***Assignment 1 || String***

Arjun Patel – FRN\_006

Q1) Write a program to scan string from user then scan a single character and search it in a accepted string.

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

char \*findChar(char \*str, char c)

{

    while (\*str != '\0')

    {

        if (\*str == c) return str;

        str++;

    }

    return 0;

}

int main()

{

    char\* str = (char\*)(malloc(sizeof(char)\*30));

    char c;

    printf("Enter string\n");

    // fgets(str, sizeof(str), stdin);

    scanf("%s", str);

    printf("Enter a charracter you want to find\n");

    fflush(stdin);

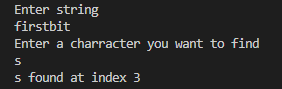
    scanf("%c", &c);

    char \*ptr = findChar(str, c);

    ptr ? printf("%c found at index %d\n", c, ptr - str) : printf("%c not found!!\n", c);

    return 0;

}



Q2 WAP Replace all Occurrences of ‘a’ with $ in a String

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

char \*replaceA(char \*str)

{

    char\* ptr = (char\*)(malloc(sizeof(char)\*strlen(str)));

    char\* temp = ptr;

    while (\*str != '\0')

    {

        if(\*str=='a') \*ptr = '$';

        else \*ptr = \*str;

        ptr++;

        str++;

    }

    \*ptr = '\0';

    return temp;

}

int main()

{

    char str[30];

    // char\* str = (char\*)(malloc(sizeof(char)\*30));

    char c;

    printf("Enter string\n");

    // fgets(str, sizeof(str), stdin);

    scanf("%s", str);

    printf("%s",replaceA(str));

    return 0;

}



Q3) WAP to Remove the nth Index Character from a Non-Empty String

#include<stdio.h>

#include<string.h>

char\* removeNthChar(char\* str, int n){

    int length = strlen(str), i;

    for (i = 0; i < length-1; i++)

    {

        if(i>=n) str[i]=str[i+1];

    }

    str[i]='\0';

    return str;

}

int main(){

    char str[30];

    printf("Enter string\n");

    scanf("%s", str);

    int n;

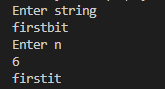
    printf("Enter n\n");

    scanf("%d",&n);

    printf("%s",removeNthChar(str, n-1));

    return 0;

}



Q4) WAP to Form a New String where the First Character and the Last Character have been Exchanged.

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

char \*exchangeFirstNLast(char \*str)

{

    char\* ptr = (char\*)(malloc(sizeof(char)\*strlen(str)));

    int len = strlen(str), i;

    for (i = 0; i < len; i++)

    {

        if(i==0) ptr[len-1] = str[i];

        else if(i==len-1) ptr[0] = str[i];

        else ptr[i] = str[i];

    }

    ptr[i]='\0';

    return ptr;

}

int main()

{

    char str[30];

    printf("Enter string\n");

    scanf("%s", str);

    printf("%s",exchangeFirstNLast(str));

    return 0;

}



Q5) WAP to Count the Number of Vowels in a String

#include <stdio.h>

#include <string.h>

int countVowels(char \*str)

{

    int count = 0;

    while (\*str != '\0')

    {

        if (\*str == 'a' || \*str == 'e' || \*str == 'i' || \*str == 'o' || \*str == 'u' || \*str == 'A' || \*str == 'E' || \*str == 'I' || \*str == 'O' || \*str == 'U')

        {

            count++;

        }

        str++;

    }

    return count;

}

int main()

{

    char str[30];

    printf("Enter string\n");

    scanf("%s", str);

    printf("No of vowels %d\n", countVowels(str));

    return 0;

}



Q6) WAP to Take in a String and Replace Every Blank Space with special symbol.

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

char \*replaceWhiteSpace(char \*str)

{

    char\* ptr = (char\*)(malloc(sizeof(char)\*strlen(str)));

    char\* temp = ptr;

    while (\*str != '\0')

    {

        if(\*str==' ') \*ptr = '\_';

        else \*ptr = \*str;

        ptr++;

        str++;

    }

    \*ptr = '\0';

    return temp;

}

int main()

{

    char str[30];

    printf("Enter string\n");

    fgets(str, sizeof(str), stdin);

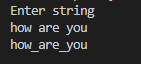
    // scanf("%s", str);

    printf("%s\n",replaceWhiteSpace(str));

    //heap should be free after use

    return 0;

}



Q7) WAP to Remove the Characters of Odd Index Values in a String

#include<stdio.h>

#include<string.h>

char\* removeOddIndex(char\* str){

    int length = strlen(str), j=0;

    for (int i = 0; i < length-1; i++)

    {

        if(i%2==0){

            str[j] = str[i];

            j++;

        }

    }

    str[length/2] = '\0';

    return str;

}

int main(){

    char str[30];

    printf("Enter string\n");

    fgets(str, sizeof(str), stdin);

    // scanf("%s", str);

    printf("%s",removeOddIndex(str));

    return 0;

}



Q) WAP to Calculate the Number of Words Present in a String

#include<stdio.h>

#include<string.h>

int countWords(char\* str){

    int count=0, length = strlen(str)-1;

    // printf("\n%d\n", length);

    int start=0, end = length-1;

    for (int i = 0; i < length; i++)

    {

        if(str[i]!=' ') break;

        start++;

    }

    for(int i=length-1; i>=0; i--){

        if(str[i]!=' ') break;

        end--;

    }

    // printf("start-> %d, end-> %d\n", start, end);

    for (int i = start; i<end; i++)

    {

        if(str[i]==' ' && str[i+1]!=' '){

            count++;

        }

    }

    if(!count) return count;

    return ++count;

}

int main(){

    char str[30];

    printf("Enter Sentence\n");

    fgets(str, sizeof(str), stdin);

    printf("%d words in sentence\n", countWords(str));

    return 0;

}



Q9) WAP to Take in Two Strings and Display the Larger String without Using Built-in Functions

#include<stdio.h>

#include<string.h>

int cmpstr(char\* str, char\* str2){

    int len1 = strlen(str);

    int len2 = strlen(str2);

    if(len1>len2) return 1;

    else if(len2>len1) return -1;

    else {

        for (int i = 0; i < len1;  i++)

        {

            if(str[i]>str2[i]) return 1;

            if(str2[i]>str[i]) return -1;

        }

        return 0;

    }

}

int main(){

    char str[30];

    printf("Enter String 1\n");

    scanf("%s", str);

    char str2[30];

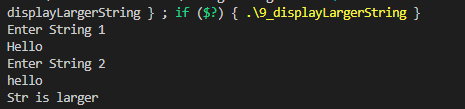
    printf("Enter String 2\n");

    scanf("%s", str2);

    cmpstr(str, str2) ? printf("Str is larger\n"): printf("Str2 is larger\n");

    return 0;

}



Q10) Write a program to check the string is palindrome or not.

#include<stdio.h>

#include<string.h>

#include<stdlib.h>

char\* checkPalindrome(char \*str){

    int length = strlen(str), i;

    char\* dup = (char\*)malloc(length\* sizeof(char));

    for ( i = 0; i < length; i++)

    {

        dup[i] = str[length-i-1];

    }

    dup[i] = '\0';

    return dup;

}

int main(){

    char str[30];

    printf("Enter String 1\n");

    scanf("%s", str);

    char\* dup = checkPalindrome(str);

    !strcmp(str, dup) ? printf("Palindrome\n") : printf("Not Palindrome\n");

    free(dup);

    return 0;

}



***------END------***