**(2347207) ALWIN TOMY**

1.Write a program to print N equal parts of a given string.

#include <stdio.h>

#include <string.h>

int main() {

char str[100];

int n, len, part\_size, i;

printf("Enter a string: ");

gets(str);

printf("Enter the number of parts: ");

scanf("%d", &n);

len = strlen(str);

part\_size = len / n;

if (len % n != 0) {

printf("Cannot divide the string into %d equal parts.\n", n);

} else {

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

if (i % part\_size == 0) {

printf("\n");

}

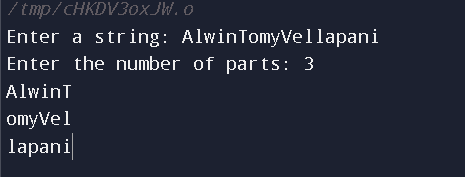
printf("%c", str[i]);

}

}

return 0;

}



2.Write a C Program to insert characters in a string at a certain position

#include <stdio.h>

#include <string.h>

int main() {

char str[100], ch;

int pos, len, i;

printf("Enter a string: ");

gets(str);

printf("Enter the character to insert: ");

scanf("%c", &ch);

printf("Enter the position to insert: ");

scanf("%d", &pos);

len = strlen(str);

if (pos > len) {

printf("Invalid position.\n");

} else {

for (i = len; i >= pos; i--) {

str[i+1] = str[i];

}

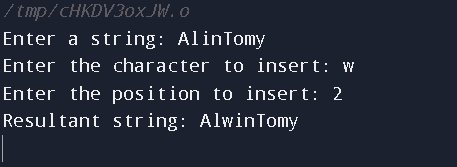
str[pos] = ch;

printf("Resultant string: %s\n", str);

}

return 0;

}



3. Write a C Program to implement Anagram

#include <stdio.h>

#include <string.h>

int main() {

char str1[100], str2[100];

int len1, len2, i, j, found = 0;

printf("Enter the first string: ");

gets(str1);

printf("Enter the second string: ");

gets(str2);

len1 = strlen(str1);

len2 = strlen(str2);

if (len1 != len2) {

printf("Strings are not anagram.\n");

} else {

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

found = 0;

for (j = 0; j < len2; j++) {

if (str1[i] == str2[j]) {

found = 1;

break;

}

}

if (found == 0) {

printf("Strings are not anagram.\n");

break;

}

}

if (found == 1) {

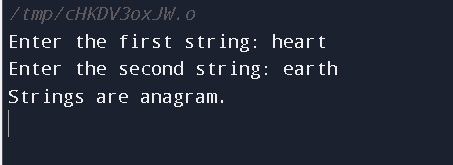
printf("Strings are anagram.\n");

}

}

return 0;

}



4. Write a program in C to remove characters from a string except alphabets.

#include <stdio.h>

#include <string.h>

void remove\_non\_alphabetic\_characters(char \*str) {

int i, j;

for (i = 0; str[i] != '\0'; i++) {

if (!isalpha(str[i])) {

for (j = i; str[j] != '\0'; j++) {

str[j] = str[j + 1];

}

str[j] = '\0';

}

}

}

int main() {

char str[100];

printf("Enter a string: ");

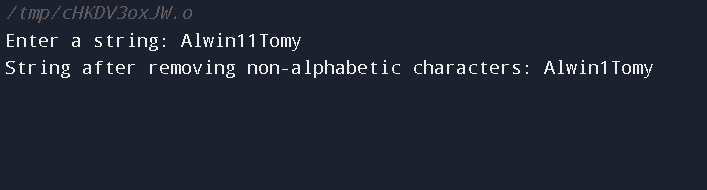
gets(str);

remove\_non\_alphabetic\_characters(str);

printf("String after removing non-alphabetic characters: %s\n", str);

return 0;

}



5. Write a program in C to find the frequency of characters.

#include <stdio.h>

#include <string.h>

void find\_character\_frequency(char \*str) {

int i, j;

int frequency[256] = {0};

for (i = 0; str[i] != '\0'; i++) {

frequency[str[i]]++;

}

for (i = 0; i < 256; i++) {

if (frequency[i] > 0) {

printf("%c: %d\n", i, frequency[i]);

}

}

}

int main() {

char str[100];

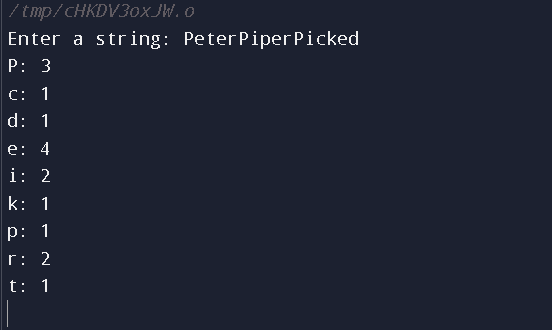
printf("Enter a string: ");

gets(str);

find\_character\_frequency(str);

return 0;

}



6. Write a program in C to check whether a character is a Hexadecimal Digit

or not.

#include <stdio.h>

#include <ctype.h>

int is\_hexadecimal\_digit(char c) {

return isdigit(c) || (c >= 'a' && c <= 'f') || (c >= 'A' && c <= 'F');

}

int main() {

char c;

printf("Enter a character: ");

scanf("%c", &c);

if (is\_hexadecimal\_digit(c)) {

printf("The character '%c' is a hexadecimal digit.\n", c);

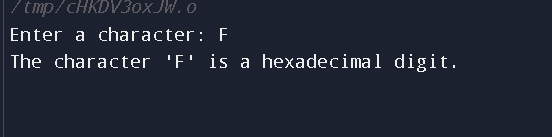
} else {

printf("The character '%c' is not a hexadecimal digit.\n", c);

}

return 0;

}



7. Write a program in C to replace the spaces in a string with a specific

character.

#include <stdio.h>

#include <string.h>

void replace\_spaces(char \*str, char new\_char) {

int i, len;

len = strlen(str);

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

if (str[i] == ' ') {

str[i] = new\_char;

}

}

}

int main() {

char str[100];

char new\_char;

printf("Enter a string: ");

gets(str);

printf("Enter the character to replace spaces with: ");

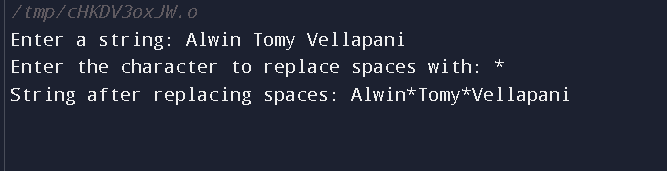
scanf("%c", &new\_char);

replace\_spaces(str, new\_char);

printf("String after replacing spaces: %s\n", str);

return 0;

}



8.

Write a program in C to split strings by space into words.

#include <stdio.h>

#include <string.h>

int main() {

char str[100], word[20][20];

int i, j = 0, k = 0;

printf("Enter a string: ");

gets(str);

for (i = 0; str[i] != '\0'; i++) {

if (str[i] == ' ') {

word[j][k] = '\0';

j++;

k = 0;

} else {

word[j][k] = str[i];

k++;

} }

word[j][k] = '\0';

printf("Words in the string:\n");

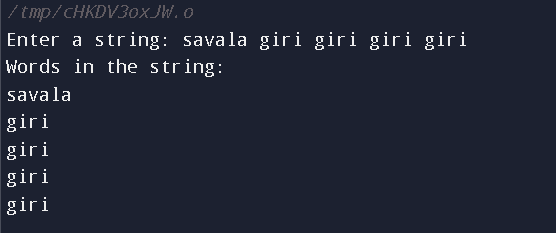
for (i = 0; i <= j; i++) {

printf("%s\n", word[i]);

}

return 0;

}



9. Write a C program to reverse all the vowels present in a given string.

Return the newly created string

#include <stdio.h>

#include <string.h>

int is\_vowel(char ch) {

if (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u' ||

ch == 'A' || ch == 'E' || ch == 'I' || ch == 'O' || ch == 'U') {

return 1;

} else {

return 0;

}

}

int main() {

char str[100], new\_str[100];

int len, i, j = 0;

printf("Enter a string: ");

gets(str);

len = strlen(str);

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

if (is\_vowel(str[i])) {

new\_str[j] = str[i];

j++;

}

}

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

if (is\_vowel(str[i])) {

j--;

str[i] = new\_str[j];

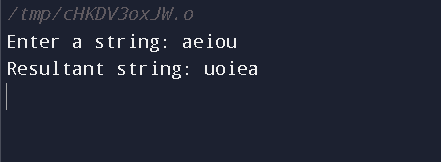
}

}

printf("Resultant string: %s\n", str);

return 0;

}



10. Write a C program to find the longest palindromic substring from a given

string. Return the substring.

#include <stdio.h>

#include <string.h>

int main() {

char str[100], substr[100];

int len, i, j, k, max\_len = 0;

printf("Enter a string: ");

gets(str);

len = strlen(str);

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

for (j = i; j < len; j++) {

int is\_palindrome = 1;

for (k = i; k <= j; k++) {

if (str[k] != str[i+j-k]) {

is\_palindrome = 0;

break;

}

}

if (is\_palindrome && j-i+1 > max\_len) {

max\_len = j-i+1;

strncpy(substr, &str[i], max\_len);

substr[max\_len] = '\0';

}

}

}

printf("Longest palindromic substring: %s\n", substr);

return 0;

}

