**자료구조론 CC343\_2207**

**Reading assignment 4**

**경기대학교 컴퓨터공학부**

**201511837 이상민**

**Review Questions**

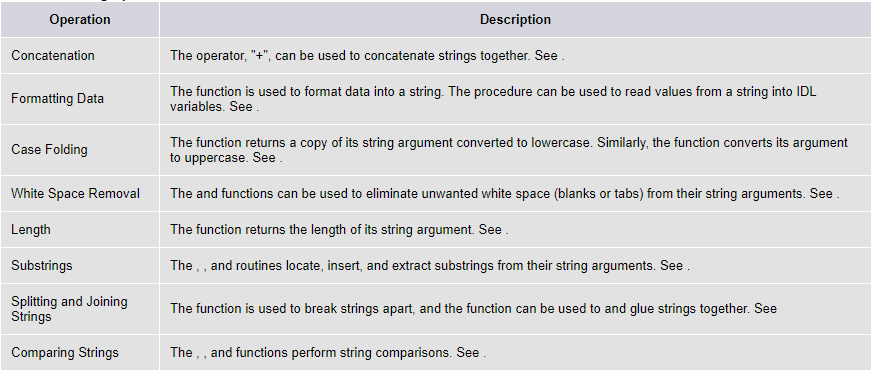
1. What are strings? Discuss some of the operations that can be performed on strings.  
String란 무엇인가? 문자열에서 수행할 수 있는 몇 가지 작업에 대해 논의하십시오.

In , a string is traditionally a of , either as a or as some kind of variable. The latter may allow its elements to be mutated and the length changed, or it may be fixed (after creation). A string is generally considered as a and is often implemented as an of (or ) that stores a sequence of elements, typically characters, using some . String may also denote more general or other sequence (or ) data types and structures.

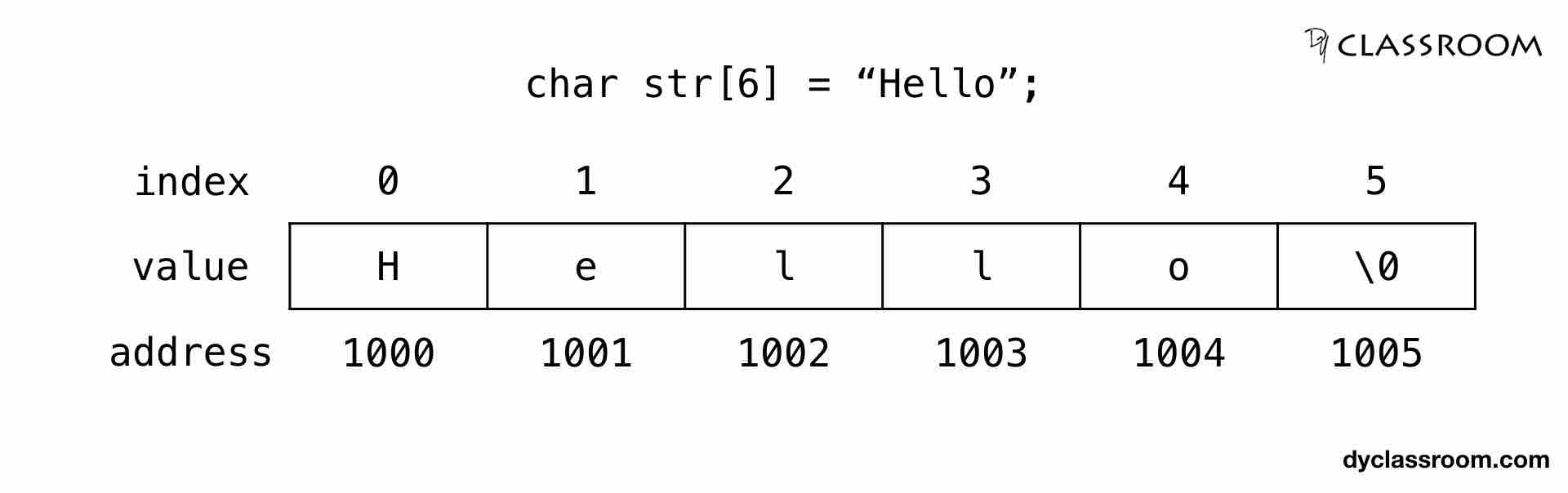
Depending on the programming language and precise data type used, a declared to be a string may either cause storage in memory to be statically allocated for a predetermined maximum length or employ to allow it to hold a variable number of elements.

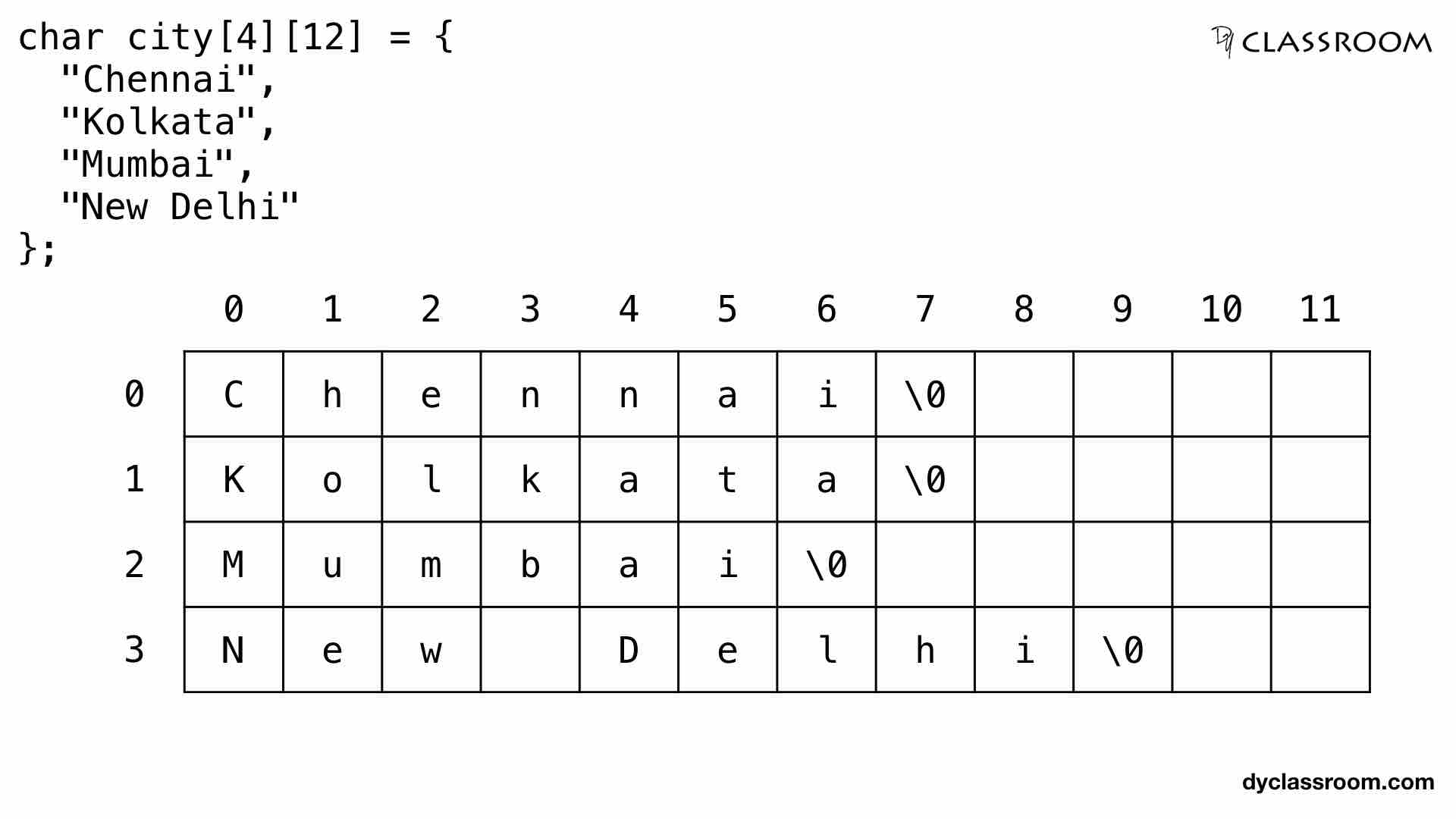
When a string appears literally in , it is known as a or an anonymous string.

In , which are used in and , a string is a finite sequence of that are chosen from a called an .



2. Explain how strings are represented in the main memory.  
메인 메모리에 string이 어떻게 표현되는지 설명하라.



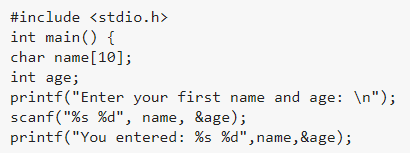


3. How are strings read from the standard input device? Explain the different functions used to perform the string input operation.  
표준 입력 장치에서 문자열을 어떻게 읽는가? 문자열 입력 작동을 수행하는 데 사용되는 다양한 기능에 대해 설명하십시오.

String Input: Read a String

When writing interactive programs which ask the user for input, C provides the scanf(), gets(), and fgets() functions to find a line of text entered from the user.

When we use scanf() to read, we use the "%s" format specifier without using the "&" to access the variable address because an array name acts as a pointer. For example:

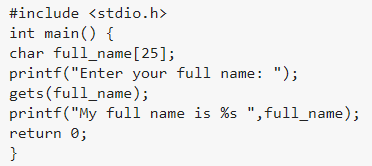


Output :



The problem with the scanf function is that it never reads an entire string. It will halt the reading process as soon as whitespace, form feed, vertical tab, newline or a carriage return occurs. Suppose we give input as "Guru99 Tutorials" then the scanf function will never read an entire string as a whitespace character occurs between the two names. The scanf function will only read Guru99.

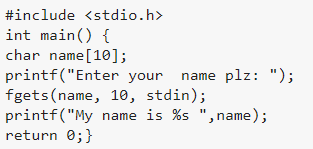
In order to read a string contains spaces, we use the gets() function. Gets ignores the whitespaces. It stops reading when a newline is reached (the Enter key is pressed).For example:



Output :



Another safer alternative to gets() is fgets() function which reads a specified number of characters. For example:



Output :



The fgets() arguments are :

* the string name,
* the number of characters to read,
* stdin means to read from the standard input which is the keyboard.

4. Explain how strings can be displayed on the screen.  
화면에 문자열을 표시할 수 있는 방법을 설명한다.

The standard printf function is used for printing or displaying a string on an output device. The format specifier used is %s

Example,

Printf(“%s”, name);

String output is done with the fputs() and printf() functions.

fputs() function

The fputs() needs the name of the string and a pointer to where you want to display the text. We use stdout which refers to the standard output in order to print to the screen.For example:

#include <stdio.h>

Int main() {

Char town[40];

Printf(“Enter your town : “);

Gets(town);

Fputs(town, stdout);

Return 0; }

Output :

Enter your town : New York

New York

puts function

The puts function prints the string on an output device and moves the cursor back to the first position. A puts function can be used in the following way,

#include <stdio.h>

int main() {

char name[15];

gets(name); //reads a string

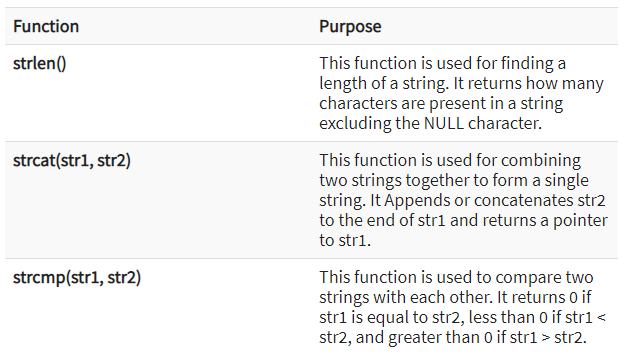
puts(name); //displays a string

return 0;}

The syntax of this function is comparatively simple than other functions.

The string library

The standard 'C' library provides various functions to manipulate the strings within a program. These functions are also called as string handlers. All these handlers are present inside <string.h> header file.



5. Explain the syntax of printf() and scanf().  
printf()와 scanf()의 구문을 설명한다.

PRINTF() FUNCTION IN C LANGUAGE:

* In C programming language, printf() function is used to print the “character, string, float, integer, octal and hexadecimal values” onto the output screen.
* We use printf() function with %d format specifier to display the value of an integer variable.
* Similarly %c is used to display character, %f for float variable, %s for string variable, %lf for double and %x for hexadecimal variable.
* To generate a newline,we use “\n” in C printf() statement.
* %d got replaced by value of an integer variable (no),
* %c got replaced by value of a character variable (ch),
* %f got replaced by value of a float variable (flt),
* %lf got replaced by value of a double variable (dbl),
* %s got replaced by value of a string variable (str),
* %o got replaced by a octal value corresponding to integer variable (no),
* %x got replaced by a hexadecimal value corresponding to integer variable
* \n got replaced by a newline.

SCANF() FUNCTION IN C LANGUAGE:

* In C programming language, scanf() function is used to read character, string, numeric data from keyboard
* Consider below example program where user enters a character. This value is assigned to the variable “ch” and then displayed.
* Then, user enters a string and this value is assigned to the variable “str” and then displayed.

KEY POINTS TO REMEMBER IN C PRINTF() AND SCANF():

1. printf() is used to display the output and scanf() is used to read the inputs.
2. printf() and scanf() functions are declared in “stdio.h” header file in C library.
3. All syntax in C language including printf() and scanf() functions are case sensitive.

6. List all the substrings that can be formed from the string ‘ABCD’.  
문자열 'ABCD'에서 형성할 수 있는 모든 하위 문자열을 나열하십시오.

“a”, “b”, “c”, “d”, “ab”, “bc”, “cd”, “abc”, “bcd” and “abcd”

7. What do you understand by pattern matching? Give an algorithm for it.   
패턴 매칭으로 무엇을 이해하십니까? 알고리즘을 줘봐

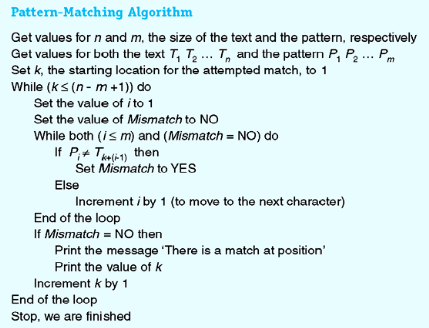
Definition - What does Pattern Matching mean?

Pattern matching in computer science is the checking and locating of specific sequences of data of some pattern among raw data or a sequence of tokens. Unlike pattern recognition, the match has to be exact in the case of pattern matching. Pattern matching is one of the most fundamental and important paradigms in several programming languages. Many applications make use of pattern matching as a major part of their tasks.

Pattern Matching explains

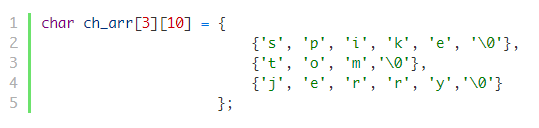
Pattern matching, in its classical form, involves the use of one-dimensional string matching. Patterns are either tree structures or sequences. There are different classes of programming languages and machines which make use of pattern matching. In the case of machines, the major classifications include deterministic finite state automata, deterministic pushdown automata, nondeterministic pushdown automata and Turing machines. Regular programming languages make use of regular expressions for pattern matching. Tree patterns are also used in certain programming languages like Haskell as a tool to process data based on the structure. Compared to regular expressions, tree patterns lack simplicity and efficiency.

There are many applications for pattern matching in computer science. High-level language compilers make use of pattern matching in order to parse source files to determine if they are syntactically correct. In programming languages and applications, pattern matching is used in identifying the matching pattern or substituting the matching pattern with another token sequence.



8. Write a short note on array of strings.  
배열의 스트링에 대해서 짧은 메모를 써라.

9. Explain with an example how an array of strings is stored in the main memory.  
줄의 배열들이 메인 메모리에 어떻게 저장되어 있는지 예를 들어 설명하라.



The first subscript of the array i.e 3 denotes the number of strings in the array and the second subscript denotes the maximum length of the string. Recall the that in C, each character occupies 1 byte of data, so when the compiler sees the above statement it allocates 30 bytes (3\*10) of memory.

We already know that the name of an array is a pointer to the 0th element of the array. Can you guess the type of ch\_arr?

The ch\_arr is a pointer to an array of 10 characters or int(\*)[10].

Therefore, if ch\_arr points to address 1000 then ch\_arr + 1 will point to address 1010.

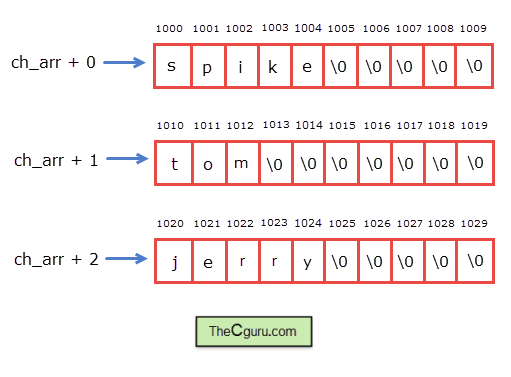
From here we can conclude that:

ch\_arr + 0 points to the 0th string or 0th 1-D array.

ch\_arr + 1 points to the 1st string or 1st 1-D array.

ch\_arr + 2 points to the 2nd string or 2nd 1-D array.

In general, ch\_arr + i points to the ith string or ith 1-D array.



10. Explain how pointers and strings are related to each other with the help of a suitable program.  
적절한 프로그램의 도움으로 포인터와 string이 서로 어떻게 관련되어 있는지 설명하라.

Creating a pointer for the string

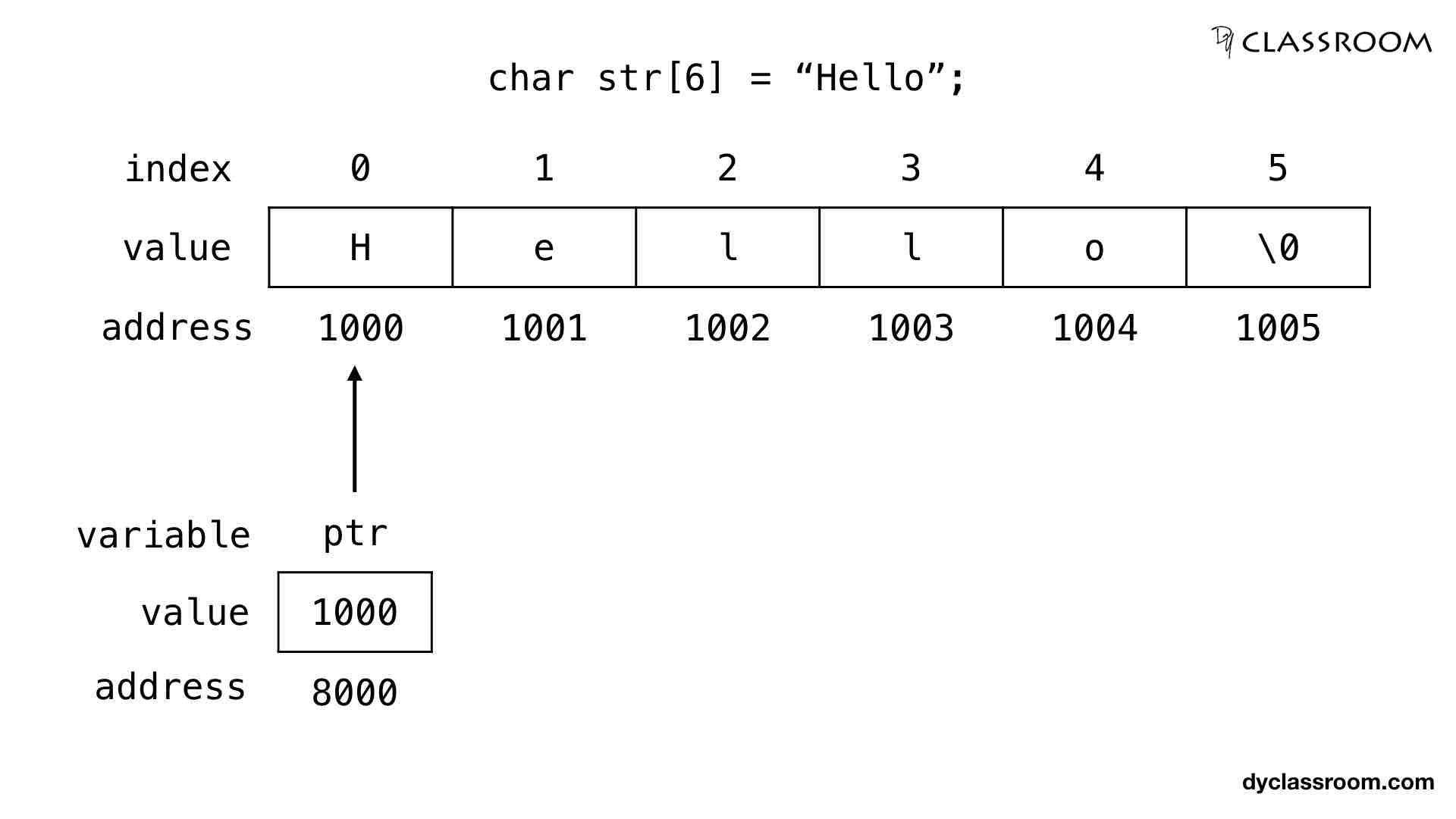
The variable name of the string str holds the address of the first element of the array i.e., it points at the starting memory address.

So, we can create a character pointer ptr and store the address of the string str variable in it. This way, ptr will point at the string str.

In the following code we are assigning the address of the string str to the pointer ptr.

Char \*ptr = str;

We can represent the character pointer variable ptr as follows.



The pointer variable ptr is allocated memory address 8000 and it holds the address of the string variable str i.e., 1000.

11. If the substring function is given as SUBSTRING (string,position,length), then find S(5,9) if S = "Welcome to World of C Programming"  
하위 문자열 함수가 SUBSTRING(줄, 위치, 길이)으로 주어진 경우 S = "C 프로그래밍의 세계에 온 것"인 경우 S(5,9)를 찾으십시오.

#include <stdio.h>

int main()

{

char string[1000], sub[1000];

int position, length, c = 0;

printf("Input a string\n");

gets(string);

printf("Enter the position and length of substring\n");

scanf("%d%d", &position, &length);

while (c < length) {

sub[c] = string[position + c - 1];

c++;

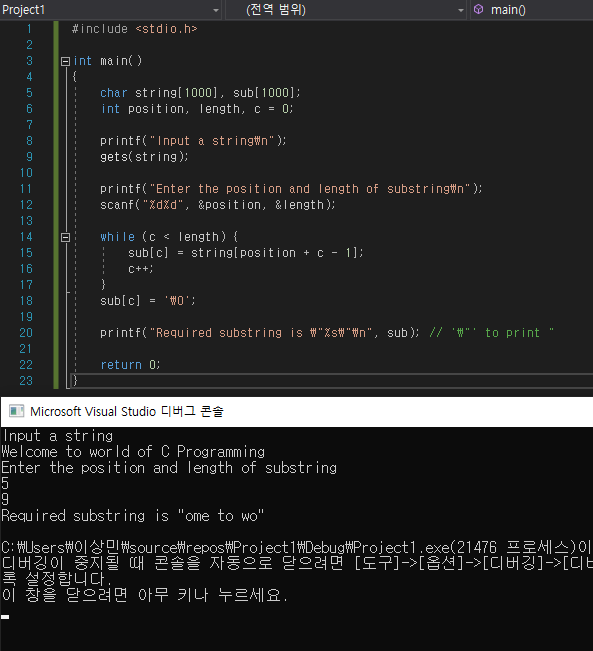
}

sub[c] = '\0';

printf("Required substring is \"%s\"\n", sub); // '\"' to print "

return 0;

}



12. If the index function is given as INDEX(text, pattern), then find index(T, P) where T =  
"Welcome to World of C Programming" and P = "of"  
인덱스 함수가 INDEX(텍스트, 패턴)로 주어진 경우 T = 있는 인덱스(T, P)를 찾으십시오.  
"Welcome to World of C Programming" 및 P = "of"

#include <stdio.h>

#include <string.h>

void main()

{

char \*string = "Welcome to World of C Programming";

char \*of;

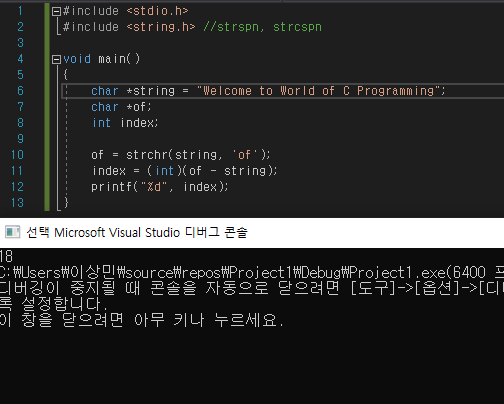
int index;ghkl

of = strchr(string, 'of');

index = (int)(of - string);

printf("%d", index);

}



13. Differentiate between gets() and scanf().  
get()와 scanf()를 구분한다.

scanf()

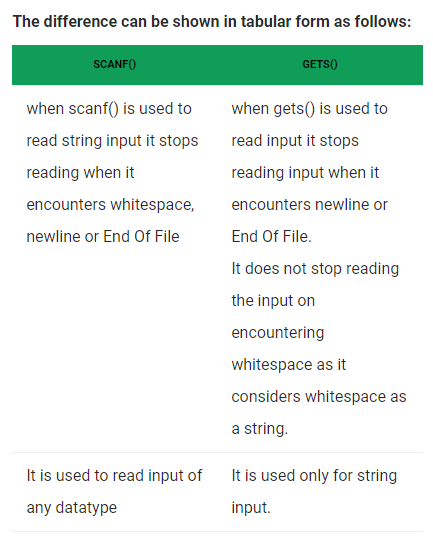
* It is used to read the input(character, string, numeric data) from the standard input(keyboard).
* It is used to read the input until it encounters a whitespace, newline or End Of File(EOF).

gets

* It is used to read input from the standard input(keyboard).
* It is used to read the input until it encounters newline or End Of File(EOF).

The main difference between them is:

1. scanf() reads input until it encounters whitespace, newline or End Of File(EOF) whereas gets() reads input until it encounters newline or End Of File(EOF), gets() does not stop reading input when it encounters whitespace instead it takes whitespace as a string.
2. scanf can read multiple values of different data types whereas gets() will only get character string data.



14. Give the drawbacks of getchar() and scanf().  
getchar()와 scanf()의 단점을 제시한다.

scanf()의 단점은 자료형마다 %c, %s, %d 등의 서식 설정의 귀찮음이 있습니다. 또한 버퍼 때문에 연속 사용시 데이터 누락이 발생할 수 있습니다.

버퍼는 데이터 임시 저장소입니다. C언어는 문자를 한 글자 입력할 때 마다 입력과 출력을 하지 않고, 버퍼라는 임시 저장소에 저장을 해둡니다. 이 후 Enter를 누르면 \n 개행문자가 들어가면서 그제서야 버퍼에 저장했던 문자를 내보냅니다. 이 방법의 단점은 개행 문자 \n도 임시 저장소에 들어간다는 점입니다. scanf() 함수를 연속 사용해보면 오류가 나기 때문에 버퍼가 어떤 의미인지 직관적으로 알 수 있습니다.

getchar()의 단점은 scanf()와 마찬가지로 버퍼가 있습니다. 해결 방법은 scanf()와 동일하게 getchar()함수를 중간에 삽입하거나 처음부터 입력 문자 개수를 지시합니다.

15. Which function can be used to overcome the shortcomings of getchar() and scanf()?  
getchar()와 scanf()의 단점을 극복하기 위해 사용할 수 있는 기능은 무엇인가?

How to solve above problem?

We can make scanf() to read a new line by using an extra “\n”, i.e., scanf(“%d\n”, &x) . In fact scanf(“%d “, &x) also works (Note extra space).

We can add a getchar() after scanf() to read an extra newline.

16. How can putchar() be used to print a string?  
문자열을 인쇄할 때 어떻게 putchar()를 사용할 수 있는가?

void printString(char \*ch)  
{  
 while (\*ch = getchar(), \*ch >= 0)  
 putchar(\*ch);  
}

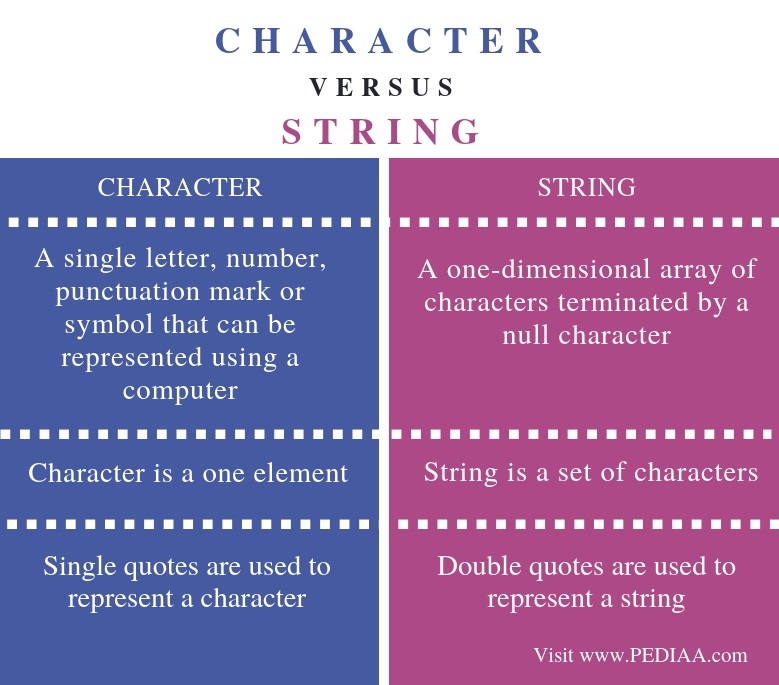
to change :

void printString(char \*ch)  
{  
 while(\*ch) {  
 putchar(\*ch);  
 ch++;  
 }  
}

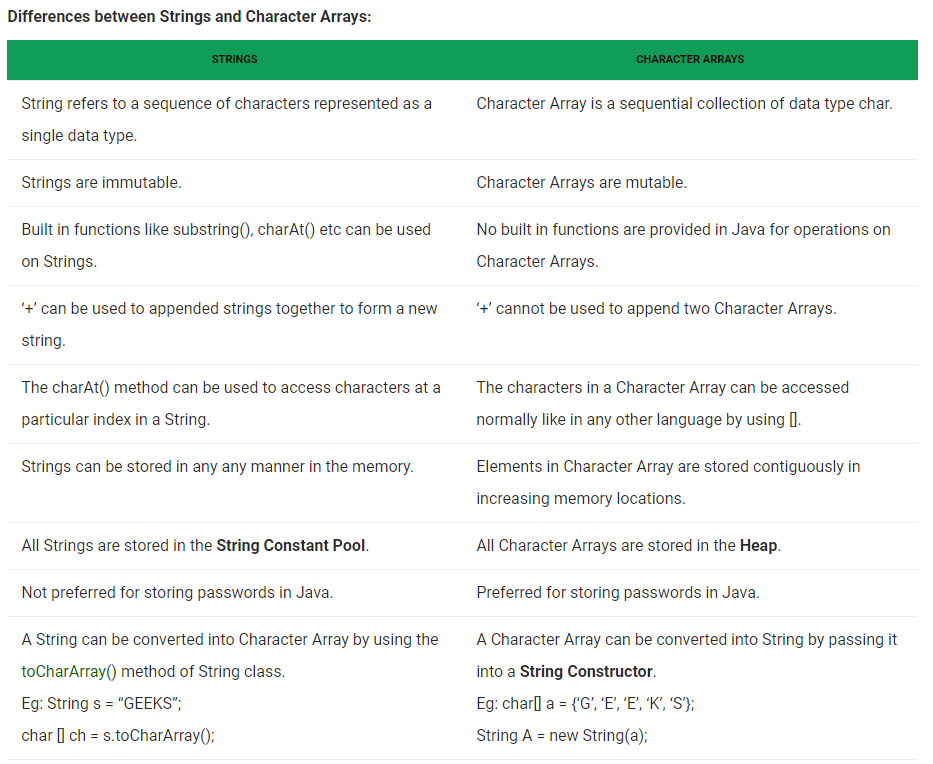
Does a lot of stuff:

1. reads characters from stdin
2. stores the character read from stdin into the first char pointed to by ch (this might not even work if you pass in a string literal.
3. writes characters to stdout.
4. Terminates when the read character is < 0 (this won't work on some platforms. Since the result is stored in a char you can't distinguish between EOF and a valid character. ch should be an int, as getchar() returns an int so you can check for EOF)

17. Differentiate between a character and a string.  
캐릭터와 문자열을 구분한다.

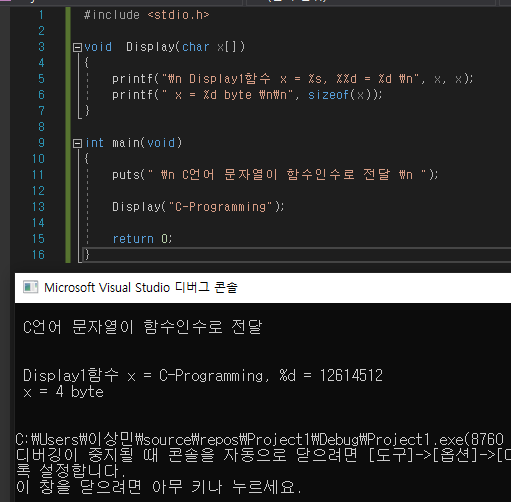


18. Differentiate between a character array and a string.  
문자열과 문자열을 구분한다.



**Programming exercises**

1. Write a program in which a string is passed as an argument to a function.  
문자열이 함수에 대한 인수로서 전달되는 프로그램을 작성한다.



#include <stdio.h>

void Display(char x[])

{

printf("\n Display1함수 x = %s, %%d = %d \n", x, x);

printf(" x = %d byte \n\n", sizeof(x));

}

int main(void)

{

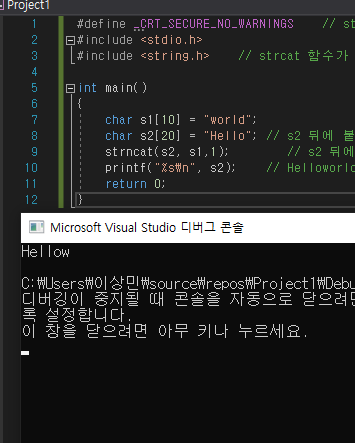
puts(" \n C언어 문자열이 함수인수로 전달 \n ");

Display("C-Programming");

return 0;

}

2. Write a program in C to concatenate first n characters of a string with another string.  
C에 프로그램을 작성하여 문자열의 처음 n자를 다른 문자열과 연결한다.



#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <string.h>

int main()

{

char s1[10] = "world";

char s2[20] = "Hello";

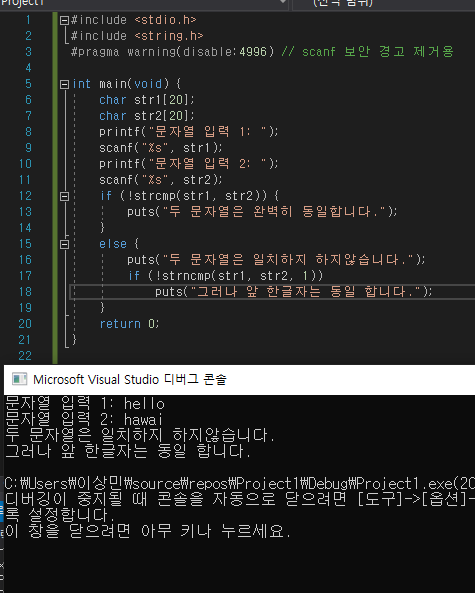
strncat(s2, s1,1);

printf("%s\n", s2);

return 0;

}

3. Write a program in C that compares first n characters of one string with first n characters of another string.  
C에서 한 문자열의 첫 번째 n자를 다른 문자열의 첫 번째 n자와 비교하는 프로그램을 작성한다.



#include <stdio.h>

#include <string.h>

#pragma warning(disable:4996) // scanf 보안 경고 제거용

int main(void) {

char str1[20];

char str2[20];

printf("문자열 입력 1: ");

scanf("%s", str1);

printf("문자열 입력 2: ");

scanf("%s", str2);

if (!strcmp(str1, str2)) {

puts("두 문자열은 완벽히 동일합니다.");

}

else {

puts("두 문자열은 일치하지 하지않습니다.");

if (!strncmp(str1, str2, 1))

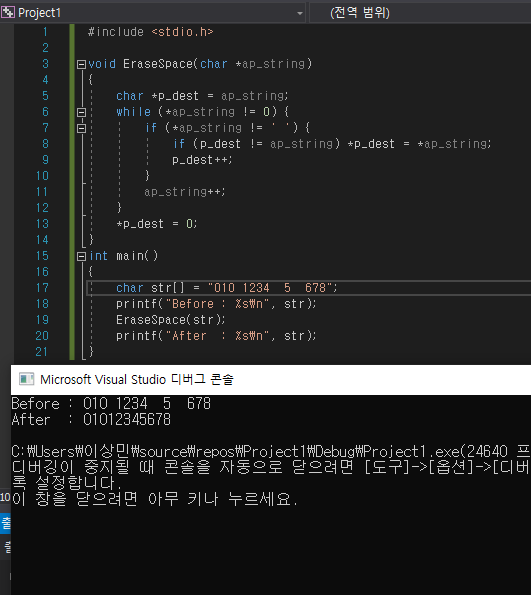
puts("그러나 앞 한글자는 동일 합니다.");

}

return 0;

}

4. Write a program in C that removes leading and trailing spaces from a string.  
문자열에서 선행과 후행 공백을 제거하는 프로그램을 C로 작성한다.



#include <stdio.h>

void EraseSpace(char \*ap\_string)

{

char \*p\_dest = ap\_string;

while (\*ap\_string != 0) {

if (\*ap\_string != ' ') {

if (p\_dest != ap\_string) \*p\_dest = \*ap\_string;

p\_dest++;

}

ap\_string++;

}

\*p\_dest = 0;

}

int main()

{

char str[] = "010 1234 5 678";

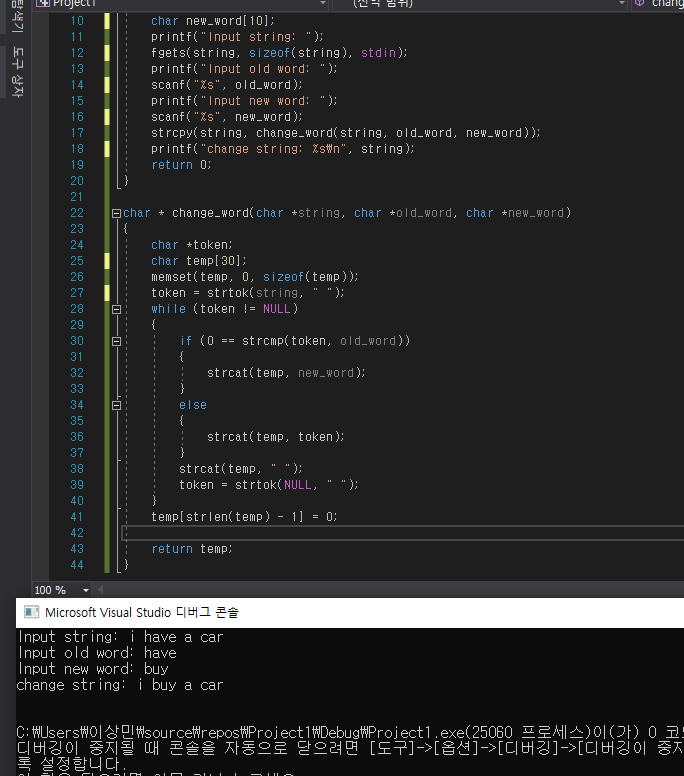
printf("Before : %s\n", str);

EraseSpace(str);

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

}

5. Write a program in C that replaces a given character with another character in a string.  
주어진 문자를 문자열의 다른 문자로 대체하는 프로그램을 C로 작성한다.



#include <stdio.h>

#include <string.h>

char \* change\_word(char \*string, char \*old\_word, char \*new\_word);

int main()

{

char string[30];

char old\_word[10];

char new\_word[10];

printf("Input string: ");

fgets(string, sizeof(string), stdin);

printf("Input old word: ");

scanf("%s", old\_word);

printf("Input new word: ");

scanf("%s", new\_word);

strcpy(string, change\_word(string, old\_word, new\_word));

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

return 0;

}

char \* change\_word(char \*string, char \*old\_word, char \*new\_word)

{

char \*token;

char temp[30];

memset(temp, 0, sizeof(temp));

token = strtok(string, " ");

while (token != NULL)

{

if (0 == strcmp(token, old\_word))

{

strcat(temp, new\_word);

}

else

{

strcat(temp, token);

}

strcat(temp, " ");

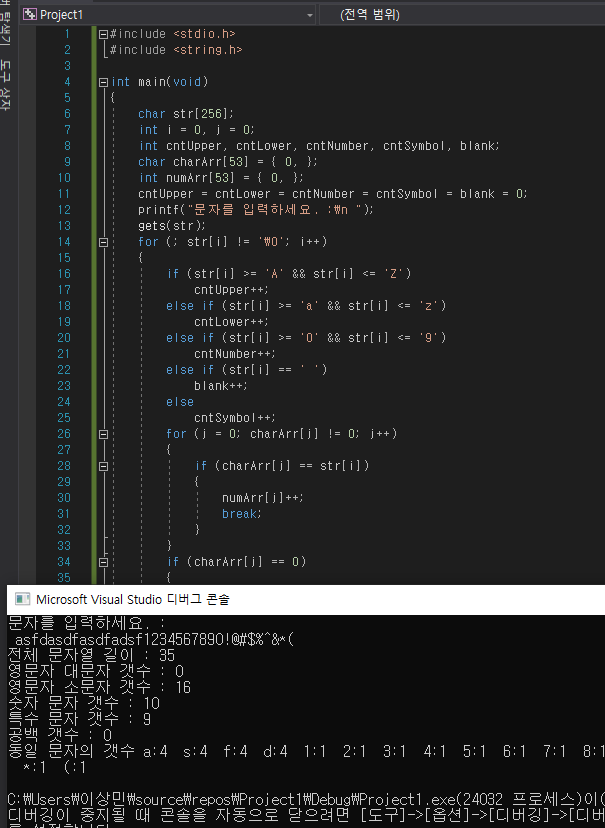
token = strtok(NULL, " ");

}

temp[strlen(temp) - 1] = 0;

return temp;

}

6. Write a program to count the number of digits, upper case characters, lower case characters, and special characters in a given string.  
주어진 문자열의 자릿수, 대문자, 소문자, 특수문자를 세는 프로그램을 작성한다.  
****

#include <stdio.h>

#include <string.h>

int main(void)

{

char str[256];

int i = 0, j = 0;

int cntUpper, cntLower, cntNumber, cntSymbol, blank;

char charArr[53] = { 0, };

int numArr[53] = { 0, };

cntUpper = cntLower = cntNumber = cntSymbol = blank = 0;

printf("문자를 입력하세요. :\n ");

gets(str);

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

{

if (str[i] >= 'A' && str[i] <= 'Z')

cntUpper++;

else if (str[i] >= 'a' && str[i] <= 'z')

cntLower++;

else if (str[i] >= '0' && str[i] <= '9')

cntNumber++;

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

blank++;

else

cntSymbol++;

for (j = 0; charArr[j] != 0; j++)

{

if (charArr[j] == str[i])

{

numArr[j]++;

break;

}

}

if (charArr[j] == 0)

{

charArr[j] = str[i];

numArr[j]++;

}

}

printf("전체 문자열 길이 : %d\n", i);

printf("영문자 대문자 갯수 : %d\n", cntUpper);

printf("영문자 소문자 갯수 : %d\n", cntLower);

printf("숫자 문자 갯수 : %d\n", cntNumber);

printf("특수 문자 갯수 : %d\n", cntSymbol);

printf("공백 갯수 : %d\n", blank);

printf("동일 문자의 갯수 ");

for (i = 0; charArr[i] != 0; i++)

{

if (charArr[i] == ' ')

{

printf("blank:%d ", numArr[i]);

}

else

{

printf("%c:%d ", charArr[i], numArr[i]);

}

}

printf("\n");

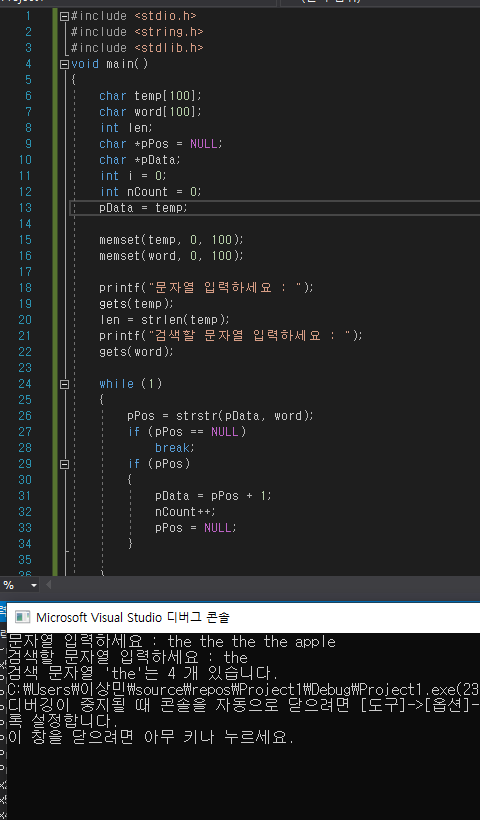
return 0;

}

7. Write a program to count the total number of occurrences of a given character in the string.  
문자열에서 주어진 캐릭터의 발생 횟수를 세는 프로그램을 작성한다.

위의 6번 문제에 함께 구현 하였습니다.

8. Write a program to accept a text. Count and display the number of times the word ‘the’ appears in the text.  
텍스트를 받아들이기 위한 프로그램을 작성한다. 텍스트에 'the'라는 단어가 나타나는 횟수를 세어 표시하십시오.

****

#include <stdio.h>

#include <string.h>

#include <stdlib.h>

void main()

{

char temp[100];

char word[100];

int len;

char \*pPos = NULL;

char \*pData;

int i = 0;

int nCount = 0;

pData = temp;

memset(temp, 0, 100);

memset(word, 0, 100);

printf("문자열 입력하세요 : ");

gets(temp);

len = strlen(temp);

printf("검색할 문자열 입력하세요 : ");

gets(word);

while (1)

{

pPos = strstr(pData, word);

if (pPos == NULL)

break;

if (pPos)

{

pData = pPos + 1;

nCount++;

pPos = NULL;

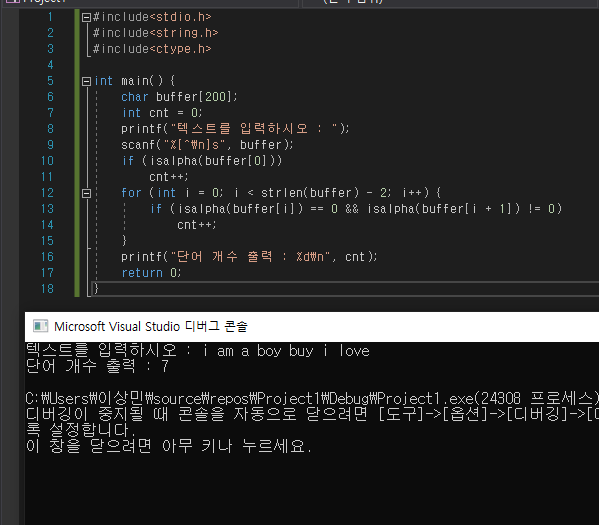
}

}

printf("검색 문자열 '%s'는 %d 개 있습니다.", word, nCount);

}

9. Write a program to count the total number of occurrences of a word in the text.  
본문에 있는 단어의 발생 총수를 세는 프로그램을 작성한다.

****

#include<stdio.h>

#include<string.h>

#include<ctype.h>

int main() {

char buffer[200];

int cnt = 0;

printf("텍스트를 입력하시오 : ");

scanf("%[^\n]s", buffer);

if (isalpha(buffer[0]))

cnt++;

for (int i = 0; i < strlen(buffer) - 2; i++) {

if (isalpha(buffer[i]) == 0 && isalpha(buffer[i + 1]) != 0)

cnt++;

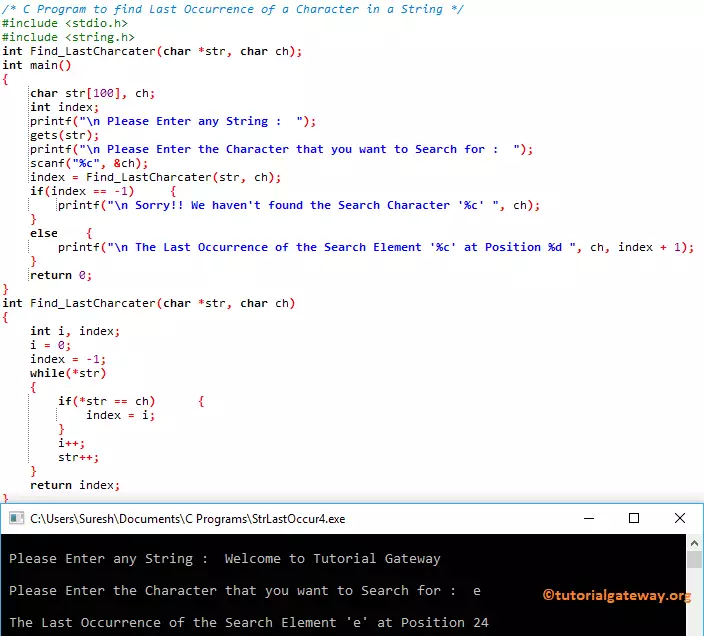
}

printf("단어 개수 출력 : %d\n", cnt);

return 0;

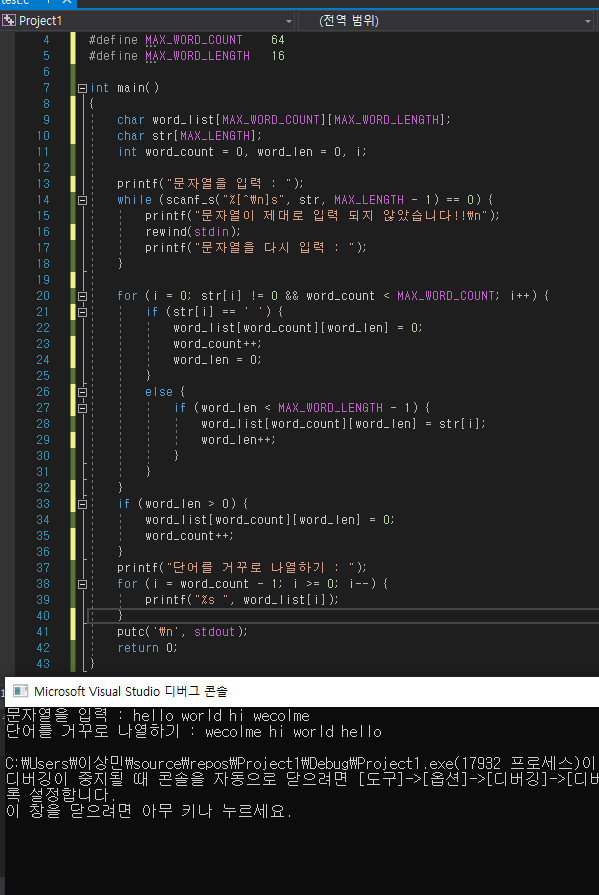
}

10. Write a program to find the last instance of occurrence of a sub-string within a string.  
프로그램을 작성하여 문자열 내에서 서브 문자열이 마지막으로 발생한 경우를 찾는다.



11. Write a program to input an array of strings. Then, reverse the string in the format shown below.  
일련의 문자열을 입력하는 프로그램을 작성한다. 그런 다음 아래 표시된 형식으로 문자열을 반전하십시오.

"HAPPY BIRTHDAY TO YOU" should be displayed as "YOU TO BIRTHDAY HAPPY"  
"Happy Birthday TO YOU"는 "YOU TO BEATHDAY HAPPY"로 표시되어야 한다.



#include <stdio.h>

#define MAX\_LENGTH 128

#define MAX\_WORD\_COUNT 64

#define MAX\_WORD\_LENGTH 16

int main()

{

char word\_list[MAX\_WORD\_COUNT][MAX\_WORD\_LENGTH];

char str[MAX\_LENGTH];

int word\_count = 0, word\_len = 0, i;

printf("문자열을 입력 : ");

while (scanf\_s("%[^\n]s", str, MAX\_LENGTH - 1) == 0) {

printf("문자열이 제대로 입력 되지 않았습니다!!\n");

rewind(stdin);

printf("문자열을 다시 입력 : ");

}

for (i = 0; str[i] != 0 && word\_count < MAX\_WORD\_COUNT; i++) {

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

word\_list[word\_count][word\_len] = 0;

word\_count++;

word\_len = 0;

}

else {

if (word\_len < MAX\_WORD\_LENGTH - 1) {

word\_list[word\_count][word\_len] = str[i];

word\_len++;

}

}

}

if (word\_len > 0) {

word\_list[word\_count][word\_len] = 0;

word\_count++;

}

printf("단어를 거꾸로 나열하기 : ");

for (i = word\_count - 1; i >= 0; i--) {

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

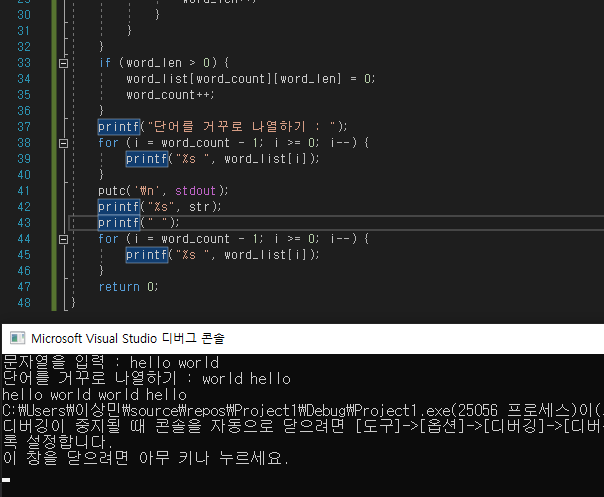
}

putc('\n', stdout);

return 0;

}

12. Write a program to append a given string in the following format.  
주어진 문자열을 다음과 같은 형식으로 추가하는 프로그램을 작성한다.  
"GOOD MORNING MORNING GOOD"



#include <stdio.h>

#define MAX\_LENGTH 128

#define MAX\_WORD\_COUNT 64

#define MAX\_WORD\_LENGTH 16

int main()

{

char word\_list[MAX\_WORD\_COUNT][MAX\_WORD\_LENGTH];

char str[MAX\_LENGTH];

int word\_count = 0, word\_len = 0, i;

printf("문자열을 입력 : ");

while (scanf\_s("%[^\n]s", str, MAX\_LENGTH - 1) == 0) {

printf("문자열이 제대로 입력 되지 않았습니다!!\n");

rewind(stdin);

printf("문자열을 다시 입력 : ");

}

for (i = 0; str[i] != 0 && word\_count < MAX\_WORD\_COUNT; i++) {

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

word\_list[word\_count][word\_len] = 0;

word\_count++;

word\_len = 0;

}

else {

if (word\_len < MAX\_WORD\_LENGTH - 1) {

word\_list[word\_count][word\_len] = str[i];

word\_len++;

}

}

}

if (word\_len > 0) {

word\_list[word\_count][word\_len] = 0;

word\_count++;

}

printf("단어를 거꾸로 나열하기 : ");

for (i = word\_count - 1; i >= 0; i--) {

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

}

putc('\n', stdout);

printf("%s", str);

printf(" ");

for (i = word\_count - 1; i >= 0; i--) {

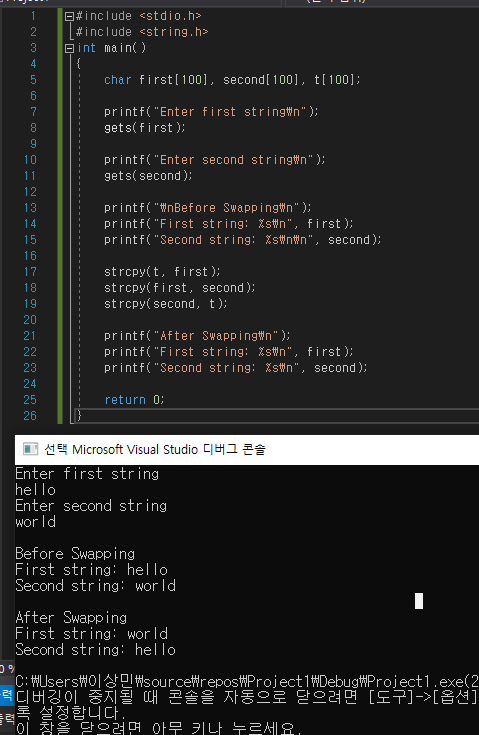
printf("%s ", word\_list[i]);

}

return 0;

}

13. Write a program to input a text of at least two paragraphs. Interchange the first and second paragraphs and then re-display the text on the screen.  
최소 두 단락의 텍스트를 입력하는 프로그램을 작성한다. 첫 번째 단락과 두 번째 단락을 교환한 다음 텍스트를 화면에 다시 표시하십시오.



#include <stdio.h>

#include <string.h>

int main()

{

char first[100], second[100], t[100];

printf("Enter first string\n");

gets(first);

printf("Enter second string\n");

gets(second);

printf("\nBefore Swapping\n");

printf("First string: %s\n", first);

printf("Second string: %s\n\n", second);

strcpy(t, first);

strcpy(first, second);

strcpy(second, t);

printf("After Swapping\n");

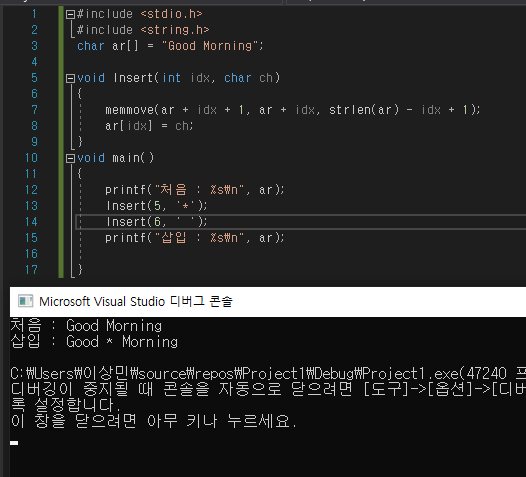
printf("First string: %s\n", first);

printf("Second string: %s\n", second);

return 0;

}

**14. Write a program to input a text of at least two paragraphs. Construct an array PAR such that PAR[I] contains the location of the it h paragraph in text.  
최소 두 단락의 텍스트를 입력하는 프로그램을 작성한다. PAR[I]에 해당 단락의 위치가 텍스트에 포함되도록 배열 PAR을 구성하십시오.**

****

**#include <stdio.h>**

**#include <string.h>**

**char ar[] = "Good Morning";**

**void Insert(int idx, char ch)**

**{**

**memmove(ar + idx + 1, ar + idx, strlen(ar) - idx + 1);**

**ar[idx] = ch;**

**}**

**void main()**

**{**

**printf("처음 : %s\n", ar);**

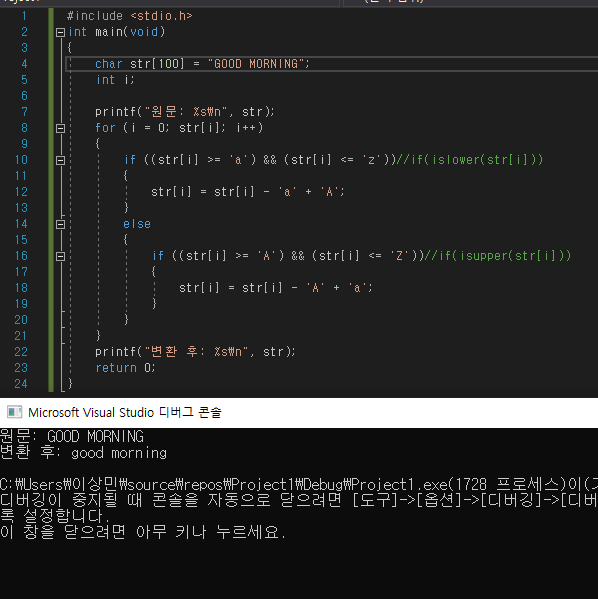
**Insert(5, '\*');**

**Insert(6, ' ');**

**printf("삽입 : %s\n", ar);**

**}**

15. Write a program to convert the given string "GOOD MORNING" to "good morning".  
주어진 문자열 "굿모닝"을 "굿모닝"으로 변환하는 프로그램을 작성한다.



#include <stdio.h>

int main(void)

{

char str[100] = "GOOD MORNING";

int i;

printf("원문: %s\n", str);

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

{

if ((str[i] >= 'a') && (str[i] <= 'z'))//if(islower(str[i]))

{

str[i] = str[i] - 'a' + 'A';

}

else

{

if ((str[i] >= 'A') && (str[i] <= 'Z'))//if(isupper(str[i]))

{

str[i] = str[i] - 'A' + 'a';

}

}

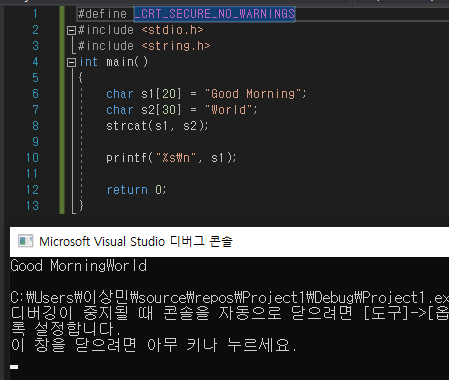
}

printf("변환 후: %s\n", str);

return 0;

}

16. Write a program to concatenate two given strings "Good Morning" and "World". Display the resultant string.  
주어진 두 줄 "굿모닝"과 "월드"를 연결하기 위한 프로그램을 작성하라. 결과 문자열을 표시하십시오.



#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <string.h>

int main()

{

char s1[20] = "Good Morning";

char s2[30] = "World";

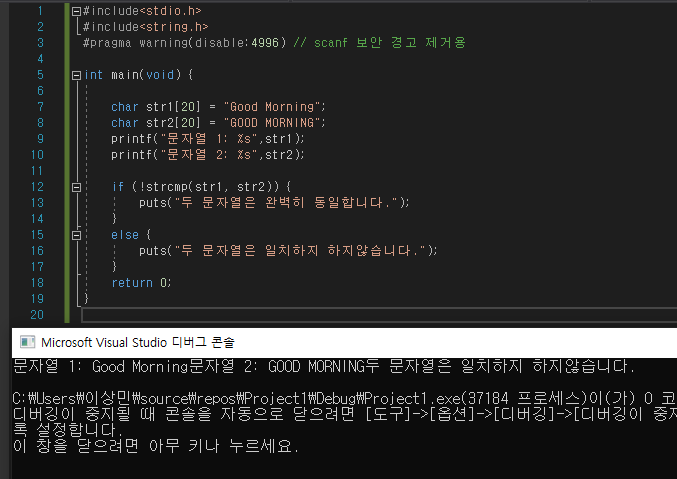
strcat(s1, s2);

printf("%s\n", s1);

return 0;

}

17. Write a program to check whether the two given strings "Good Morning" and "GOOD MORNING" are same.  
주어진 두 문자열 "굿모닝"과 "굿모닝"이 같은지 확인하는 프로그램을 작성한다.



#include<stdio.h>

#include<string.h>

#pragma warning(disable:4996) // scanf 보안 경고 제거용

int main(void) {

char str1[20] = "Good Morning";

char str2[20] = "GOOD MORNING";

printf("문자열 1: %s",str1);

printf("문자열 2: %s",str2);

if (!strcmp(str1, str2)) {

puts("두 문자열은 완벽히 동일합니다.");

}

else {

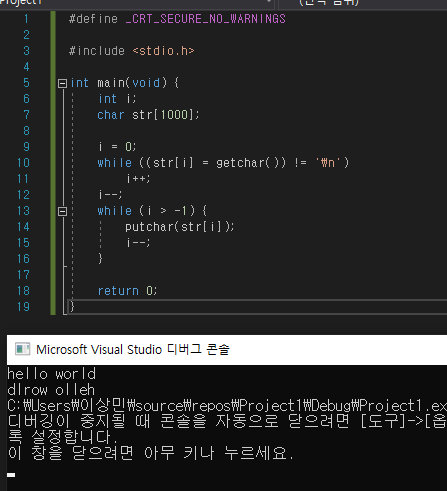
puts("두 문자열은 일치하지 하지않습니다.");

}

return 0;

}

18. Write a program to convert the given string "hello world" to "dlrow olleh".  
주어진 문자열 "hello world"를 "dlrow olleh"로 변환하는 프로그램을 작성한다.



#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

int main(void) {

int i;

char str[1000];

i = 0;

while ((str[i] = getchar()) != '\n')

i++;

i--;

while (i > -1) {

putchar(str[i]);

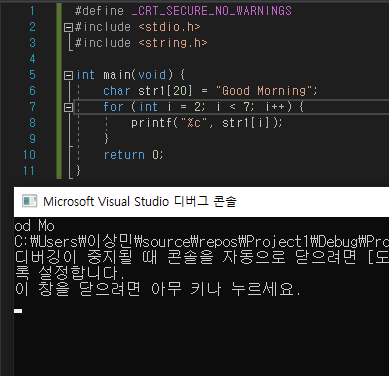
i--;

}

return 0;

}

19. Write a program to extract the string "od Mo" from the given string "Good Morning".  
주어진 문자열 "굿모닝"에서 문자열 "od Mo"를 추출하는 프로그램을 작성한다.



#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

#include <string.h>

int main(void) {

char str1[20] = "Good Morning";

for (int i = 2; i < 7; i++) {

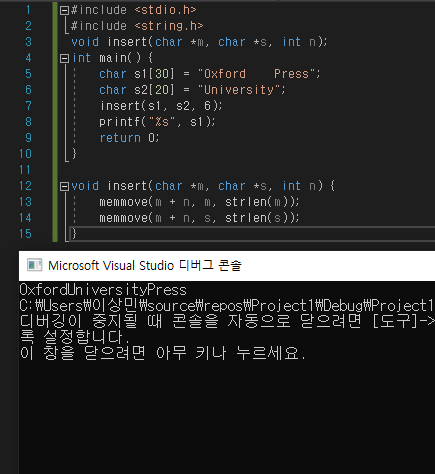
printf("%c", str1[i]);

}

return 0;

}

**20. Write a program to insert "University" in the given string "Oxford Press" so that the string should read as "Oxford University Press".  
주어진 문자열 "Oxford Press"에 "University"를 삽입하는 프로그램을 작성하여 해당 문자열이 "Oxford University Press"로 읽도록 한다.**

****

**#include <stdio.h>**

**#include <string.h>**

**void insert(char \*m, char \*s, int n);**

**int main() {**

**char s1[30] = "0xford Press";**

**char s2[20] = "University";**

**insert(s1, s2, 6);**

**printf("%s", s1);**

**return 0;**

**}**

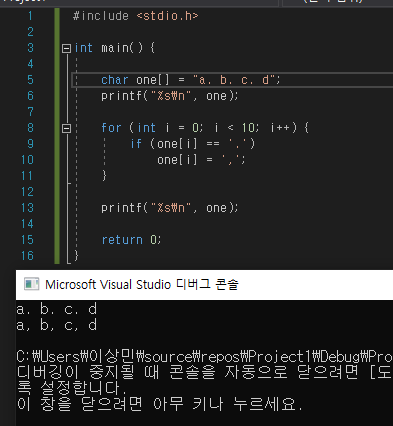
**void insert(char \*m, char \*s, int n) {**

**memmove(m + n, m, strlen(m));**

**memmove(m + n, s, strlen(s));**

**}**

21. Write a program to read a text, delete all the semi- colons it has, and finally replace all '.' with a ','.  
텍스트를 읽을 수 있는 프로그램을 작성하고, 그것이 가지고 있는 세미콜론을 모두 삭제하고, 마지막으로 모든 '.'을 ','로 대체한다.



#include <stdio.h>

int main() {

char one[] = "a. b. c. d";

printf("%s\n", one);

for (int i = 0; i < 10; i++) {

if (one[i] == '.')

one[i] = ',';

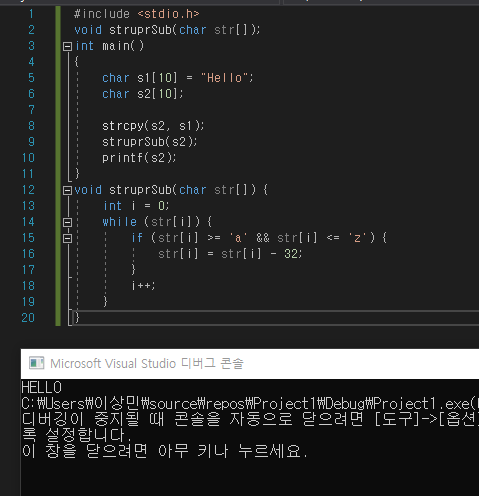
}

printf("%s\n", one);

return 0;

}

**22. Write a program to copy the last n characters of a character array in another character array. Also, convert the lower case letters into upper case letters while copying.  
다른 문자 배열에서 문자 배열의 마지막 n자를 복사하는 프로그램을 작성한다. 또한 복사하는 동안 소문자를 대문자로 변환하십시오**.



#include <stdio.h>

void struprSub(char str[]);

int main()

{

char s1[10] = "Hello";

char s2[10];

strcpy(s2, s1);

struprSub(s2);

printf(s2);

}

void struprSub(char str[]) {

int i = 0;

while (str[i]) {

if (str[i] >= 'a' && str[i] <= 'z') {

str[i] = str[i] - 32;

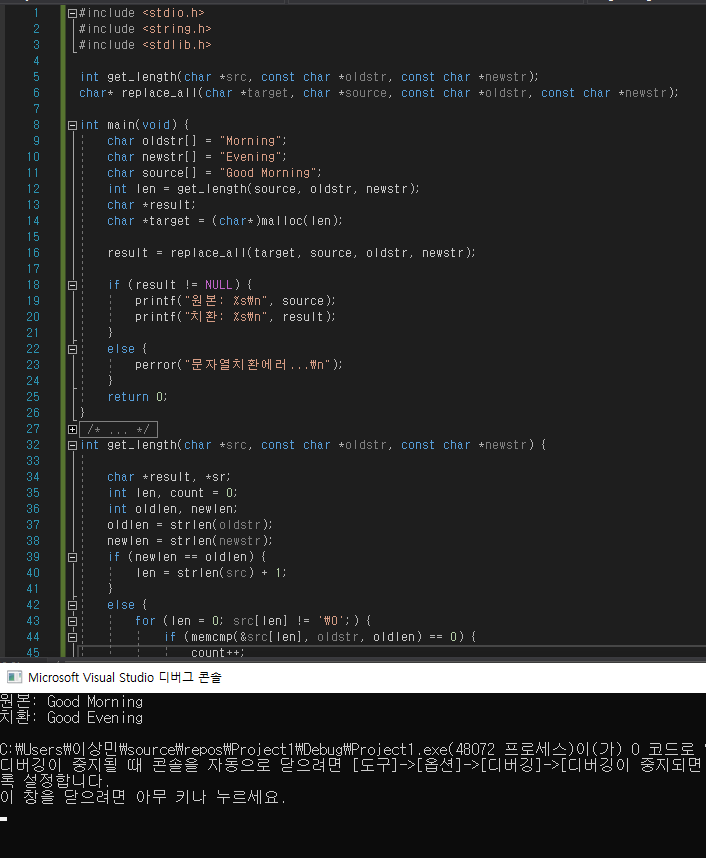
}

i++;

}

}

23. Write a program to rewrite the string "Good Morning" to "Good Evening".  
"Good Morning"이라는 string을 "Good Evening"으로 다시 쓰는 프로그램을 작성한다.



#include <stdio.h>

#include <string.h>

#include <stdlib.h>

int get\_length(char \*src, const char \*oldstr, const char \*newstr);

char\* replace\_all(char \*target, char \*source, const char \*oldstr, const char \*newstr);

int main(void) {

char oldstr[] = "Morning";

char newstr[] = "Evening";

char source[] = "Good Morning";

int len = get\_length(source, oldstr, newstr);

char \*result;

char \*target = (char\*)malloc(len);

result = replace\_all(target, source, oldstr, newstr);

if (result != NULL) {

printf("원본: %s\n", source);

printf("치환: %s\n", result);

}

else {

perror("문자열치환에러...\n");

}

return 0;

}

int get\_length(char \*src, const char \*oldstr, const char \*newstr) {

char \*result, \*sr;

int len, count = 0;

int oldlen, newlen;

oldlen = strlen(oldstr);

newlen = strlen(newstr);

if (newlen == oldlen) {

len = strlen(src) + 1;

}

else {

for (len = 0; src[len] != '\0';) {

if (memcmp(&src[len], oldstr, oldlen) == 0) {

count++;

len += oldlen;

}

else {

len++;

}

}

len = strlen(src) + 1 + count \* (newlen - oldlen);

}

return len;

}

char\* replace\_all(char \*target, char \*source, const char \*oldstr, const char \*newstr) {

char \*src, \*tar;

int oldlen = strlen(oldstr);

int newlen = strlen(newstr);

src = source;

tar = target;

while (\*src) {

if (memcmp(src, oldstr, oldlen) == 0) {

memcpy(tar, newstr, newlen);

tar += newlen;

src += oldlen;

}

else {

\*tar++ = \*src++;

}

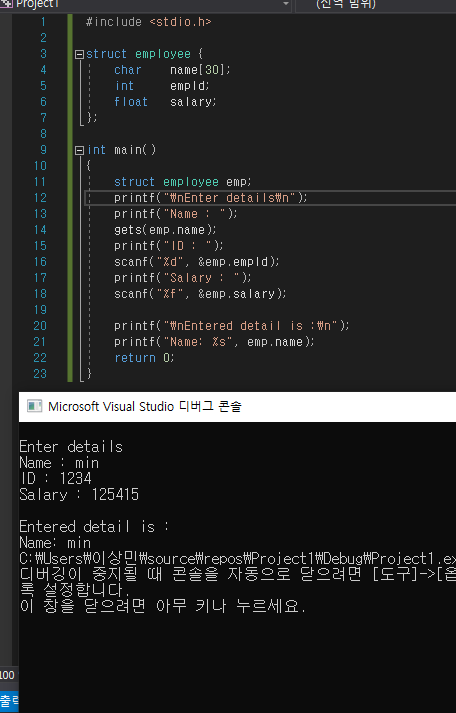
}

\*tar = '\0';

return target;

}

24. Write a program to read and display names of employees in a department.  
부서 직원의 이름을 읽고 표시하는 프로그램을 작성한다.



#include <stdio.h>

struct employee {

char name[30];

int empId;

float salary;

};

int main()

{

struct employee emp;

printf("\nEnter details\n");

printf("Name : ");

gets(emp.name);

printf("ID : ");

scanf("%d", &emp.empId);

printf("Salary : ");

scanf("%f", &emp.salary);

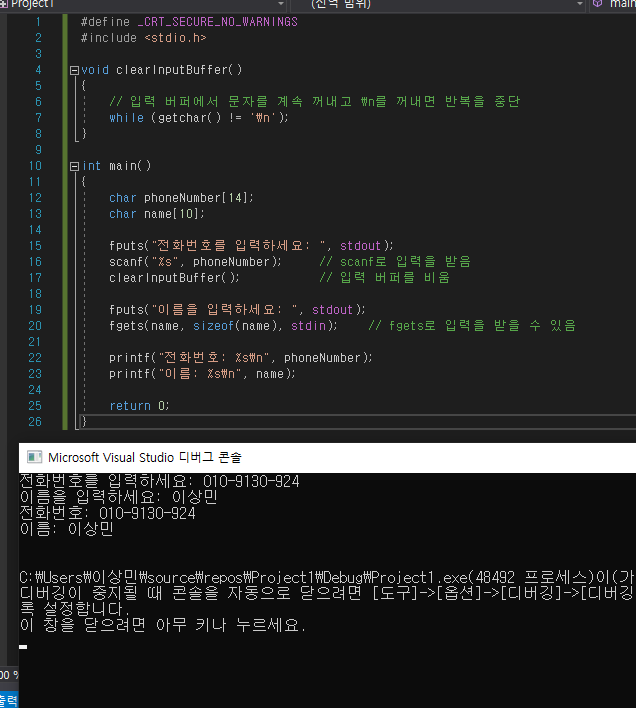
printf("\nEntered detail is :\n");

printf("Name: %s", emp.name);

return 0;

}

25. Write a program to read a line until a newline is entered.  
새로운 라인이 입력될 때까지 줄을 읽을 수 있는 프로그램을 작성한다.



#define \_CRT\_SECURE\_NO\_WARNINGS

#include <stdio.h>

void clearInputBuffer()

{

// 입력 버퍼에서 문자를 계속 꺼내고 \n를 꺼내면 반복을 중단

while (getchar() != '\n');

}

int main()

{

char phoneNumber[14];

char name[10];

fputs("전화번호를 입력하세요: ", stdout);

scanf("%s", phoneNumber); // scanf로 입력을 받음

clearInputBuffer(); // 입력 버퍼를 비움

fputs("이름을 입력하세요: ", stdout);

fgets(name, sizeof(name), stdin); // fgets로 입력을 받을 수 있음

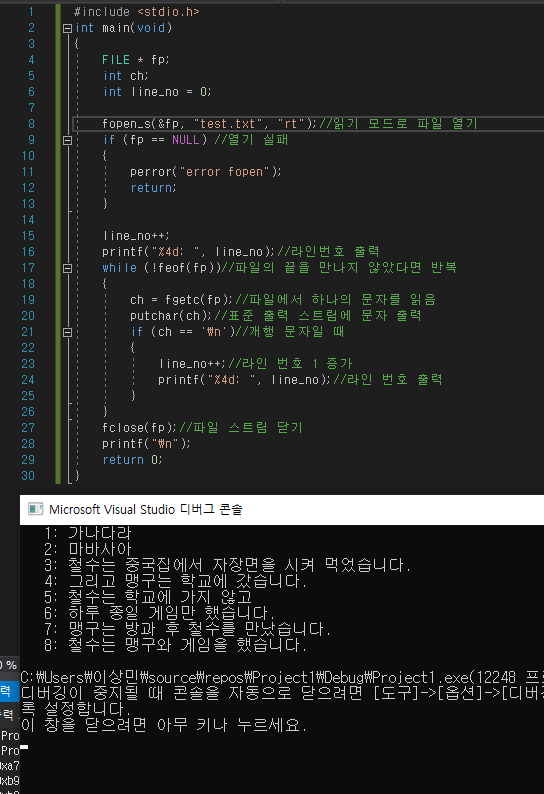
printf("전화번호: %s\n", phoneNumber);

printf("이름: %s\n", name);

return 0;

}

26. Write a program to read a short story. Rewrite the story by printing the line number before the starting of each line.  
단편소설을 읽을 수 있는 프로그램을 작성하라. 각 행의 시작 전에 행 번호를 인쇄하여 이야기를 다시 쓰십시오.



#include <stdio.h>

int main(void)

{

FILE \* fp;

int ch;

int line\_no = 0;

fopen\_s(&fp, "test.txt", "rt");//읽기 모드로 파일 열기

if (fp == NULL) //열기 실패

{

perror("error fopen");

return;

}

line\_no++;

printf("%4d: ", line\_no);//라인번호 출력

while (!feof(fp))//파일의 끝을 만나지 않았다면 반복

{

ch = fgetc(fp);//파일에서 하나의 문자를 읽음

putchar(ch);//표준 출력 스트림에 문자 출력

if (ch == '\n')//개행 문자일 때

{

line\_no++;//라인 번호 1 증가

printf("%4d: ", line\_no);//라인 번호 출력

}

}

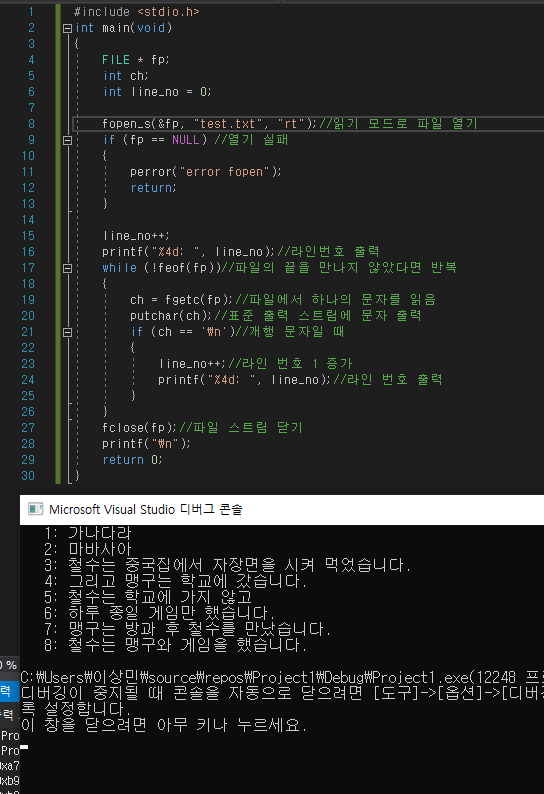
fclose(fp);//파일 스트림 닫기

printf("\n");

return 0;

}

27. Write a program to enter a text that contains multiple lines. Display the n lines of text starting from the mth line.  
여러 줄이 들어 있는 텍스트를 입력하는 프로그램을 작성한다. m번째 줄에서 시작하는 n개의 텍스트 행을 표시하십시오.



#include <stdio.h>

int main(void)

{

FILE \* fp;

int ch;

int line\_no = 0;

fopen\_s(&fp, "test.txt", "rt");//읽기 모드로 파일 열기

if (fp == NULL) //열기 실패

{

perror("error fopen");

return;

}

line\_no++;

printf("%4d: ", line\_no);//라인번호 출력

while (!feof(fp))//파일의 끝을 만나지 않았다면 반복

{

ch = fgetc(fp);//파일에서 하나의 문자를 읽음

putchar(ch);//표준 출력 스트림에 문자 출력

if (ch == '\n')//개행 문자일 때

{

line\_no++;//라인 번호 1 증가

printf("%4d: ", line\_no);//라인 번호 출력

}

}

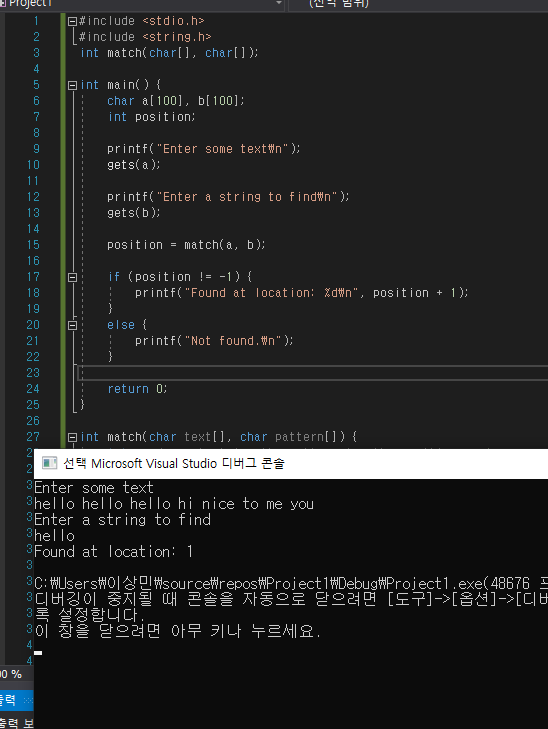
fclose(fp);//파일 스트림 닫기

printf("\n");

return 0;

}

28. Write a program to check whether a pattern exists in a text. If it does, delete the pattern and display it.  
텍스트에 패턴이 존재하는지 확인하는 프로그램을 작성한다. 이 경우 패턴을 삭제하고 표시하십시오.



#include <stdio.h>

#include <string.h>

int match(char[], char[]);

int main() {

char a[100], b[100];

int position;

printf("Enter some text\n");

gets(a);

printf("Enter a string to find\n");

gets(b);

position = match(a, b);

if (position != -1) {

printf("Found at location: %d\n", position + 1);

}

else {

printf("Not found.\n");

}

return 0;

}

int match(char text[], char pattern[]) {

int c, d, e, text\_length, pattern\_length, position = -1;

text\_length = strlen(text);

pattern\_length = strlen(pattern);

if (pattern\_length > text\_length) {

return -1;

}

for (c = 0; c <= text\_length - pattern\_length; c++) {

position = e = c;

for (d = 0; d < pattern\_length; d++) {

if (pattern[d] == text[e]) {

e++;

}

else {

break;

}

}

if (d == pattern\_length) {

return position;

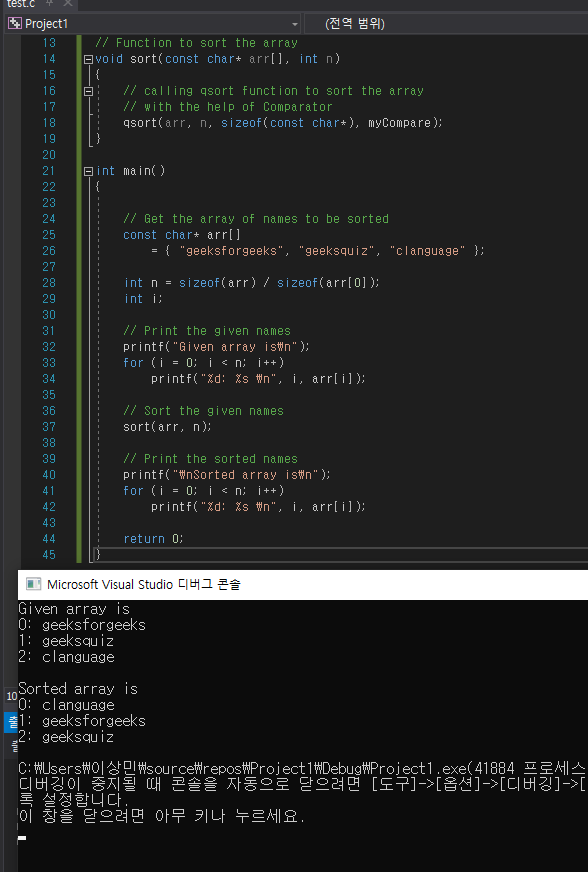
}

}

return -1;

}

29. Write a program to insert a new name in the string array STUD[][], assuming that names are sorted alphabetically.  
이름이 알파벳 순으로 정렬된다고 가정하여 문자열 배열 STUD[][]에 새 이름을 삽입하는 프로그램을 작성한다.



#include <stdio.h>

#include <stdlib.h>

#include <string.h>

// Defining comparator function as per the requirement

static int myCompare(const void\* a, const void\* b)

{

// setting up rules for comparison

return strcmp(\*(const char\*\*)a, \*(const char\*\*)b);

}

// Function to sort the array

void sort(const char\* arr[], int n)

{

// calling qsort function to sort the array

// with the help of Comparator

qsort(arr, n, sizeof(const char\*), myCompare);

}

int main()

{

// Get the array of names to be sorted

const char\* arr[]

= { "geeksforgeeks", "geeksquiz", "clanguage" };

int n = sizeof(arr) / sizeof(arr[0]);

int i;

// Print the given names

printf("Given array is\n");

for (i = 0; i < n; i++)

printf("%d: %s \n", i, arr[i]);

// Sort the given names

sort(arr, n);

// Print the sorted names

printf("\nSorted array is\n");

for (i = 0; i < n; i++)

printf("%d: %s \n", i, arr[i]);

return 0;

}

30. Write a program to delete a name in the string array STUD[][], assuming that names are sorted alphabetically.  
이름이 알파벳 순으로 정렬된다고 가정하여 문자열 배열 STUD[][]에서 이름을 삭제하는 프로그램을 작성한다.

**#define \_CRT\_SECURE\_NO\_WARNINGS**

**#include <stdio.h>**

**#include <time.h>**

**void prt\_menu();**

**int main(void) {**

**int data[7][10] = { NULL };**

**int i, j;**

**int key;**

**int a, b, c;**

**srand(time(NULL));**

**for (j = 0; j < 7; j++) {**

**for (i = 0; i < 10; i++) {**

**data[j][i] = rand() % 500 + 1;**

**if (j == 6 && i == 0) break;**

**}**

**if (j == 6 && i == 0) break;**

**}**

**for (j = 0; j < 7; j++) {**

**for (i = 0; i < 10; i++) {**

**printf("%5d", data[j][i]);**

**}**

**printf("\n");**

**}**

**do {**

**do {**

**prt\_menu();**

**scanf("%d", &key);**

**} while (key > 3 || key < 1);**

**if (key == 1) {**

**do {**

**printf("삽입할 좌표 (행,열): ");**

**scanf("%d,%d", &a, &b);**

**} while (a > 6 || a < 0 || b>9 || b < 0);**

**printf("삽입할 값 : ");**

**scanf("%d", &c);**

**for (j = 6; j > 0; j--) {**

**for (i = 9; i > 0; i--) {**

**if (data[j][i] != NULL) {**

**if (i == 9) data[j + 1][0] = data[j][i];**

**else data[j][i + 1] = data[j][i];**

**if (i == (b - 1) && j == (a - 1)) break;**

**}**

**}**

**if (i == (b - 1) && j == (a - 1)) break;**

**}**

**data[6][1] = 0;**

**data[a - 1][b - 1] = c;**

**for (j = 0; j < 7; j++) {**

**for (i = 0; i < 10; i++) {**

**printf("%5d", data[j][i]);**

**}**

**printf("\n");**

**}**

**}**

**if (key == 2) {**

**printf("삭제할 좌표 (행,열): ");**

**scanf("%d,%d", &a, &b);**

**for (j = 0; j < 7; j++) {**

**for (i = 0; i < 10; i++) {**

**if (i < (b - 1) && j == (a - 1)) continue;**

**if (data[j][i] != NULL) {**

**if (j >= a - 1) {**

**if (i == 9) data[j][i] = data[j + 1][0];**

**else data[j][i] = data[j][i + 1];**

**}**

**}**

**}**

**}**

**for (j = 0; j < 7; j++) {**

**for (i = 0; i < 10; i++) {**

**printf("%5d", data[j][i]);**

**}**

**printf("\n");**

**}**

**}**

**} while (key != 3);**

**}**

**void prt\_menu() {**

**printf("\n\n\tmenu select\n");**

**printf("\t1.삽입");**

**printf("\t2.삭제");**

**printf("\t3.종료");**

**printf("\tinput[ ]\b\b");**

**}**

**Multiple-choice Questions**

1. Insert("XXXYYYZZZ", 1, "PPP") =

(a) PPPXXXYYYZZZ **(b) XPPPXXYYYZZZ** (c) XXXYYYZZZPPP

2. Delete("XXXYYYZZZ", 4,3) =

(a) XXYZ (b) XXXYYZZ **(c) XXXYZZ**

3. If str[] = "Welcome to the world of programming", then SUBSTRING(str, 15, 5) =

**(a) world** (b) programming (c) welcome (d) none of these

4. strcat() is defined in which header file?

(a) ctype.h (b) stdio.h **(c) string.h** (d) math.h

5. A string can be read using which function(s)?

(a) gets() (b) scanf() (c) getchar() **(d) all of these**

6. Replace("XXXYYYZZZ", "XY", "AB") =

**(a) XXABYYZZZ** (b) XABYYYZZZ (c) ABXXXYYYZZ

7. The index of U in Oxford University Press is?

(a) 5 (b) 6 **(c) 7** (d) 8

8. s1 = “HI”, s2 = “HELLO”, s3 = “BYE”. How can we concatenate the three strings?

(a) strcat(s1,s2,s3) **(b) strcat(s1(strcat(s2,s3)))** (c) strcpy(s1, strcat(s2,s3))

9. strlen(“Oxford University Press”) is ?

(a) 22 **(b) 23** (c) 24 (d) 25

10. Which function adds a string to the end of another string?

(a) stradd() **(b) strcat()** (c) strtok() (d) strcpy()

**True or False**

1. String Hello World can be read using scanf(). **: False**

2. A string when read using scanf() needs an ampersand character. **: False**

3. The gets() function takes the starting address of a string which will hold the input. **: True**

4. tolower() is defined in ctype.h header file. **: True**

5. If S, and S, are two strings, then the concatenation operation produces a string which contains the characters of S, followed by the characters of S. **: False**

6. Appending one string to another string involves copying the contents of the source string at the end of the destination string. **: False**

7. S1<S2, when in dictionary order, S1 precedes S2. **: True**

8. If S1 = "GOOD MORNING", then Substr\_Right (S1, 5) = MORNING. **: False**

9. Replace ("AAABBBCCC", "X", "YYY")= AAABBBCC. **: True**

10. Initializing a string as char str[] = "HELLO"; is incorrect as a null character has not been explicitly added. **: False**

11. The scanf() function automatically appends a null character at the end of the string read from the keyboard. **: True**

12. String variables can be present either on the left or on the right side of the assignment operator. **: False**

13. When a string is initialized during its declaration, the string must be explicitly terminated with a null character. **: True**

14. strcmp("and", "ant"); will return a positive value. **: False**

15. Assignment operator can be used to copy the contents of one string into another. **: False**

**Fill in the blanks**

1. Strings are \_\_\_\_\_\_. **: A null-terminated character array**

2. Every string is terminated with a \_\_\_\_\_\_. **: Null character**

3. If a string is given as "AB CD", the length of this string is **\_\_\_\_\_\_. : 5**

4. The subscript of a string starts with \_\_\_\_\_\_. **: zero**

5. Characters of a string are stored in \_\_\_\_\_\_ memory locations**. : Consecutive**

6. char mesg[100]; can store a maximum of \_\_\_\_\_\_ characters. **: 99**

7. \_\_\_\_\_\_ function terminates as soon as it finds a blank space. **: scanf()**

8. The ASCII code for A–Z varies from \_\_\_\_\_\_. **: 65-97**

9. toupper() is used to **\_\_\_\_\_\_. : Convert a character into upper case**

10. S1>S2 means \_\_\_\_\_\_. **: When in dictionary order S1 will come after S2**

11. The function to reverse a string is **\_\_\_\_\_\_. : strrev()**

12. If S1 = "GOOD MORNING", then Substr\_Left (S1, 7) = **\_\_\_\_\_\_. : Morning**

13. INDEX("Welcome to the world of programming", "world") = **\_\_\_\_\_\_. : 15**

14. \_\_\_\_\_\_ returns the position in the string where the string pattern first occurs. **: Index operation**

15. strcmp(str1, str2) returns 1 if \_\_\_\_\_\_\_\_\_\_\_. **: str2 is less than str1**

16. \_\_\_\_\_\_\_\_\_\_\_ function computes the length of a string**. : strlen**

17. Besides printf(), \_\_\_\_\_\_\_\_\_\_\_ function can be used to print a line of text on the screen**. : puts**