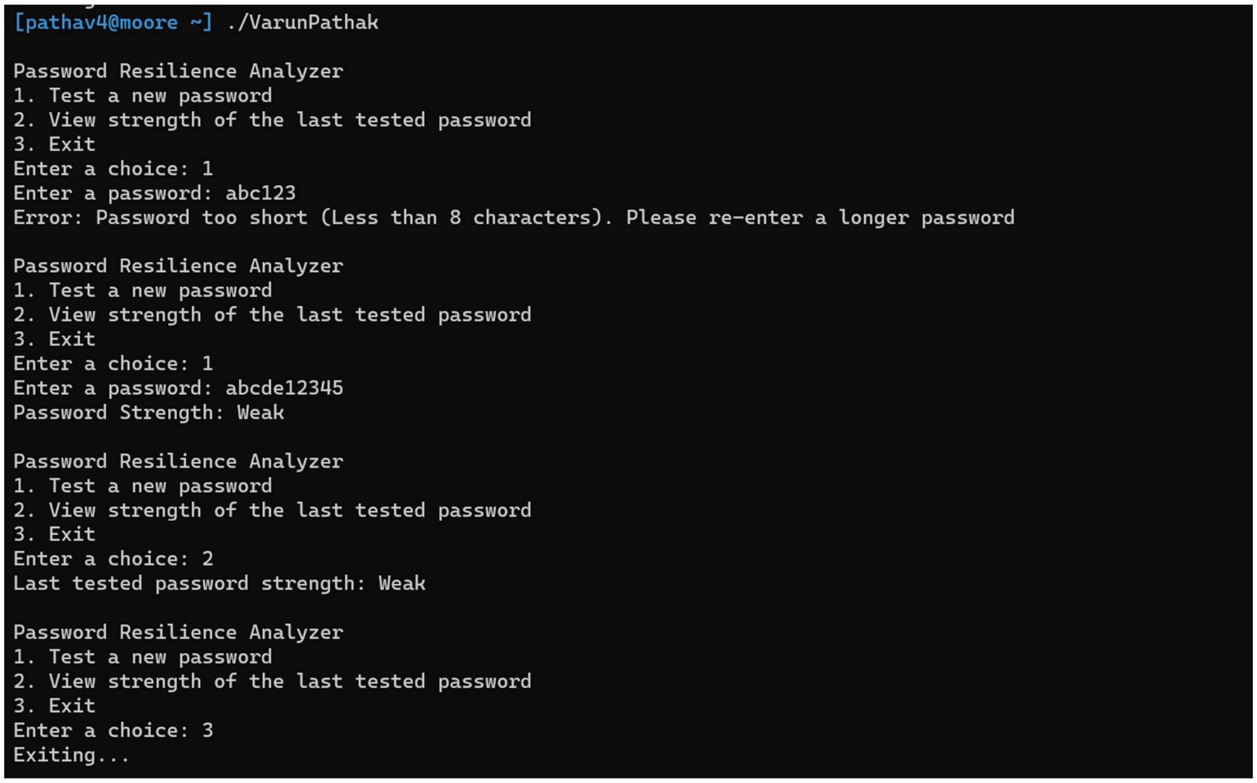
2XC3 FINAL C PROJECT L01– NOV 28

Submitted by (Name, MacID, St. #): Varun Pathak, pathav4, 400444566

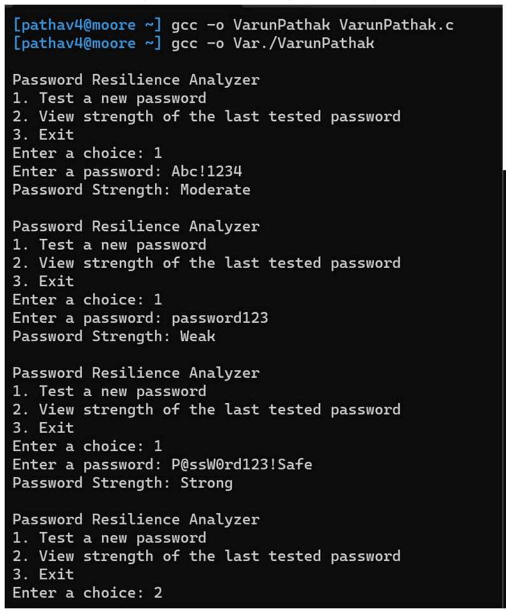
Accuracy/User Experience:

Test Case 1: Simple bounds, testing each of the three functions' versatility.

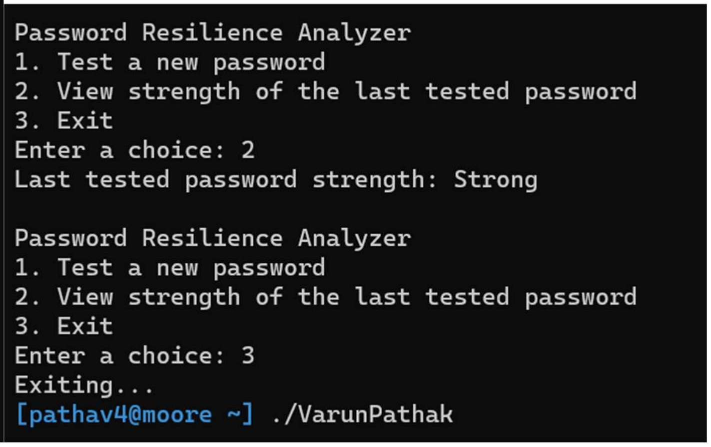


```
[pathav4@moore ~] ./VarunPathak

Password Resilience Analyzer
1. Test a new password
2. View strength of the last tested password
3. Exit
Enter a choice: 1
Enter a password: abc123
Error: Password too short (Less than 8 characters). Please re-enter a longer password

Password Resilience Analyzer
1. Test a new password
2. View strength of the last tested password
3. Exit
Enter a choice: 1
Enter a password: abcde12345
Password Strength: Weak

Password Resilience Analyzer
1. Test a new password
2. View strength of the last tested password
3. Exit
Enter a choice: 2
Last tested password strength: Weak

Password Resilience Analyzer
1. Test a new password
2. View strength of the last tested password
3. Exit
Enter a choice: 3
Exiting...
```

Test Case 2: Edge case testing, with special characters and repeating options



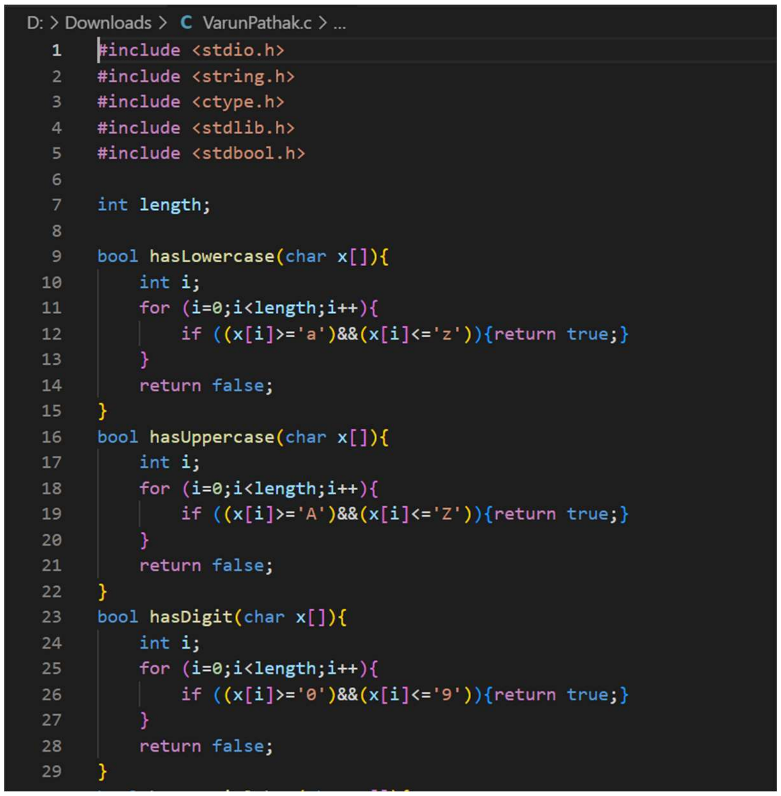
```
[pathav4@moore ~] gcc -o VarunPathak VarunPathak.c
[pathav4@moore ~] gcc -o Var./VarunPathak

Password Resilience Analyzer
1. Test a new password
2. View strength of the last tested password
3. Exit
Enter a choice: 1
Enter a password: Abc!1234
Password Strength: Moderate

Password Resilience Analyzer
1. Test a new password
2. View strength of the last tested password
3. Exit
Enter a choice: 1
Enter a password: password123
Password Strength: Weak

Password Resilience Analyzer
1. Test a new password
2. View strength of the last tested password
3. Exit
Enter a choice: 1
Enter a password: P@ssW0rd123!Safe
Password Strength: Strong

Password Resilience Analyzer
1. Test a new password
2. View strength of the last tested password
3. Exit
Enter a choice: 2
```



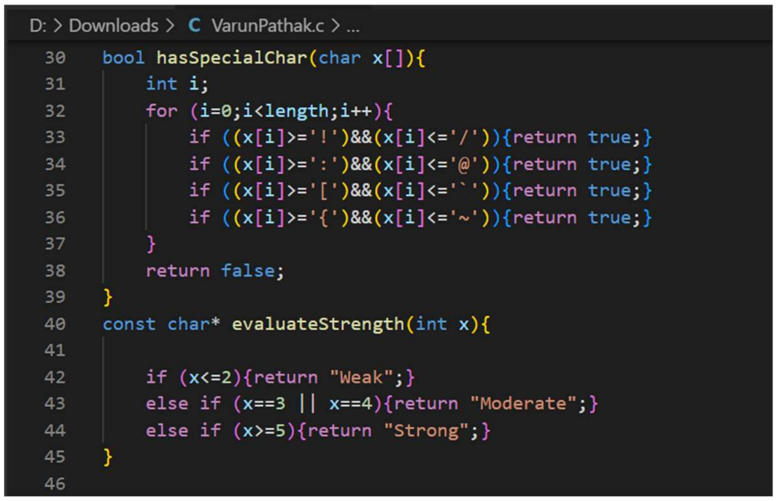
```
Password Resilience Analyzer
1. Test a new password
2. View strength of the last tested password
3. Exit
Enter a choice: 2
Last tested password strength: Strong

Password Resilience Analyzer
1. Test a new password
2. View strength of the last tested password
3. Exit
Enter a choice: 3
Exiting...
[pathav4@moore ~] ./VarunPathak
```

Code Organization:

Modularized functions:

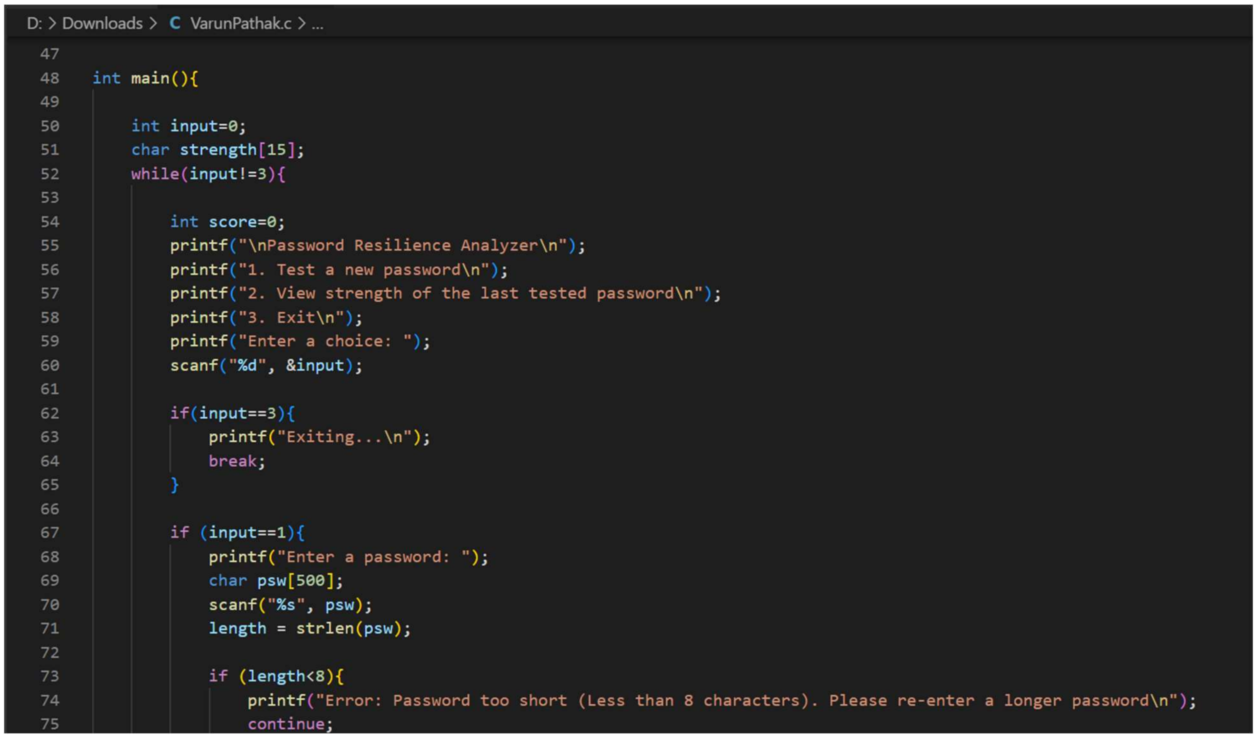


```c
#include <stdio.h>
#include <string.h>
#include <ctype.h>
#include <stdlib.h>
#include <stdbool.h>

int length;

bool hasLowercase(char x[]){
    int i;
    for (i=0;i<length;i++){
        if ((x[i]>='a')&&(x[i]<='z')){return true;}
    }
    return false;
}
bool hasUppercase(char x[]){
    int i;
    for (i=0;i<length;i++){
        if ((x[i]>='A')&&(x[i]<='Z')){return true;}
    }
    return false;
}
bool hasDigit(char x[]){
    int i;
    for (i=0;i<length;i++){
        if ((x[i]>='0')&&(x[i]<='9')){return true;}
    }
    return false;
}
```



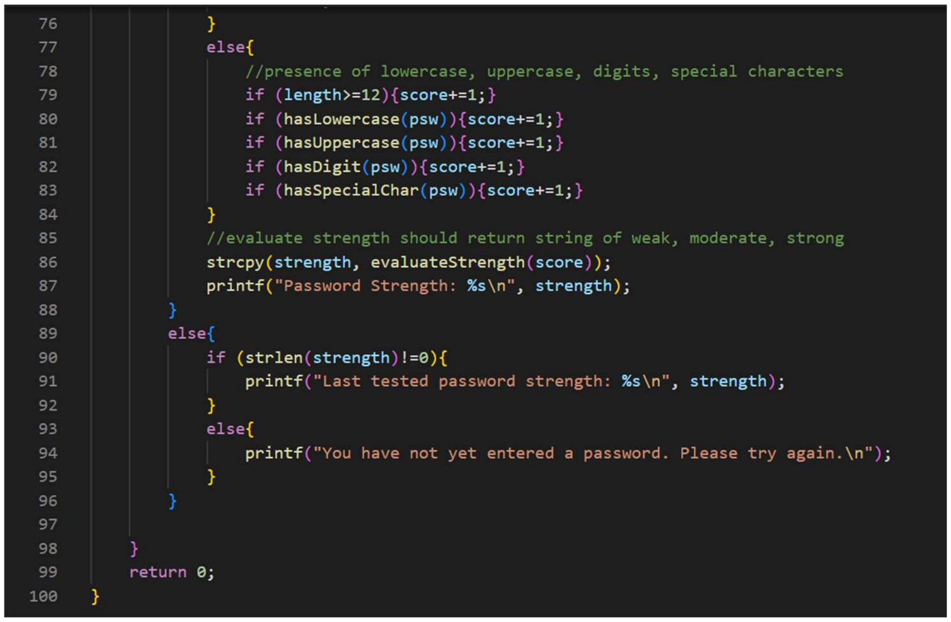
```c
bool hasSpecialChar(char x[]){
    int i;
    for (i=0;i<length;i++){
        if ((x[i]>='!')&&(x[i]<='/')){return true;}
        if ((x[i]>=':')&&(x[i]<='@')){return true;}
        if ((x[i]>='[')&&(x[i]<='`')){return true;}
        if ((x[i]>='{')&&(x[i]<='~')){return true;}
    }
    return false;
}
const char* evaluateStrength(int x){

    if (x<=2){return "Weak";}
    else if (x==3 || x==4){return "Moderate";}
    else if (x>=5){return "Strong";}
}

```

Main Function:



D: > Downloads > C VarunPathak.c > ...

```c

int main(){

    int input=0;
    char strength[15];
    while(input!=3){

        int score=0;
        printf("\nPassword Resilience Analyzer\n");
        printf("1. Test a new password\n");
        printf("2. View strength of the last tested password\n");
        printf("3. Exit\n");
        printf("Enter a choice: ");
        scanf("%d", &input);

        if(input==3){
            printf("Exiting...\n");
            break;
        }

        if (input==1){
            printf("Enter a password: ");
            char psw[500];
            scanf("%s", psw);
            length = strlen(psw);

            if (length<8){
                printf("Error: Password too short (Less than 8 characters). Please re-enter a longer password\n");
                continue;
```



```c
            }
            else{
                //presence of lowercase, uppercase, digits, special characters
                if (length>=12){score+=1;}
                if (hasLowercase(psw)){score+=1;}
                if (hasUppercase(psw)){score+=1;}
                if (hasDigit(psw)){score+=1;}
                if (hasSpecialChar(psw)){score+=1;}
            }
            //evaluate strength should return string of weak, moderate, strong
            strcpy(strength, evaluateStrength(score));
            printf("Password Strength: %s\n", strength);
        }
        else{
            if (strlen(strength)!=0){
                printf("Last tested password strength: %s\n", strength);
            }
            else{
                printf("You have not yet entered a password. Please try again.\n");
            }
        }

    }
    return 0;
}
```