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: Hakan Gunerli

Campus ID: hgunerli1

Panther #: 002504797

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."

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

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).

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.

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)

PART I:

Section I:

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

int main(){

char ch[10]; // initiate 10 character string

printf("Enter a password : ");
scanf("%s", ch); // scan that

int length=strlen(ch); //  the amount of characters in our ch string
int points,length_sub_ten; // we'll use these variables to calculate the p
oints

if(length!=10)
{ // if the amount of character is not 10, which is safe, run

length_sub_ten=10-length; // how many characters are missing,
points=5*length_sub_ten; // give that a number

if(points>30) // if the point calculated is higher than 30, give an error
{ printf("Deduction points amount: %d \n The password is unsafe! Please re
set.", points); }
else { printf("The password is safe."); }

}

else
{ //  this is for the upper if.
 printf("The password is safe.");
}

return 0;

}
```

main.c

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

int main(){

char ch[10]; // initiate 10 character string

printf("Enter a password : ");
scanf("%s", ch); // scan that

int length=strlen(ch); //  the amount of characters in our ch string
int points,length_sub_ten; // we'll use these variables to calculate the points

if(length!=10)
{ // if the amount of character is not 10, which is safe, run

length_sub_ten=10-length; // how many characters are missing,
points=5*length_sub_ten; // give that a number

if(points>30) // if the point calculated is higher than 30, give an error
{ printf("Deduction points amount: %d \n The password is unsafe! Please reset.", points); }
else { printf("The password is safe."); }

}

else
{ //  this is for the upper if.
 printf("The password is safe.");
}

return 0;

}
```

Console Shell

```
> clang-7 -pthread -lm -o main main.c
> ./main
Enter a password : abc
Deduction points amount: 35
 The password is unsafe! Please reset.>
```

SECTION II:

```c
//import useful libraries...
#include <stdio.h>
#include <string.h>

int main() {
char password[10]; //declare char string,
int points = 0;

printf("Enter the password: ");
scanf("%s",password); //Tae password from the user...

int len = strlen(password); // length of the value


int lower_count = 0;
int upper_count=0;
for(int i = 0;i<len;i++)
{
if(password[i] >= 'A' && password[i] <= 'Z')  // user input and iterate th
rough the passwd [UPPERCASE]
{
upper_count += 1;
}
if(password[i] >= 'a' && password[i] <= 'z') // user input and iterate thr
ough the passwd [LOWERCASE]
{
```

```
lower_count += 1;
}

}


if(upper_count < 2) // lack of capital letters, implying there needs to be
 at least 2
{
points += 20;
}

if(lower_count <= 0) // missing lower case, implying we only need one.
{
points += 20;
}


int num_count=0;

for(int i = 0;i<len;i++)
{
if(password[i] >= '0' && password[i] <= '9')
{
num_count += 1;
}
}

if(num_count <= 0)
{
points += 20;
}

int consecutive_count = 0;
// we'll do nested iteration to check for consective counts.
for(int i=0;i<len;i++)
{
for(int j = i+1;j<len;j++)
{
if(password[j] - password[i] == 1)  { consecutive_count += 1;} // if the i
 and the one next to it is the same, then consecutive_count goes up
}
}
```

```c
if(consecutive_count >= 2) // if there are more than 2 consecutive, add po
ints to the tally
{
points += 20;
}
if(points > 30) // finally if the deduction points is higher than 30, say
the passwd is bad.
{
printf("Deduction Points: %d, The password is unsafe! Please reset.", poin
ts);
}
else
{
printf("Deduction Points: %d, the password is safe.",points);
}

return 0;
}
```

main.c

```c
//import useful libraries...
#include <stdio.h>
#include <string.h>

int main() {
char password[10]; //declare char string,
int points = 0;

printf("Enter the password: ");
scanf("%s",password); //Tae password from the user...

int len = strlen(password); // length of the value


int lower_count = 0;
int upper_count=0;
for(int i = 0;i<len;i++)
{
if(password[i] >= 'A' && password[i] <= 'Z')  // user input and iterate through the passwd [UPPERCASE]
{
upper_count += 1;
}
if(password[i] >= 'a' && password[i] <= 'z') // user input and iterate through the passwd [LOWERCASE]
{
lower_count += 1;
}

}


if(upper count < 2) // lack of capital letters, implying there needs to be at least 2
```

Console Shell

```
> clang-7 -pthread -lm -o main main.c
> ./main
Enter the password: Test
Deduction Points: 40, The password is unsafe! Please reset.>
```

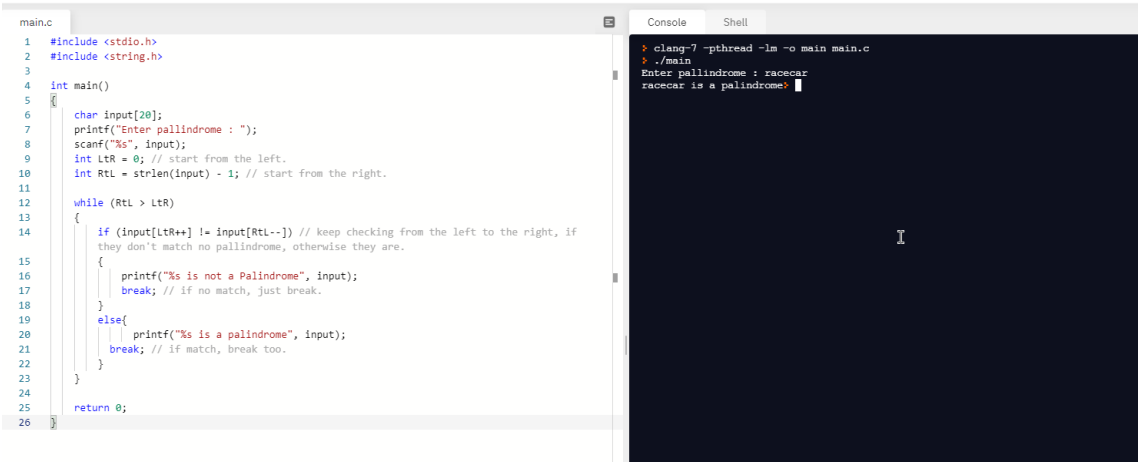
PART II:
SECTION I:

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

int main()
{
    char input[20];
    printf("Enter pallindrome : ");
    scanf("%s", input);
    int LtR = 0; // start from the left.
    int RtL = strlen(input) - 1; // start from the right.

    while (RtL > LtR)
    {
        if (input[LtR++] != input[RtL--
]) // keep checking from the left to the right, if they don't match no pal
lindrome, otherwise they are.
        {
            printf("%s is not a Palindrome", input);
            break; // if no match, just break.
        }
        else{
              printf("%s is a palindrome", input);
          break; // if match, break too.
        }
    }

    return 0;
}
```



SECTION II:

```
#include <stdio.h>
#include <string.h>
    void subs(char firstStr[], char secondStr[], int start, int end){  //
i'll make this a function for substringing.
       int i=0;
       while ((i) < end) {   // while the length exists, i want to add the
 the value of firstStr (start+i) into secondStr.
          secondStr[i] = firstStr[start+i];  // iterate
          i++;  // keep doing until reaching end
       }
    }

int main()
{
    char str1[20] = "testingthis";
    char str2[20] = "valuetestin";

    printf("string one : %s",str1);
    printf("\nstring two:  %s ", str2 );

    printf("\n---------------------------");

    strcat(str1, str2);   // concat the strings into a large array

    subs(str1, str2, 0, (strlen(str1) - strlen(str2)));  // iterate starti
ng from 0 to the length starting from the val of str1 - str2

    subs(str1, str1, strlen(str2), strlen(str1));  // iterate within the l
enght of the char arrays

    printf("\nstring one : %s",str1);
    printf("\nstring two:  %s", str2 );

    return 0;
}
```

main.c

```c

#include <stdio.h>
#include <string.h>
    void subs(char firstStr[], char secondStr[], int start, int end){  // i'll make this a function for substringing.
        int i=0;
        while ((i) < end) {   // while the length exists, i want to add the the value of firstStr (start+i) into secondStr.
            secondStr[i] = firstStr[start+i];  // iterate
            i++;  // keep doing until reaching end
        }
    }

int main()
{
    char str1[20] = "testingthis";
    char str2[20] = "valuetestin";

    printf("string one : %s",str1);
    printf("\nstring two:  %s ", str2 );

    printf("\n---------------------------");

    strcat(str1, str2);   // concat the strings into a large array

    subs(str1, str2, 0, (strlen(str1) - strlen(str2)));  // iterate starting from 0 to the length starting from the val of str1 - str2

    subs(str1, str1, strlen(str2), strlen(str1));  // iterate within the lenght of the char arrays

    printf("\nstring one : %s",str1);
    printf("\nstring two:  %s", str2 );

    return 0;
}

```

Console Shell

```
> clang-7 -pthread -lm -o main main.c
> ./main
string one : testingthis
string two:  valuetestin
---------------------------
string one : valuetestin
string two:  testingthis> 
```

PART III:

```
#include <stdio.h>

int main (int argc, char* argv[])
{

struct dialing_code {
    char *country;
    int code;
};


    int intl_code; // code for exit, in case the user wants to exit the pr
ogram.

    const struct dialing_code country_codes[] = {
        {"Argentina", 54}, {"Bangladesh",        880},  {"Germany",    49}
,          {"India",              91},
       {"Indonesia", 62}, {"Russia",              7},  {"Brazil",     55},
          {"Iran",               98},
        {"Ethiopia", 251}, {"France",             33},  {"China",      86}
,         {"Colombia",           57},
        {"Italy",      39}, {"Congo,Dem.",        243}, {"Egypt",     20},
         {"Japan",              81},
        {"Mexico",     52}, {"Nigeria",           234}, {"Burma",    95},
          {"Turkey",     90},{"United States",     1},


        };


    do // i would like to ask the same question until i get a response, or
 a quit sequence so i'll use do-while.
    {
      int country_code_found; // a variable for holding the value.

        printf("Please input the international code: "); // ask input
        scanf("%d", &intl_code);

        for (int i = 0; i < sizeof(country_codes) / sizeof(*country_codes)
 ; i++) { // for loop until we reach the size of the country codes, array
and pointer.
            if (country_codes[i].code == intl_code) // if we have a match
within each iteration and input
```

```
            {
                printf("The country is: %s\n", country_codes[i].country);
// print the code and the country
             country_code_found+=1;
                            break;
            }
        }
        if (!country_code_found){ // if the found code has not changed, th
en it's code not found.
  printf("Code not found.\n");
        }

    } while(1); // same thing as saying while True,

    return 0;
}
```

main.c

```
#include <stdio.h>

int main (int argc, char* argv[])
{

struct dialing_code {
    char *country;
    int code;
};


    int intl_code; // code for exit, in case the user wants to exit the program.

    const struct dialing_code country_codes[] = {
        {"Argentina", 54}, {"Bangladesh",        880},  {"Germany",    49},          {"India",
                    91},
        {"Indonesia", 62}, {"Russia",              7},  {"Brazil",     55},          {"Iran",
                    98},
        {"Ethiopia", 251}, {"France",             33},  {"China",      86},
        {"Colombia",              57},
        {"Italy",     39}, {"Congo,Dem.",        243}, {"Egypt",     20},         {"Japan",
                81},
        {"Mexico",     52}, {"Nigeria",          234}, {"Burma",    95},          {"Turkey",
            90},{"United States",     1},
```

Console Shell

```
clang-7 -pthread -lm -o main main.c
./main
Please input the international code: 90
The country is: Turkey
Please input the international code:
```