**Bài 1: Đếm chữ hoa, chữ thường và chữ số**

#include <stdio.h>

#include <ctype.h>

int main() {

char sentence[1000];

int uppercase = 0, lowercase = 0, digits = 0;

printf("Please enter a sentence: ");

gets(sentence);

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

if (isupper(sentence[i])) {

uppercase++;

} else if (islower(sentence[i])) {

lowercase++;

} else if (isdigit(sentence[i])) {

digits++;

}

}

printf("The number of uppercase alphabets is %d\n", uppercase);

printf("The number of lowercase alphabets is %d\n", lowercase);

printf("The number of digits is %d\n", digits);

return 0;

}

**Bài 2: Đếm số nguyên âm và phụ âm**

#include <stdio.h>

#include <ctype.h>

int main() {

char sentence[1000];

int vowels = 0, consonants = 0;

printf("Please enter a sentence: ");

gets(sentence);

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

char ch = tolower(sentence[i]);

if (isalpha(ch)) {

if (ch == 'a' || ch == 'e' || ch == 'i' || ch == 'o' || ch == 'u' || ch == 'y') {

vowels++;

} else {

consonants++;

}

}

}

printf("The number of vowel alphabets is %d\n", vowels);

printf("The number of consonant alphabets is %d\n", consonants);

return 0;

}

**Bài 3: Đếm số lần xuất hiện của một ký tự trong chuỗi**

#include <stdio.h>

#include <ctype.h>

int main() {

char sentence[1000], target;

int count = 0;

printf("Please enter a sentence: ");

gets(sentence);

printf("Letter to find: ");

scanf(" %c", &target);

if (!isalpha(target)) {

printf("Character '%c' is not a letter. Accept letter only!\n", target);

return 0;

}

target = tolower(target);

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

if (tolower(sentence[i]) == target) {

count++;

}

}

if (count > 0) {

printf("The letter '%c' appears %d times in the entered sentence\n", target, count);

} else {

printf("The character '%c' was not found in the entered sentence\n", target);

}

return 0;

}

**Bài 4: Đếm và hiển thị vị trí xuất hiện của một ký tự trong chuỗi**

#include <stdio.h>

#include <ctype.h>

int main() {

char sentence[1000], target;

int count = 0;

printf("Please enter a sentence: ");

gets(sentence);

printf("Letter to find: ");

scanf(" %c", &target);

if (!isalpha(target)) {

printf("Character '%c' is not a letter. Accept letter only!\n", target);

return 0;

}

target = tolower(target);

printf("The letter '%c' appears at indexes: ", target);

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

if (tolower(sentence[i]) == target) {

printf("[%d] ", i);

count++;

}

}

if (count == 0) {

printf("The character '%c' was not found in the entered sentence\n", target);

} else {

printf("\nThe letter '%c' appears %d times\n", target, count);

}

return 0;

}

**Bài 5: Đếm số từ trong câu**

#include <stdio.h>

#include <string.h>

#include <ctype.h>

int main() {

char sentence[1000];

int word\_count = 0;

printf("Please enter a sentence: ");

gets(sentence);

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

if (!isspace(sentence[i]) && (i == 0 || isspace(sentence[i - 1]))) {

word\_count++;

}

}

printf("There are %d words in the sentence\n", word\_count);

return 0;

}

**Bài 6: Chuyển tất cả ký tự thành chữ thường**

#include <stdio.h>

#include <ctype.h>

int main() {

char sentence[1000];

printf("Please enter a sentence: ");

gets(sentence);

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

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

}

printf("The normalized sentence is \"%s\"\n", sentence);

return 0;

}

**Bài 7: Chuyển tất cả ký tự thành chữ hoa**

#include <stdio.h>

#include <ctype.h>

int main() {

char sentence[1000];

printf("Please enter a sentence: ");

gets(sentence);

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

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

}

printf("The normalized sentence is \"%s\"\n", sentence);

return 0;

}

**Bài 8: Đổi chữ hoa thành chữ thường và ngược lại**

#include <stdio.h>

#include <ctype.h>

int main() {

char sentence[1000];

printf("Please enter a sentence: ");

gets(sentence);

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

if (isupper(sentence[i])) {

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

} else if (islower(sentence[i])) {

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

}

}

printf("The normalized sentence is \"%s\"\n", sentence);

return 0;

}

**Bài 9: Viết hoa chữ cái đầu của mỗi từ**

#include <stdio.h>

#include <ctype.h>

int main() {

char sentence[1000];

int capitalize = 1;

printf("Please enter a sentence: ");

gets(sentence);

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

if (capitalize && isalpha(sentence[i])) {

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

capitalize = 0;

} else {

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

}

if (isspace(sentence[i])) {

capitalize = 1;

}

}

printf("The normalized sentence is \"%s\"\n", sentence);

return 0;

}

**Bài 10: Loại bỏ khoảng trắng thừa**

#include <stdio.h>

#include <string.h>

#include <ctype.h>

int main() {

char sentence[1000];

char normalized[1000];

int i = 0, j = 0;

printf("Please enter a sentence: ");

gets(sentence);

// Bỏ khoảng trắng đầu

while (isspace(sentence[i])) i;

// Bỏ khoảng trắng giữa và cuối

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

if (!isspace(sentence[i]) || (j > 0 && !isspace(normalized[j - 1]))) {

normalized[j++] = sentence[i];

}

}

// Bỏ khoảng trắng cuối

if (j > 0 && isspace(normalized[j - 1])) j--;

normalized[j] = '\0';

printf("The normalized sentence: \"%s\"\n", normalized);

return 0;

}

**Bài 11: Loại bỏ chữ số trong chuỗi**

#include <stdio.h>

#include <ctype.h>

int main() {

char sentence[1000];

char result[1000];

int j = 0;

printf("Please enter a sentence: ");

gets(sentence);

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

if (!isdigit(sentence[i])) {

result[j++] = sentence[i];

}

}

result[j] = '\0';

printf("The normalized sentence is \"%s\"\n", result);

return 0;

}

**Bài 12: Loại bỏ một ký tự cụ thể trong chuỗi**

#include <stdio.h>

int main() {

char sentence[1000], target, result[1000];

int j = 0;

printf("Please enter a sentence: ");

gets(sentence);

printf("Letter to remove: ");

scanf(" %c", &target);

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

if (sentence[i] != target) {

result[j++] = sentence[i];

}

}

result[j] = '\0';

printf("The normalized sentence is \"%s\"\n", result);

return 0;

}

**Bài 13: Thay thế một ký tự bằng ký tự khác**

#include <stdio.h>

int main() {

char sentence[1000], oldChar, newChar;

printf("Please enter a sentence: ");

gets(sentence);

printf("Letter to find: ");

scanf(" %c", &oldChar);

printf("Letter to replace: ");

scanf(" %c", &newChar);

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

if (sentence[i] == oldChar) {

sentence[i] = newChar;

}

}

printf("The normalized sentence is \"%s\"\n", sentence);

return 0;

}

**Bài 14: Đảo ngược toàn bộ ký tự trong câu**

#include <stdio.h>

#include <string.h>

int main() {

char sentence[1000];

printf("Please enter a sentence: ");

gets(sentence);

int len = strlen(sentence);

for (int i = 0; i < len / 2; i++) {

char temp = sentence[i];

sentence[i] = sentence[len - i - 1];

sentence[len - i - 1] = temp;

}

printf("The normalized sentence is \"%s\"\n", sentence);

return 0;

}

**Bài 15: Đảo ngược thứ tự các từ trong câu**

#include <stdio.h>

#include <string.h>

void reverse\_words(char \*sentence) {

int len = strlen(sentence);

char reversed[1000];

int start = len - 1, end = len, j = 0;

while (start >= 0) {

if (sentence[start] == ' ' || start == 0) {

int i = (start == 0) ? start : start + 1;

while (i < end) {

reversed[j++] = sentence[i++];

}

if (start > 0) reversed[j++] = ' ';

end = start;

}

start--;

}

reversed[j] = '\0';

strcpy(sentence, reversed);

}

int main() {

char sentence[1000];

printf("Please enter a sentence: ");

gets(sentence);

reverse\_words(sentence);

printf("The normalized sentence is \"%s\"\n", sentence);

return 0;

}

**Bài 16: Kiểm tra chuỗi palindrome**

#include <stdio.h>

#include <string.h>

#include <ctype.h>

int is\_palindrome(char \*text) {

int left = 0, right = strlen(text) - 1;

while (left < right) {

if (text[left] != text[right]) {

return 0;

}

left++;

right--;

}

return 1;

}

int main() {

char text[1000];

printf("Please enter any text: ");

gets(text);

if (is\_palindrome(text)) {

printf("The text \"%s\" is a palindrome text\n", text);

} else {

printf("The text \"%s\" is not a palindrome text\n", text);

}

return 0;

}

**Bài 17: Trích xuất họ, tên đệm và tên**

#include <stdio.h>

#include <string.h>

int main() {

char full\_name[1000], first[100], middle[1000], last[100];

char \*token;

int count = 0;

printf("Please enter your full name: ");

gets(full\_name);

token = strtok(full\_name, " ");

strcpy(first, token);

count++;

while ((token = strtok(NULL, " ")) != NULL) {

if (strtok(NULL, " ") == NULL) {

strcpy(last, token);

break;

} else {

strcat(middle, token);

strcat(middle, " ");

}

}

printf("The first name is %s\n", first);

printf("The middle name is %s\n", middle);

printf("The last name is %s\n", last);

return 0;

}

**Bài 18: Tính toán vị trí của LeO**

#include <stdio.h>

int main() {

char message[1000];

int x = 0, y = 0;

printf("Please enter the message: ");

gets(message);

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

switch (message[i]) {

case 'N': y++; break;

case 'S': y--; break;

case 'E': x++; break;

case 'W': x--; break;

}

}

printf("The location of LeO is (%d, %d)\n", x, y);

return 0;

}

**Bài 19: Kiểm tra DNA của vua và cô gái**

#include <stdio.h>

#include <string.h>

int count\_occurrences(char \*dna, char \*sequence) {

int count = 0, len = strlen(sequence);

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

if (strncmp(&dna[i], sequence, len) == 0) {

count++;

}

}

return count;

}

int main() {

char king\_dna[1000], girl\_dna[1000];

printf("Please enter the king's DNA: ");

gets(king\_dna);

printf("Please enter the girl's DNA: ");

gets(girl\_dna);

for (int i = 0; king\_dna[i] != '\0'; i++) {

if (strchr("ATCG", king\_dna[i]) == NULL) {

printf("Only accept A, T, C, G for king's DNA!\n");

return 0;

}

}

for (int i = 0; girl\_dna[i] != '\0'; i++) {

if (strchr("ATCG", girl\_dna[i]) == NULL) {

printf("Only accept A, T, C, G for girl's DNA!\n");

return 0;

}

}

int occurrences = count\_occurrences(king\_dna, girl\_dna);

printf("Analysis results: %d times\n", occurrences);

if (occurrences >= 3) {

printf("The girl is the king's biological daughter\n");

} else {

printf("She lied and was sentenced to prison\n");

}

return 0;

}

**Bài 20: Xử lý chuỗi backspace**

#include <stdio.h>

#include <string.h>

int main() {

char message[1000001], result[1000001];

int j = 0;

printf("Please enter the message: ");

gets(message);

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

if (message[i] == '<') {

if (j > 0) j--;

} else {

result[j++] = message[i];

}

}

result[j] = '\0';

printf("The final message is \"%s\"\n", result);

return 0;

}

**Bài 21: Loại bỏ ký tự liên tiếp lặp lại trong tên**

#include <stdio.h>

#include <string.h>

int main() {

char name[251], result[251];

int j = 0;

printf("Please enter the name: ");

gets(name);

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

if (i == 0 || name[i] != name[i - 1]) {

result[j++] = name[i];

}

}

result[j] = '\0';

printf("The compact version of the name: \"%s\"\n", result);

return 0;

}

**Bài 22: Kiểm tra chuỗi pangram**

#include <stdio.h>

#include <string.h>

#include <ctype.h>

int main() {

char phrase[101];

int alphabet[26] = {0};

printf("Please enter the phrase: ");

gets(phrase);

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

if (isalpha(phrase[i])) {

alphabet[tolower(phrase[i]) - 'a'] = 1;

}

}

int is\_pangram = 1;

for (int i = 0; i < 26; i++) {

if (alphabet[i] == 0) {

is\_pangram = 0;

break;

}

}

if (is\_pangram) {

printf("The test result: pangram\n");

} else {

printf("The test result: missing ");

for (int i = 0; i < 26; i++) {

if (alphabet[i] == 0) {

printf("%c", 'a' + i);

}

}

printf("\n");

}

return 0;

}

**Bài 23: Mã hóa Caesar Cipher**

#include <stdio.h>

#include <ctype.h>

int main() {

char message[1000];

int key;

printf("Please enter the original message: ");

gets(message);

printf("Please enter the encryption key: ");

scanf("%d", &key);

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

if (isalpha(message[i])) {

char base = islower(message[i]) ? 'a' : 'A';

message[i] = (message[i] - base + key) % 26 + base;

}

}

printf("The encrypted message: %s\n", message);

return 0;

}