if the password is safe or not
//safe if no more than 30 points have been deducted
const char* checkSafty(int deduction){
    printf("total deductions: %d\n", deduction);
    return (deduction > 30) ? "The password is unsafe! Please reset.\n" : "The password i
}

void checkCriteria(char *passwd, int *deduct){
    char *p = passwd;
    int numCap = 0;
    int numLow = 0;
    int numNum = 0;
    int numConsecutive = 0;
    char prev;
    int i = 0;

    //TODO remove debugs
    printf("before for\n");
    for (i; passwd[i] != '\0'; i++){
        //printf("character: %c\n", passwd[i]);
        if(passwd[i] >= 'A' && passwd[i] <= 'Z') ++numCap;
        //printf("numCap: %d\n", numCap);
        if(passwd[i] >= 'a' && passwd[i] <= 'z') ++numLow;
        //printf("numLow: %d\n", numLow);
        if(passwd[i] >= '0' && passwd[i] <= '9') ++numNum;
        //printf("numNum: %d\n", numNum);
        if(passwd[i] == prev) numConsecutive++;
        //printf("numCon: %d\n", numConsecutive);
        prev = passwd[i];
    }
    printf("after for\n");

    if(numCap < 1) *deduct += 20;
    if(numLow < 1) *deduct += 20;
    if(numNum < 1) *deduct += 20;
    if(numConsecutive > 0) *deduct += 20;
}
```

```
[tmichaels1@gsuad.gsu.edu@snowball hw4]$ gcc -o checkPasswd checkPasswd.c
[tmichaels1@gsuad.gsu.edu@snowball hw4]$ ./checkPasswd
Enter Password: Th1s1sAStr0ngPasSw0rd
total deductions: 0
The password is safe
[tmichaels1@gsuad.gsu.edu@snowball hw4]$ ./checkPasswd
Enter Password: weakpass
total deductions: 70
The password is unsafe! Please reset.
[tmichaels1@gsuad.gsu.edu@snowball hw4]$
```

Part II : 40pts
Must incorporate the use of Functions and Pointer arrays

3. Write a program that reads a message (can be characters, numeric or alphanumeric) and checks whether it is a palindrome (the characters in the message are the same when read from left-to-right or right-to-left).

hw4 > C Palindrome.c

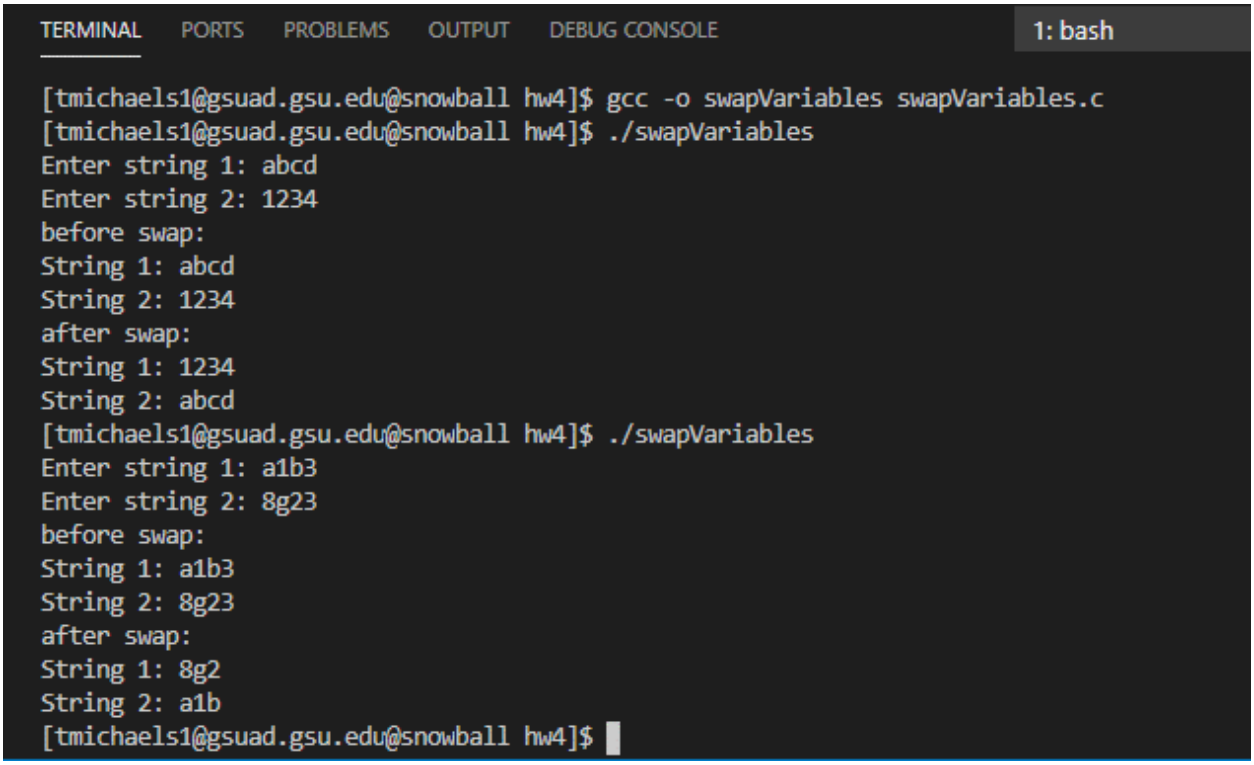
```c
#include<stdio.h>
#include<string.h>

//Tracy Michaels
//This program tell user if an entered message is a palindrome

#define MAXIMUM 256

int isPalindrome(char *);

int main(){

    char message[MAXIMUM];

    printf("Enter Message: ");
    fgets(message, MAXIMUM, stdin);

    (isPalindrome(message)) ? printf("Is a Palindrome\n") : printf("Not a Palindrome\n");

    return 0;
}


//function to determine if a string is a palindrome
//returns 1 if it is a palindrome and
//0 if it is not
int isPalindrome(char *s){
    int i, j;
    for(i = 0, j = strlen(s) - 2; i <= strlen(s)/2; i++, j--){

        //skip white space
        if(s[i] == ' ') i++;
        if(s[j] == ' ') j--;

        if(i >= j){
            return 1;
        } else if(s[i] != s[j]){
            return 0;
        }
    }
}
```

```
[tmichaels1@gsuad.gsu.edu@snowball hw4]$ gcc -o Palindrome Palindrome.c
[tmichaels1@gsuad.gsu.edu@snowball hw4]$ ./Palindrome
Enter Message: aaaa
Is a Palindrome
[tmichaels1@gsuad.gsu.edu@snowball hw4]$ ./Palindrome
Enter Message: radar
Is a Palindrome
[tmichaels1@gsuad.gsu.edu@snowball hw4]$ ./Palindrome
Enter Message: asdf
Not a Palindrome
[tmichaels1@gsuad.gsu.edu@snowball hw4]$ ./Palindrome
Enter Message: murder for a jar of red rum
Not a Palindrome
[tmichaels1@gsuad.gsu.edu@snowball hw4]$ gcc -o Palindrome Palindrome.c
[tmichaels1@gsuad.gsu.edu@snowball hw4]$ ./Palindrome
Enter Message: murder for a jar of red rum
Is a Palindrome
[tmichaels1@gsuad.gsu.edu@snowball hw4]$
```

4. Write a program that will swap two variables without the use of any third variable. Utilize this program to write a program that reads two sentences that contain alphanumeric characters and the program must swap all the numerics in sentence1 with alphabet characters from sentence 2 and vice-versa. Keep the lengths of the sentences as identical.

hw4 > C swapVariables.c

```c
#include<stdio.h>
#include<string.h>

//Tracy Michaels
//This program swaps 2 variables without the use of a 3rd
//it is then utilized in swapSentence  and swap all numbers in one
// with letters in the other and vise versa
//precondition: variables must be of same length

#define MAXIMUM 256

void swapSentence(char *, char *);

int main(){

    char string1[MAXIMUM];
    char string2[MAXIMUM];

    //get strings
    printf("Enter string 1: ");
    fgets(string1, MAXIMUM, stdin);

    printf("Enter string 2: ");
    fgets(string2, MAXIMUM, stdin);

    if(strlen(string1) != strlen(string2)) {
        printf("Strings must be same length\n");
        return 0;
    }

    printf("before swap:\n");
    printf("String 1: %s", string1);
    printf("String 2: %s", string2);

    swapSentence(string1, string2);

    printf("after swap:\n");
    printf("String 1: %s\n", string1);
    printf("String 2: %s\n", string2);

    return 0;
}

void swapSentence(char *str1, char *str2){


    int i = 0;
    while(str1[i] != '\0'){
        if(((str1[i] >= '0' && str1[i] <= '9') && (str2[i] >= 'A' && str2[i] <= 'z'))
        || ((str2[i] >= '0' && str2[i] <= '9') && (str1[i] >= 'A' && str1[i] <= 'z'))) {
            str1[i] = str1[i] + str2[i];
            str2[i] = str1[i] - str2[i];
            str1[i] = str1[i] - str2[i];
        } else {
            str1[i] = ' ';
            str2[i] = ' ';
        }
        i++;
    }

    str1[i] = '\0';
    str2[i] = '\0';
}

```


TERMINAL
PORTS
PROBLEMS
OUTPUT
DEBUG CONSOLE
1: bash
[tmichaels1@gsuad.gsu.edu@snowball hw4]$ gcc -o swapVariables swapVariables.c
[tmichaels1@gsuad.gsu.edu@snowball hw4]$ ./swapVariables
Enter string 1: abcd
Enter string 2: 1234
before swap:
String 1: abcd
String 2: 1234
after swap:
String 1: 1234
String 2: abcd
[tmichaels1@gsuad.gsu.edu@snowball hw4]$ ./swapVariables
Enter string 1: a1b3
Enter string 2: 8g23
before swap:
String 1: a1b3
String 2: 8g23
after swap:
String 1: 8g2
String 2: a1b
[tmichaels1@gsuad.gsu.edu@snowball hw4]$

Part III : 20pts
Must incorporate Functions, Pointers or PointerArrays, and Structures or Unions

5. Write a program that asks the user to enter an international dialing code and then looks it up in the country_codes array (see Sec 16.3 in C textbook). If it finds the code, the program should display the name of the corresponding country; if not, the program should print an error message. For demonstration purposes have at least 20 countries in your list.
   (Programming Project 1 on pg412 in C textbook)

hw4 > C countryLookup.c

```c
#include<stdio.h>

//Tracy Michaels
//this program prompts the user to enter a countries code and
//if it exists the prints the name of the country

#define NUM_CODES 20

//struct containing country codes from section 16.3 in book
struct dialing_code{
    char *country;
    int code;
};

const struct dialing_code country_code[NUM_CODES] =
    {
        {"Argentina", 54}, {"Bangladesh", 880},
        {"Brazil",    55}, {"Burma",       95},
        {"China",     86}, {"Colombia",    57},
        {"Congo",    243}, {"Egypt",       20},
        {"Ethopia",  251}, {"France",      33},
        {"Germany",   49}, {"India",       91},
        {"Indonesia", 62}, {"Iran",        98},
        {"Italy",     39}, {"Japan",       81},
        {"Mexico",    52}, {"Nigera",     234},
        {"Poland",    48}, {"Russia",       7}
    };

void lookup(int);

int main(){

    int input;

    printf("Enter coutnry code: ");
    scanf("%d", &input);
    lookup(input);
    return 0;
}

//checks if entered code is in country_codes
//prints country if it is
void lookup(int n){
    int i = 0;
    for(; i < NUM_CODES; i++){
        if(country_code[i].code == n){
            printf("%s\n", country_code[i].country);
            return;
        }
    }
    printf("Code not found\n");
}
```

```
[tmichaels1@gsuad.gsu.edu@snowball hw4]$ gcc -o countryLookup countryLookup.c
[tmichaels1@gsuad.gsu.edu@snowball hw4]$ ./countryLookup
Enter coutnry code: 49
Germany
[tmichaels1@gsuad.gsu.edu@snowball hw4]$ ./countryLookup
Enter coutnry code: 55
Brazil
[tmichaels1@gsuad.gsu.edu@snowball hw4]$ ./countryLookup
Enter coutnry code: 1
Code not found
[tmichaels1@gsuad.gsu.edu@snowball hw4]$
```