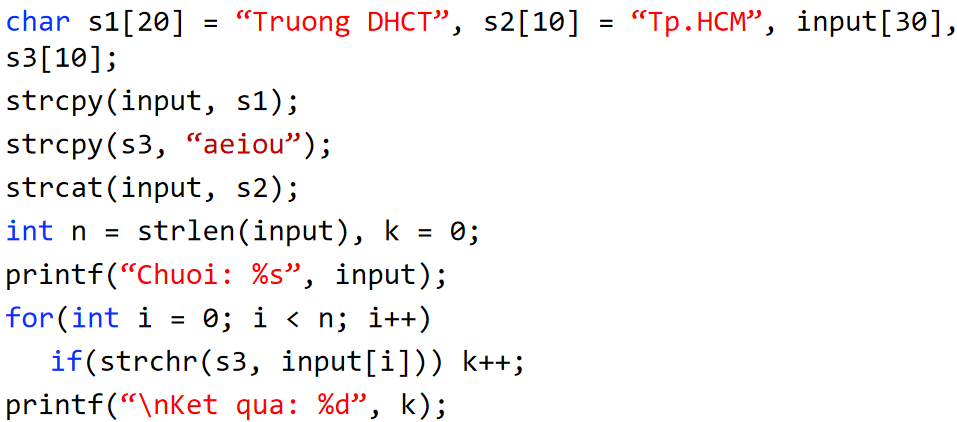
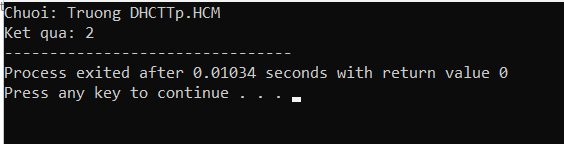
BÀI TẬP CHƯƠNG 3 CHUỖI KÝ TỰ

*Note\*: Em đã làm các bài từ bai1- bai10 bao gồm source code và kết quả em đã tổng hợp và viết vào file word để trình bài em đảm bảo tất cả các bài đều do chính em làm và em cũng có gửi cho thầy các file code và đảm bảo nó chạy được và hoàn chỉnh theo yêu cầu đặt ra.*

1. Cho biết kết quả của đọan chương trình sau:



Kết quả:



1. Viết hàm đếm có bao nhiêu khoảng trắng trong chuỗi.

Code Bai2:

#include <stdio.h>

#include <string.h>

// Hàm đếm số lượng khoảng trắng trong chuỗi

int countSpaces(const char \*str) {

int count = 0;

int length = strlen(str);

// Duyệt qua từng ký tự trong chuỗi

for (int i = 0; i < length; i++) {

// Kiểm tra nếu ký tự là khoảng trắng

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

count++;

}

}

return count;

}

int main() {

// Khai báo chuỗi cần kiểm tra

char input[] = "This is a test string with some spaces.";

// Gọi hàm và in kết quả

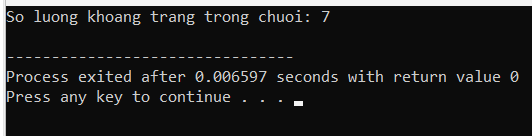
int numberOfSpaces = countSpaces(input);

printf("So luong khoang trang trong chuoi: %d\n", numberOfSpaces);

return 0;

}

Kết quả:



1. Nhập vào một chuỗi, xóa những khoảng trắng thừa trong chuỗi.

#include <stdio.h>

#include <string.h>

#include <ctype.h>

// Hàm xóa khoảng trắng thừa trong chuỗi

void removeExtraSpaces(char \*str) {

int length = strlen(str);

char result[length + 1]; // Tạo một chuỗi kết quả

int index = 0; // Chỉ số cho chuỗi kết quả

// Bỏ qua các khoảng trắng ở đầu chuỗi

int start = 0;

while (str[start] == ' ') {

start++;

}

// Duyệt qua từng ký tự trong chuỗi

for (int i = start; i < length; i++) {

// Nếu ký tự không phải là khoảng trắng, thêm vào chuỗi kết quả

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

result[index++] = str[i];

}

// Nếu ký tự là khoảng trắng và ký tự trước đó không phải là khoảng trắng

else if (i > 0 && str[i - 1] != ' ') {

result[index++] = ' ';

}

}

// Loại bỏ khoảng trắng cuối chuỗi

if (index > 0 && result[index - 1] == ' ') {

index--;

}

// Thêm ký tự kết thúc chuỗi

result[index] = '\0';

// Sao chép kết quả vào chuỗi gốc

strcpy(str, result);

}

int main() {

char input[100];

// Nhập chuỗi từ người dùng

printf("Nhap vao mot chuoi: ");

fgets(input, sizeof(input), stdin);

// Xóa ký tự xuống dòng (nếu có) ở cuối chuỗi do fgets thêm vào

size\_t len = strlen(input);

if (len > 0 && input[len - 1] == '\n') {

input[len - 1] = '\0';

}

// Gọi hàm xóa khoảng trắng thừa

removeExtraSpaces(input);

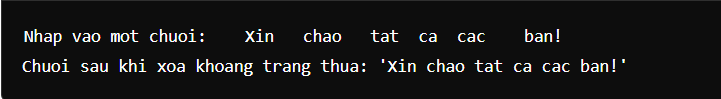
// In chuỗi sau khi xóa khoảng trắng thừa

printf("Chuoi sau khi xoa khoang trang thua: '%s'\n", input);

return 0;

}

Kết quả:



1. Viết hàm đổi những kí tự đầu tiên của mỗi từ thành chữ in hoa và những từ không phải đầu câu sang chữ thường

Code Bai4:

#include <stdio.h>

#include <ctype.h>

#include <string.h>

// Hàm đổi chữ cái đầu của mỗi từ thành chữ in hoa

void capitalizeWords(char \*str) {

int length = strlen(str);

int inWord = 0; // Biến để kiểm tra trạng thái đang trong một từ

for (int i = 0; i < length; i++) {

if (isalpha(str[i])) {

if (inWord == 0) {

// Nếu ký tự là chữ cái đầu tiên của từ, chuyển sang chữ in hoa

str[i] = toupper(str[i]);

inWord = 1;

} else {

// Nếu là ký tự giữa hoặc cuối từ, chuyển sang chữ thường

str[i] = tolower(str[i]);

}

} else {

// Nếu không phải chữ cái, đánh dấu là khoảng trắng hoặc ký tự đặc biệt

inWord = 0;

}

}

}

int main() {

char input[100];

// Nhập chuỗi từ người dùng

printf("Nhap vao mot chuoi: ");

fgets(input, sizeof(input), stdin);

// Xóa ký tự xuống dòng (nếu có) ở cuối chuỗi do fgets thêm vào

size\_t len = strlen(input);

if (len > 0 && input[len - 1] == '\n') {

input[len - 1] = '\0';

}

// Gọi hàm đổi chữ cái đầu mỗi từ thành in hoa

capitalizeWords(input);

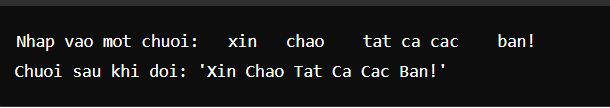
// In chuỗi sau khi đã đổi

printf("Chuoi sau khi doi: '%s'\n", input);

return 0;

}

Kết quả:



1. Viết chương trình tìm kiếm tên trong chuỗi họ tên. Nếu có thì xuất ra là tên này đã nhập đúng, ngược lại thông báo là đã nhập sai.

Code Bai5:

#include <stdio.h>

#include <string.h>

#include <ctype.h>

// Hàm chuyển đổi chuỗi thành chữ thường

void toLowerCase(char \*str) {

for (int i = 0; str[i]; i++) {

str[i] = tolower(str[i]);

}

}

// Hàm tìm kiếm tên trong chuỗi họ tên (không phân biệt chữ hoa, chữ thường)

int searchName(const char \*fullName, const char \*name) {

char lowerFullName[100];

char lowerName[50];

// Sao chép và chuyển đổi chuỗi họ tên và tên cần tìm thành chữ thường

strcpy(lowerFullName, fullName);

strcpy(lowerName, name);

toLowerCase(lowerFullName);

toLowerCase(lowerName);

// Sử dụng hàm strstr để tìm tên trong chuỗi họ tên

char \*position = strstr(lowerFullName, lowerName);

// Kiểm tra nếu tìm thấy và ký tự liền trước hoặc sau nó không phải chữ cái

while (position != NULL) {

// Kiểm tra xem tên có phải là một từ riêng biệt không

if ((position == lowerFullName || !isalpha(\*(position - 1))) &&

(\*(position + strlen(lowerName)) == '\0' || !isalpha(\*(position + strlen(lowerName))))) {

return 1; // Tìm thấy tên

}

// Tìm vị trí tiếp theo của tên trong chuỗi

position = strstr(position + 1, lowerName);

}

return 0; // Không tìm thấy tên

}

int main() {

char fullName[100];

char name[50];

// Nhập chuỗi họ tên từ người dùng

printf("Nhap vao ho ten: ");

fgets(fullName, sizeof(fullName), stdin);

// Xóa ký tự xuống dòng (nếu có) ở cuối chuỗi do fgets thêm vào

size\_t len = strlen(fullName);

if (len > 0 && fullName[len - 1] == '\n') {

fullName[len - 1] = '\0';

}

// Nhập tên cần tìm kiếm

printf("Nhap vao ten can tim: ");

fgets(name, sizeof(name), stdin);

// Xóa ký tự xuống dòng (nếu có) ở cuối chuỗi do fgets thêm vào

len = strlen(name);

if (len > 0 && name[len - 1] == '\n') {

name[len - 1] = '\0';

}

// Tìm kiếm tên trong chuỗi họ tên

if (searchName(fullName, name)) {

printf("Ten '%s' da nhap dung trong ho ten.\n", name);

} else {

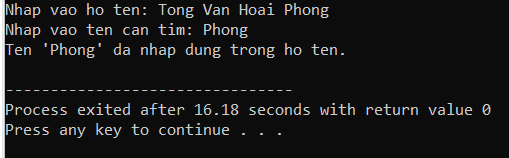
printf("Ten '%s' da nhap sai trong ho ten.\n", name);

}

return 0;

}

Kết quả:



1. Viết hàm cắt chuỗi họ tên thành chuỗi họ lót và chuỗi tên.

Ví dụ: chuỗi họ tên là: “Nguyễn Văn A” cắt ra 2 chuỗi là chuỗi họ lót: “NguyễnVăn” ,chuỗi tên là: “A”.

Code Bai6:

#include <stdio.h>

#include <string.h>

// Hàm cắt chuỗi họ tên thành chuỗi họ lót và chuỗi tên

void splitFullName(const char \*fullName, char \*lastName, char \*firstName) {

// Tìm vị trí của khoảng trắng cuối cùng

char \*lastSpace = strrchr(fullName, ' ');

if (lastSpace != NULL) {

// Tạo chuỗi họ lót

size\_t lastNameLength = lastSpace - fullName;

strncpy(lastName, fullName, lastNameLength);

lastName[lastNameLength] = '\0';

// Tạo chuỗi tên

strcpy(firstName, lastSpace + 1);

} else {

// Nếu không có khoảng trắng, toàn bộ chuỗi là tên

strcpy(firstName, fullName);

lastName[0] = '\0'; // Họ lót là rỗng

}

}

int main() {

char fullName[100];

char lastName[100];

char firstName[50];

// Nhập chuỗi họ tên từ người dùng

printf("Nhap vao ho ten: ");

fgets(fullName, sizeof(fullName), stdin);

// Xóa ký tự xuống dòng (nếu có) ở cuối chuỗi do fgets thêm vào

size\_t len = strlen(fullName);

if (len > 0 && fullName[len - 1] == '\n') {

fullName[len - 1] = '\0';

}

// Cắt chuỗi họ tên thành chuỗi họ lót và chuỗi tên

splitFullName(fullName, lastName, firstName);

// In kết quả

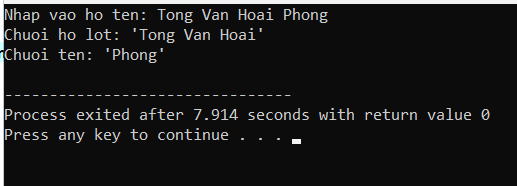
printf("Chuoi ho lot: '%s'\n", lastName);

printf("Chuoi ten: '%s'\n", firstName);

return 0;

}

Kết quả:



1. Viết chương trình chèn 1 từ ở bất cứ vị trí nào người dùng yêu cầu.

Code Bai7:

#include <stdio.h>

#include <string.h>

#define MAX\_SIZE 1000

// Hàm chèn một từ vào chuỗi văn bản tại vị trí chỉ định

void insertWord(char \*text, const char \*word, int position) {

int textLength = strlen(text);

int wordLength = strlen(word);

// Kiểm tra vị trí chèn hợp lệ

if (position < 0 || position > textLength) {

printf("Vi tri chen khong hop le.\n");

return;

}

// Tạo mảng tạm để chứa chuỗi kết quả

char temp[MAX\_SIZE];

// Sao chép phần chuỗi trước vị trí chèn

strncpy(temp, text, position);

temp[position] = '\0'; // Đặt ký tự kết thúc chuỗi

// Nối từ vào phần chuỗi trước vị trí chèn

strcat(temp, word);

// Nối phần chuỗi sau vị trí chèn

strcat(temp, text + position);

// Sao chép chuỗi kết quả trở lại vào biến gốc

strcpy(text, temp);

}

int main() {

char text[MAX\_SIZE];

char word[MAX\_SIZE];

int position;

// Nhập chuỗi văn bản từ người dùng

printf("Nhap vao chuoi van ban: ");

fgets(text, sizeof(text), stdin);

// Xóa ký tự xuống dòng (nếu có) ở cuối chuỗi do fgets thêm vào

size\_t len = strlen(text);

if (len > 0 && text[len - 1] == '\n') {

text[len - 1] = '\0';

}

// Nhập từ cần chèn

printf("Nhap vao tu can chen: ");

fgets(word, sizeof(word), stdin);

// Xóa ký tự xuống dòng (nếu có) ở cuối chuỗi do fgets thêm vào

len = strlen(word);

if (len > 0 && word[len - 1] == '\n') {

word[len - 1] = '\0';

}

// Nhập vị trí chèn

printf("Nhap vao vi tri chen: ");

scanf("%d", &position);

// Chèn từ vào chuỗi văn bản tại vị trí chỉ định

insertWord(text, word, position);

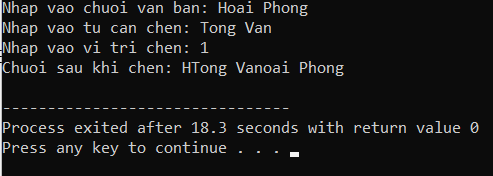
// In kết quả

printf("Chuoi sau khi chen: %s\n", text);

return 0;

}

Kết quả:



8. Viết 1 chương trình xoá một từ nào đó trong chuỗi.

Ví dụ: Chuỗi ban đầu: “KHOA CNTT”

Nhập: “CNTT”, và kết quả xuất ra: “KHOA”.

Code Bai8:

#include <stdio.h>

#include <string.h>

#include <ctype.h>

// Hàm xóa một từ khỏi chuỗi và loại bỏ khoảng trắng dư thừa

void removeWord(char \*text, const char \*word) {

char \*pos;

size\_t wordLen = strlen(word);

while ((pos = strstr(text, word)) != NULL) {

// Xóa từ bằng cách di chuyển phần sau từ

memmove(pos, pos + wordLen, strlen(pos + wordLen) + 1);

// Xóa khoảng trắng dư thừa ở đầu và cuối chuỗi

char \*start = text;

char \*end = text + strlen(text) - 1;

// Xóa khoảng trắng ở đầu chuỗi

while (\*start == ' ') start++;

// Xóa khoảng trắng ở cuối chuỗi

while (end > start && \*end == ' ') end--;

\*(end + 1) = '\0'; // Kết thúc chuỗi

// Di chuyển phần còn lại về đầu chuỗi

memmove(text, start, strlen(start) + 1);

}

}

int main() {

char text[100];

char word[50];

// Nhập chuỗi văn bản từ người dùng

printf("Nhap vao chuoi: ");

fgets(text, sizeof(text), stdin);

// Xóa ký tự xuống dòng (nếu có) ở cuối chuỗi do fgets thêm vào

size\_t len = strlen(text);

if (len > 0 && text[len - 1] == '\n') {

text[len - 1] = '\0';

}

// Nhập từ cần xóa

printf("Nhap vao tu can xoa: ");

fgets(word, sizeof(word), stdin);

// Xóa ký tự xuống dòng (nếu có) ở cuối chuỗi do fgets thêm vào

len = strlen(word);

if (len > 0 && word[len - 1] == '\n') {

word[len - 1] = '\0';

}

// Xóa từ khỏi chuỗi văn bản

removeWord(text, word);

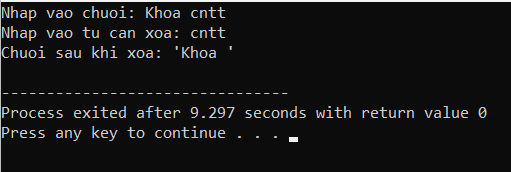
// In kết quả

printf("Chuoi sau khi xoa: '%s'\n", text);

return 0;

}

Kết quả:



9. Nhập 1 chuỗi bất kì, liệt kê xem mỗi ký tự xuất hiện mấy lần.

Code Bai9:

#include <stdio.h>

#include <string.h>

#define MAX\_CHARS 256 // Kích thước mảng đếm cho tất cả ký tự ASCII

// Hàm liệt kê số lần xuất hiện của mỗi ký tự trong chuỗi

void countCharacters(const char \*str) {

int count[MAX\_CHARS] = {0}; // Mảng đếm số lần xuất hiện của mỗi ký tự

int i;

// Đếm số lần xuất hiện của mỗi ký tự

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

count[(unsigned char)str[i]]++;

}

// In kết quả

printf("So lan xuat hien cua moi ky tu:\n");

for (i = 0; i < MAX\_CHARS; i++) {

if (count[i] > 0) {

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

}

}

}

int main() {

char str[1000];

// Nhập chuỗi từ người dùng

printf("Nhap vao chuoi: ");

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

// Xóa ký tự xuống dòng (nếu có) ở cuối chuỗi do fgets thêm vào

size\_t len = strlen(str);

if (len > 0 && str[len - 1] == '\n') {

str[len - 1] = '\0';

}

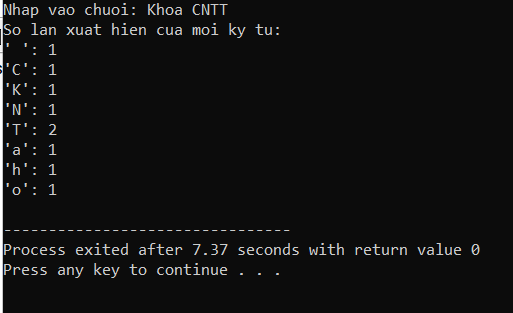
// Liệt kê số lần xuất hiện của mỗi ký tự trong chuỗi

countCharacters(str);

return 0;

}

Kết quả:



10. Viết chương trình tìm kiếm xem ký tự nào xuất nhiện nhiều nhất trong chuỗi.

Code Bai10:

#include <stdio.h>

#include <string.h>

#define MAX\_CHARS 256 // Kích thước mảng đếm cho tất cả ký tự ASCII

// Hàm tìm ký tự xuất hiện nhiều nhất

char findMostFrequentChar(const char \*str, int \*maxCount) {

int count[MAX\_CHARS] = {0}; // Mảng đếm số lần xuất hiện của mỗi ký tự

int i;

char mostFrequentChar = '\0';

\*maxCount = 0;

// Đếm số lần xuất hiện của mỗi ký tự

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

count[(unsigned char)str[i]]++;

}

// Tìm ký tự xuất hiện nhiều nhất

for (i = 0; i < MAX\_CHARS; i++) {

if (count[i] > \*maxCount) {

\*maxCount = count[i];

mostFrequentChar = i;

}

}

return mostFrequentChar;

}

int main() {

char str[1000];

int maxCount;

char mostFrequentChar;

// Nhập chuỗi từ người dùng

printf("Nhap vao chuoi: ");

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

// Xóa ký tự xuống dòng (nếu có) ở cuối chuỗi do fgets thêm vào

size\_t len = strlen(str);

if (len > 0 && str[len - 1] == '\n') {

str[len - 1] = '\0';

}

// Tìm ký tự xuất hiện nhiều nhất

mostFrequentChar = findMostFrequentChar(str, &maxCount);

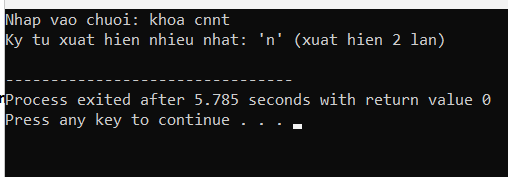
// In kết quả

printf("Ky tu xuat hien nhieu nhat: '%c' (xuat hien %d lan)\n", mostFrequentChar, maxCount);

return 0;

}

Kết quả:



The End