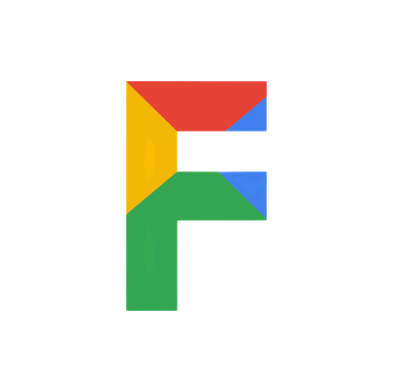
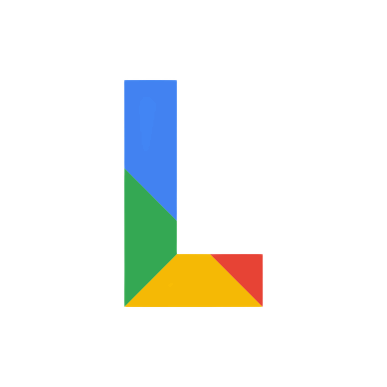
{   };



Course: CL1002 – Programming Fundamentals.

Instructor: Sir Muhammad Usman.

Submitted by: Muhammad Rehan

Roll no. 22P-9106

Class: BSE-1A (Fall 2022)

Lab Task no. 11

Date: November 30th, 2022.

Department of Computer Science

Sir, I know I updated it late. I forgot to upload it on time. I know, my bad 😐.

Problem 01:

Code:

Sir, this Code works on all kinds of strings with no tolerance or chance of error.

#include <stdio.h>

#include <string.h>

*void* palindrome(*char* *name*[]);

*char* reserve();

*int* main()

{

*char* name[50];

    puts("Enter the name:");

    scanf("%s",name);

    palindrome(name);

    return 0;

}

*void* palindrome(*char* *name*[])

{

*int* equal;

*int* size = strlen(name);

*char* reserve[50];

    for (*int* i = 0; i < size; i++)

    {

        reserve[i] = name[(size - i) - 1];

    }

    printf("\nThe reserve of the name is :%s", reserve);

    equal=strcmp(name, reserve);

    if (equal == 0)

    {

        puts("\nThe name is palindrome\n");

    }

    else

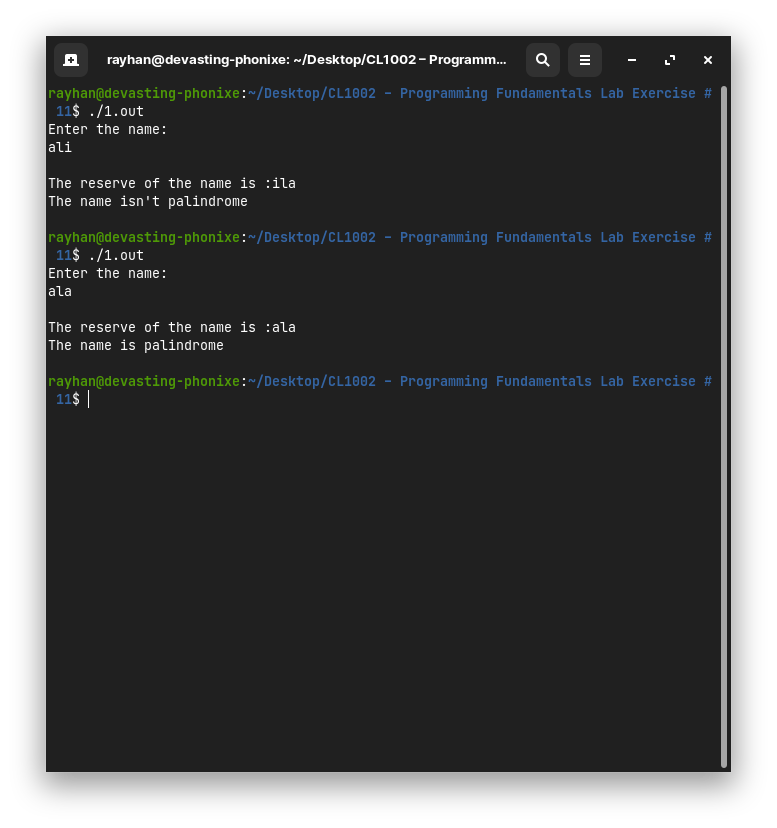
    {

        puts("\nThe name isn't palindrome\n");

    }

}

Screenshots:



Problem 02:

Code:

Sir, This Code checks everything and even tells you if no character is repeated by showing the message, "No character is frequent in the string.".The special thing is that while joining two strings, it shows you the resultant string with a space between them.

//Code Created by Muhammad Rehan, 22P-9106, BSSE 1A,

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

*void* operations(*char* *string*[], *int* *size*);

*void* vowel(*char* *string*[], *int* *size*);

*void* vandc(*char* *string*[], *int* *size*);

*void* Frechar(*char* *string*[], *int* *size*);

*void* Cont(*char* *string*[], *int* *size*);

*int* main()

{

*char* string[50];

    puts("Enter the sting:");

    scanf("%s", string);

*int* size = strlen(string);

    operations(string, size);

    return 0;

}

*void* operations(*char* *string*[], *int* *size*)

{

    printf("\nEnter the option:\nA) Count the number of vowels in the string\nB) Count both the vowels and consonants in the string.\nC) Display the most frequent character in the string.\nD) Concatenate another string with the existing string.\nE) Exit the program.");

*char* option;

    printf("\nEnter option:");

    scanf("\n%c", &option);

    switch (option)

    {

    case 'A':

        vowel(string, size);

        break;

    case 'a':

        vowel(string, size);

        break;

    case 'B':

        vandc(string, size);

        break;

    case 'b':

        vandc(string, size);

        break;

    case 'C':

        Frechar(string, size);

        break;

    case 'c':

        Frechar(string, size);

        break;

    case 'D':

        Cont(string, size);

        break;

    case 'd':

        Cont(string, size);

        break;

    case 'E':

        exit(1);

        break;

    case 'e':

        exit(1);

        break;

    default:

        puts("\nEnter a valid input.\n");

        break;

    }}

*void* vowel(*char* *string*[], *int* *size*)

{

*int* count = 0;

*char* vow[] = {'a', 'e', 'i', 'o', 'u', 'A', 'E', 'I', 'O', 'U'};

    for (*int* i = 0; i < size; i++)

    {

        for (*int* j = 0; j < 11; j++)

        {

            if (string[i] == vow[j])

                count++;

        }}

    printf("\nThe count of vowels in the string is %d\n", count);

    operations(string, size);

}

*void* vandc(*char* *string*[], *int* *size*)

{

*int* count = 0;

*char* vow[] = {'a', 'e', 'i', 'o', 'u', 'A', 'E', 'I', 'O', 'U'};

    for (*int* i = 0; i < size; i++)

    {

        for (*int* j = 0; j < 11; j++)

        {

            if (string[i] == vow[j])

                count++;

        }

    }

    printf("\nThe vowels are %d and the consonants are %d\n", count, size - count);

    operations(string, size);

}

*void* Frechar(*char* *string*[], *int* *size*)

{

*char* character = string[0];

*int* sum = 0, count = 0;

    for (*int* i = 0; i < size; i++)

    {

        count = 0;

        for (*int* j = 0; j < size; j++)

        {

            if (string[i] == string[j])

            {

                count++;

                if (count > sum)

                {

                    sum = count;

                    character = string[i];

                }

            }}

    }

    if (count == 1)

    {

        printf("\nThe frequency of all characters are same.\n");

    }

    else

    {

        printf("\nThe Most Frequent Charcter is %c\n", character);

    }

    operations(string, size);

}

*void* Cont(*char* *string*[], *int* *size*)

{

*char* co[50];

    puts("Enter you string:");

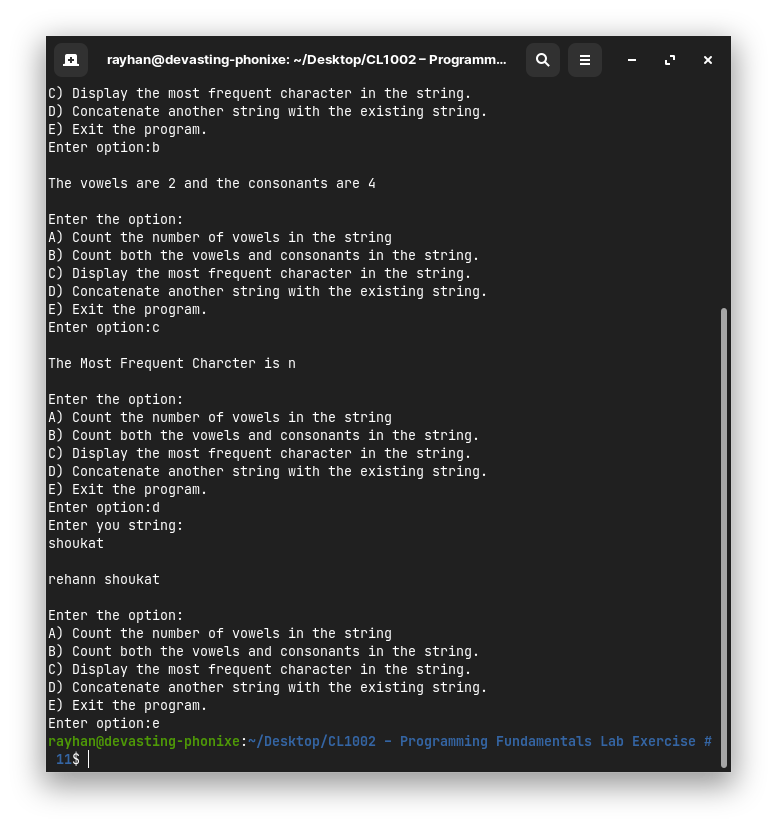
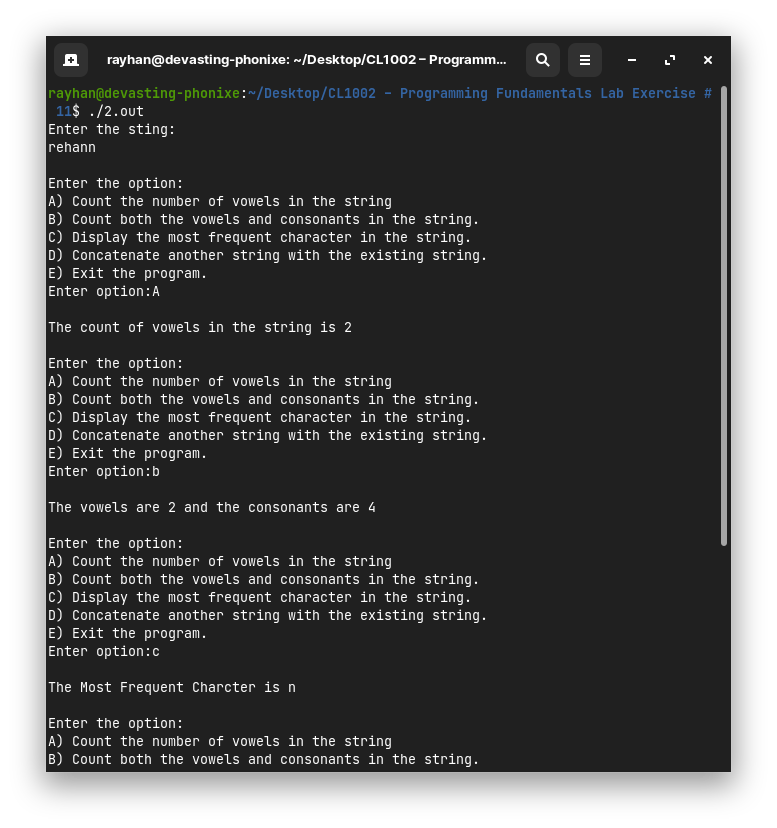
    scanf("%50s", co);

    printf("\n%s \n",strcat(strcat(string," "),co));

    operations(string, size);

}

Screenshot:



Problem 03:

Code:

Code works on any exception. Whether the alphabets are upper or lower case or the string is mixed of upper and lower alphabets, it will cipher them, even maintaining the spaces between them.

// Code Written by Muhammad Rehan 22P-9106 BSE 1A

// This Code run on smaller letter and

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

#include <ctype.h>

*void* operation(*char* *foo*[], *char* *option*);

*void* encrypt(*char* *foo*[]);

*void* decrypt(*char* *foo*[]);

*int* main()

{

*char* foo[50];

    puts("Enter the secret message:");

    fgets(foo, sizeof(foo), stdin);

    puts("Enter option:\nA) Encrpytion of the message.\nB) Decrption of the meesege.");

*char* option;

    scanf("%c", &option);

    operation(foo, option);

    return 0;

}

*void* operation(*char* *foo*[], *char* *option*)

{

    switch (option)

    {

    case 'A':

        encrypt(foo);

        break;

    case 'a':

        encrypt(foo);

        break;

    case 'B':

        decrypt(foo);

        break;

    case 'b':

        decrypt(foo);

        break;

    default:

        break;

    }

}

*void* encrypt(*char* *foo*[])

{

*int* i;

*int* len = strlen(foo);

*char* encrypt[len];

    for (i = 0; i < len - 1; i++)

    {

*int* spacechecker = isspace(foo[i]);

        if (spacechecker != 0)

        {

            encrypt[i] = ' ';

        }

        else

        {

            encrypt[i] = foo[i] + 3;

        }

    }

    for (*int* j = 0; j < len; j++)

    {

*int* spacechecker = isspace(encrypt[j]);

        if (encrypt[0] >= 65 && encrypt[len - 2] <= 96)

        {

            if (encrypt[j] >= 90)

            {

                if (spacechecker != 0)

                {

                    encrypt[j] = ' ';

                }

                else

                {

                    encrypt[j] = encrypt[j] - 26;

                }

            }

        }

        else

        {

            if (encrypt[j] >= 122)

            {

                if (spacechecker != 0)

                {

                    encrypt[j] = ' ';

                }

                else

                {

                    encrypt[j] = encrypt[j] - 26;

                }

            }

        }

    }

    printf("%s", encrypt);

}

*void* decrypt(*char* *foo*[])

{

*int* i;

*int* len = strlen(foo);

*char* dencrypt[len];

    for (i = 0; i < len - 1; i++)

    {

*int* spacechecker = isspace(foo[i]);

        if (spacechecker != 0)

        {

            dencrypt[i] = ' ';

        }

        else

        {

            dencrypt[i] = foo[i] - 3;

        }

    }

    for (*int* j = 0; j < len-1; j++)

    {

*int* spacechecker = isspace(dencrypt[j]);

        if (dencrypt[0] >= 65 && dencrypt[len - 2] <= 90)

        {

            if (dencrypt[j] <= 65)

            {

                if (spacechecker != 0)

                {

                    dencrypt[j] = ' ';

                }

                else

                {

                    dencrypt[j] = dencrypt[j] + 26;

                }

            }

        }

        else

        {

            if (dencrypt[j] <= 97)

            {

                if (spacechecker != 0)

                {

                    dencrypt[j] = ' ';

                }

                else

                {

                    dencrypt[j] = dencrypt[j] + 26;

                }

            }

        }

    }

    printf("%s", dencrypt);

}

Screenshot:

