**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: Quynh-anh Nguyen**

**Campus ID: qnguyen51**

**Panther #: 002-38-5394**

**PART 1:**

1.

#include<stdio.h>

int main() {

char ch;

//This initializes ch as a char.

printf("Enter a password : ");

//This statement prompts the user to enter a password.

int count = 0;

//This will initialize the count int.

do {

ch = getchar();

count++;

//This do loop will read the characters in the password and store it in the count.

}

while(ch != '\n');

int score = -(10 - count + 1) \* 5;

printf("Score = %d\n", score);

if(score < -30)

//This will calculate the total score depending on the number of characters.

printf("The password is unsafe! Please reset.");

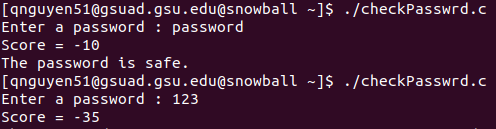
else

printf("The password is safe.\n");

//This will print whether the password is safe or not depending on the score.

return 0;

}



2.

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

int main() {

int score = 0, upperCount = 0, lowerCount =0, numCount = 0;

char ch;

//This initialize variables

printf("Enter a password :");

while ((ch = getchar()) != '\n') {

if(ch >= 'a' && ch <= 'z') {

lowerCount++;

}

//This if loop will check if there is a char within the alphabet that is lowercase.

if(ch >= 'A' && ch <= 'Z') {

upperCount++;

}

//This if loop will check if there is a uppercase letter from the alphabet.

if(ch >= '0' && ch <= '9') {

numCount++;

}

//This checks whether there is a number in the password.

}

if(lowerCount == 0) {

score += 20;

}

//This checks whether the password includes a lowercase letter, the score will add 20.

if(upperCount == 0) {

score += 20;

}

//This checks whether the password includes an uppercase letter, the score will add 20.

if(numCount == 0) {

score += 20;

}

printf("\nScore = %d\n",(-score));

//This will print the current score.

if(score > 30) {

printf("The password is unsafe! Please reset.\n");

}

else {

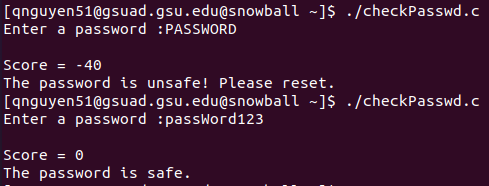
printf("The password is safe.\n");

}

//This will check if the score is higher than 30 and if not, it will have the user input a new password. Otherwise the program will print password is good.

return 0;

}



**PART 2:**

3. #include <string.h>

#include <stdio.h>

int main() {

char str[1000];

printf("Enter a string: ");

scanf("%s",str);

//THese statements will initialize the string the user inputs.

if(isPalin(str)) {

printf("Palindrome\n");

} else {

printf("Not a Palindrome\n");

}

//This uses the method created below to determine whether the string the user inputed is a palindrome and will return the respective answer.

return 0;

}

int isPalin(char str[]) {

int length = strlen(str);

int left = 0;

int right = length - 1;

char ch;

//These will initialize the string length, left right and chars to use in the method.

while(left < right) {

ch = str[left];

if(!((ch >= 'a' && ch <= 'z') || (ch >= 'A' && ch <= 'Z'))) {

left++;

continue;

}

ch = str[right];

if(!((ch >= 'a' && ch <= 'z') || (ch >= 'A' && ch <= 'Z'))) {

right--;

continue;

}

if(str[left] != str[right]) {

return 0;

}

left++;

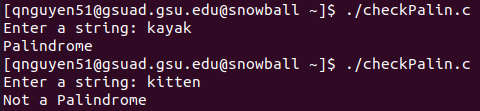
right--;

}

return 1;

//These loops will go through the string and compare the left character to the right character, if it matches it will continue through the string, if not, the loop will break and return not a palindrome.

}



**PART 3:**

5. #include <stdio.h>

struct dialCode {

char \*country;

int code;

};

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

int intl, i;

//This initializes the int intl.

const struct dialCode country\_codes[] = {

{"Afghanistan", 93}, {"Bahamas", 1-242},

{"Brazil", 55}, {"Bolivia", 591},

{"China", 86}, {"Canada", 1},

{"Costa Rica", 506}, {"Egypt", 20},

{"Ethiopia", 251}, {"France", 33},

{"Greece", 30}, {"India", 91},

{"Indonesia", 62}, {"Ireland", 353},

{"Italy", 39}, {"Japan", 81},

{"Mexico", 52}, {"Nigeria", 234},

{"Peru", 51}, {"Philippines", 63},

{"Portugal", 351}, {"Vietnam", 84}

};

//This stores the country codes to be used in the program.

int entries = sizeof(country\_codes) / sizeof(\*country\_codes);

do {

int found = 0;

printf("Enter international code (enter -1 to quit): ");

scanf("%d", &intl);

//This prompts the user to enter a number to look up.

if (intl == -1)

//If the user enters -1, the loop will break/end.

break;

for (i = 0; i < entries; i++) {

if (country\_codes[i].code == intl) {

printf("Country found: %s\n", country\_codes[i].country);

found = 1;

//This will allow the program to constantly loop until the user enters -1. This will match up the number the user entered with the country associated with the number.

}

}

if (!found)

printf("Country Code not found.\n");

} while(1);

return 0;

//This is what the program will print when the user enters a number that is not in the directory.

}

