프논 13th 수업에서 다루어지는 것들: 문자열(스트링, string)

프논에서 다룬 내용

- **문자 vs 문자열**
  문자(character): 각각의 문자에 숫자코드(아스키코드)가 할당되어 있다. 예: 'A'(65), 'a'(97), '0'( )
  문자열(string): 문자들이 여러 개 모인 것

- **문자열 상수 vs 문자열 변수**
  문자열 상수: "A", "Hello World"
  문자열 변수: char str[100]; 로 선언하고 값을 넣을 때 마지막에 '\0'이 들어가야 문자열 변수

- **문자열 vs 문자형 배열**
  문자열은 끝에는 항상 NULL문자('\0')가 들어가는 문자형 배열
  NULL은 문자열의 끝을 알려준다.

- **문자열 변수의 선언 및 초기화**

```
char str[10];
char str1[10] = {'a', 'B', 'c', 'D', 'e', '\0'};
char str2[10] = "aBcDe"; // 자동으로 '\0'을 붙여준다

char str3[] = "aBcDe"; // str3[6]로 자동 설정된다
```

- **문자열 변수의 입력**

k

```
scanf("%s", str); // 문자열 변수의 입력에서는 &를 붙이지 않는다. 사실은 str ≡ &str[0]
```

- **문자열 변수의 출력**

방법 1: *printf("**%s**", str);*

방법 2:
```
for (i = 0; str[i] != '\0'; i++)
        printf("%c", str[i]);
```

- **문자열 변수 안의 문자들을 처리**

```
char str[] = "aBcDe";

for(i = 0; str[i] != '\0'; i++)
   한 문자씩 처리;
```

☐ 연습 1: str의 홀수번 째 문자를 출력
실행예: ace

```
for(i = 0; str[i] != '\0'; i++)
      if (i % 2 == 0) // 0부터 시작하므로 0, 2, 4가 홀수번째
            printf("%d", str[i]);
```

☐ 연습2: str안에 대문자를 출력
실행예: BD

```
for(i = 0; str[i] != '\0'; i++)
      if (str[i] >= 'A' && str[i] <= 'Z')
            printf("%d", str[i]);
```

☐ 연습2-2: str안에 대문자를 가지고 str2를 만들어 str2를 출력
실행예: BD

```
j = 0; // 별도의 인덱스가 필요하다
for(i = 0; str[i] != '\0'; i++)
      if (str[i] >= 'A' && str[i] <= 'Z')
            str2[j++] = str[i];
str2[j] = '\0';
printf("%s", str2);
```


☐ 연습3: str안에 대문자를 소문자로 소문자를 대문자로 나머지 문자는 그대로 출력
실행예: AbCdE

프논에서 다룬 내용

```
for(i = 0; str[i] != '\0'; i++)
      if (str[i] >= 'A' && str[i] <= 'Z')
            printf("%d", str[i] + 32);
      else if (str[i] >= 'a' && str[i] <= 'z')
            printf("%d", str[i] – 32);
      else
            printf("%d", str[i]);
```

☐ 연습3-2: str안에 대문자를 소문자로 소문자는 대문자로 나머지 문자는 그대로 str2를 만들어 str2를 출력
실행예: AbCdE

```
for(i = 0; str[i] != '\0'; i++)
      if (str[i] >= 'A' && str[i] <= 'Z')
            str2[i] = str[i] + 32;
      else if (str[i] >= 'a' && str[i] <= 'z')
            str2[i] = str[i] - 32;
      else
            str2[i] = str[i];
str2[i] = '\0';
printf("%s", str2);
```

☐ 연습4: 문자열 str의 길이를 출력
실행예: 5

```
for(i = 0; str[i] != '\0'; i++); // body가 없는 for문

printf("%d", i);
```

☐ 연습5: str을 거꾸로 출력
실행예: eDcBa

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

// i는 str의 길이
for(j = i – 1; j >= 0; j--)
      printf("%d", str[j]);
```

☐ 연습5-2: str을 거꾸로 한 문자열을 str2에 넣어서 str2를 출력
실행예: eDcBa

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

// i는 str의 길이
for(j = i – 1; j >= 0; j--)
      str2[i – 1 - j] = str[j]);
str2[i] = '\0';
printf("%s", str2);
```

- **문자열의 함수 매개변수 전달**
  주어진 문자열에서 대문자를 출력하는 프로그램을 완성하라.

실행예:
CD

```
#include <stdio.h>
void printUpperCase(char s[]);
int main(void)
{
      Char str[10] = "abCDe";
      printUpperCase(str); // 문자열 변수의 이름
}
void printUpperCase(char s[]) // 배열의 경우는 크기를 같이 전달했으나, 문자열은 그럴 필요 없음. 왜?
{
      int i;
      for (i = 0; s[i] != '\0'; i++)
            if (s[i] >= 'A' && s[i] <= 'Z')
                  printf("%d", s[i]);
}
```

LAB 6

프논에서 다룬 내용

☞**오늘의 실습숙제에서**
(수업시간에 배우지 않은) 문자열 처리함수(strlen, strcmp 등)나
문자처리함수(isupper, islower등)는 사용하지 않는다.

■ **LAB6_0**(문자열의 정의, 표준출력, 문자열 종료방법) 다음 예제 프로그램의 결과 예상해 보세요. 그리고 프로그램의 실행한 후 자신의 예상 결과와 비교해 보세요. 1)2)3)4)를 잘 살펴보라.

```
#include <stdio.h>
int main(void)
{
	int i;

	// 1) 문자열을 정의하는 여러가지 방법들입니다
	char digits[] = "0123456789";
	char abc[] = {'A', 'B', 'C', '\0'};
	char lan[5] = "java";

	// 2) 출력 결과는?
	printf("%s %s\n", digits, abc); //%s를 사용한다.

	// 3) 아래처럼 lan 문자열변수를 두가지 방법으로 출력할 수있다. %c와 %s의 쓰임새에 주목하라.
	printf("%s\n", lan);

	for (i = 0; lan[i] != '\0'; i++) // A)문자열의 마지막을 확인하는 방법을 유의해보라
		printf("%c", lan[i]);

	// 4) digits[]의 중간에 null character를 삽입하면
	digits[6] = '\0';
	printf("\n%s\n", digits);

	return 0;
}
```

■ **LAB6_a**

□ 하나의 문자열 변수 word를 읽어들여서(%s사용) 이를 출력(%s사용)하는 간단한 프로그램을 작성하라. word의 길이는 최대 10이라 가정하자. 즉 char word[11]; 이라 선언하여 사용하면 된다.

실행예:
```
Enter a word(<= 10 chars): Happy
Happy
```

□ 위의 프로그램을 수정하여 출력할 때 word안의 문자를 하나하나를 출력하도록 하라. 즉, 실행예는 그대로이다.(위의 LAB6_0의 A)부분을 참조하라)

□ 다시 위의 프로그램을 수정하여 word의 요소중 홀수번째(즉, 첫번째, 세번째,…)의 문자를 출력하도록 하라.
실행예
```
Enter a word(<= 10 chars): Happy
Hpy
```

실행예
```
Enter a word(<= 10 chars): Love
Lv
```

■ **LAB6_1**(문자열 처리) 문자열(최대 길이 80라 가정하자)을 입력받아서, 입력받은 문자열의 길이를 출력하고 그 문자열을 뒤에서부터 한줄에 한 단어씩 출력하는 프로그램을 작성하라.

□ **LAB6_1_1(길이 구하기)** 먼저 문자열의 길이를 구하여 출력하는 프로그램을 작성하자.
실행예:

```
Enter a string: abcde
길이는 5
```

프논에서 다룬 내용

- s라는 이름의 문자열 변수를 정의하자. 크기는 얼마로 지정해야하는가.

- 문자열 변수의 문자들을 하나씩처리하기 위한 for문은 아래와 같다.
```
for(i = 0; s[i] != '\0'; i++)
```
아래와 같이 loop body부분 없는 for문을 수행하면 for문 이후 i는 어떤 값을 갖는 가를 생각해보라.
```
for(i = 0; s[i] != '\0'; i++);
```

- 자, 프로그램을 작성해보자.
```
#include <stdio.h>
int main(void)
{
	//필요한 변수

	// 문자열 변수 입력


	// 문자열 변수의 길이를 센다(for문 사용)


	// 길이를 출력한다.

}
```

□ **(길이를 이용하여 거꾸로 쓰기)** 위에서 구한 문자열의 길이를 이용하여 아래의 실행결과를 내도록 위의 프로그램을 수정하라.
실행예:

```
Enter a string: abcde
길이는 5
e
d
c
b
a
```

힌트: 길이(length)를 구하면 문자열의 마지막 요소부터의 출력이 가능하다.
예: abcde의 경우 길이는 5이고 s[4], s[3], s[2], s[1], s[0]으로 출력하면 된다.

□ **LAB6_1_2(문자 뒤집기 준비)**

프논에서 다룬 내용

```
#include <stdio.h>
int main(void)
{
	int i;
	char ch;
	char str[10] = "abcde"; // 1)

	printf("--변경 전 문자열--\n");
	printf("%s \n", str);

	/* 문자열 변경 */
	for(i=0; i < 2; i++) // 2)
	{
		ch = str[4 - i];  // 3)
		str[4 - i] = str[i]; // 4)
		str[i] = ch;
	}

	printf("\n--변경 후 문자열--\n");
	printf("%s \n", str);

	return 0;
}
```

✧ 프로그램을 이해할 수있는가? 메모리에 값저장 상태를 나타낸 후 **2)**의 for문에서 반복이 이루어 질때마다 메모리 상의 어떤 변화가 일어나는지 그려보라.

초기 상태

첫번째 반복: i가 0일 때

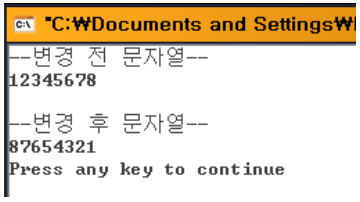
두번째 반복: i가 1일 때

✧ **1)**에 있는 초기화부분을 다음과 같이

char str[10] = "12345678" ;

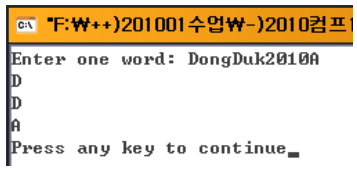
로 바꾼 후 아래의 실행결과가 나오도록 프로그램을 수정하라.

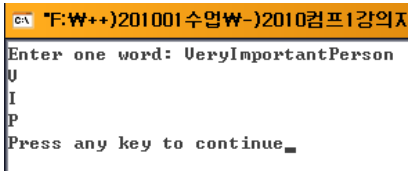
힌트: 라인 **2), 3), 4)**에서 숫자 상수만 바꾸면 된다.



■ **LAB6_2(문자열처리)**

□ **(LAB6_2_1)문자열을 입력받아서, 그 안에 존재하는 대문자 알파벳들을 출력하는 프로그램을 작성하라.**

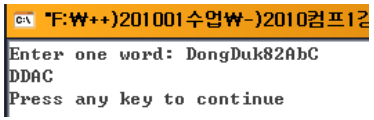


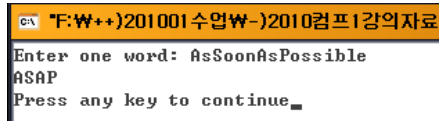



```
#include <stdio.h>
int main(void)
{
    //필요한 변수

    // 문자열 변수 입력


    // 문자열 변수안 문자를 하나씩 처리하면서 대문자 알파벳이면 출력한다. 대문자 알파벳인가를
    // 판별하기위해서 ASCII 값을 이용한다. 문자 'A'와 문자 'Z'의 ASCII값은 각각 65, 90이다.
    // 즉 65이상 90이하이면 대문자 알파벳 임을 알수있다.


}
```

프논에서 다룬 내용

□ **(LAB6_2_2)**위의 프로그램을 수정하여, 문자열 word를 입력받아서 word 안의 대문자들을 다른 문자열 변수 newWord에 넣은 후 이것을 출력하는 프로그램을 작성하라.

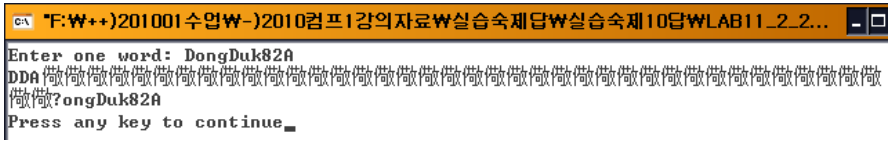




힌트 및 요구사항

- **newWord 문자열 변수를 위한 별도의 첨자, j(0으로 초기화)가 필요하다.**
- 출력시 아래와 같은 이상한 문자들이 올바른 실행결과 뒤에 따라나올 수있다. 그런 경우 왜일까를 생각해보라. 어떻게 해결할것인가?

<잘못된 실행예>




```
#include <stdio.h>
int main(void)
{
	char word[81], newWord[81];




	printf("%s\n", newWord);
}
```

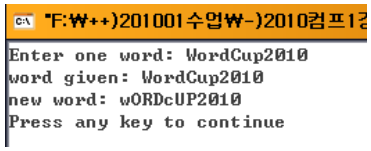
HW 6

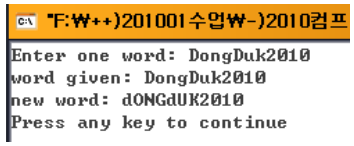
프논에서 다룬 내용

■ **HW6_1(문자열 처리) 하나의 문자열(80자이하)을 입력 받아서 문자열 안의 대문자는 소문자로, 소문자는 대문자로 바꾸는 프로그램을 작성하세요. 영문자 이외의 숫자나 기호는 변환하지 않습니다.**

힌트 및 요구사항

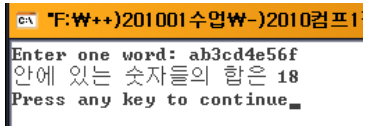
- 새로 변환한 문자열을 반드시 새로운 문자열 에 저장하라.
  즉, 입력 받을 문자열은 word, 새로 변환한 문자열은 newWord를 사용한다. 두 문자열 모두 char형이고 크기는 81로 한다.
- 대문자 알파벳과 소문자 알파벳은 각각 32의 차이값을 갖는다. 즉, 'A' 는 65이고 'a' 는 97 값을 갖는다. 그러므로
  = 대문자를 소문자로 바꾸는 방법: 대문자 + 32
  = 소문자를 대문자로 바꾸는 방법: 소문자 - 32
- 새로 변환한 문자열 출력시
  printf("%s", newWord)를 사용하라.

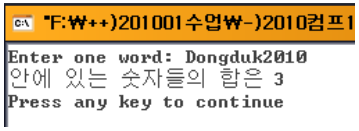




■ **HW6_2(문자열) 적당한 길이의 문자열을 입력받아서, 그 안에 존재하는 숫자들의 총합을 계산하여 출력하는 프로그램을 작성하라. '0' 의 아스키값은 48임을 활용하라.**

예를 들어, '1' 은 49값을 갖으므로 '1' - 48 계산을 통해서 1이라는 숫자 값을 얻을 수있음에 착안하라.





■ **HW6_3(문자열 palindrome만들기) 문자열(word)을 입력받아 그 속의 문자들을 거꾸로 하여 문자열(newWord)을 만들어 출력하는 프로그램을 작성하라.(LAB6_1을 참조)**

실행예:

Enter one word: abcde

The reversed word is edcba

프논에서 다룬 내용

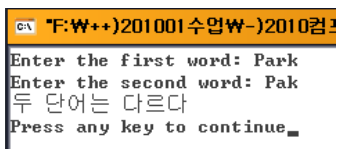
```
#include <stdio.h>
int main(void)
{
        char word[81], newWord[81];









        printf("%The reversed word is s\n", newWord);
}
```

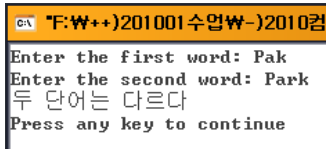
문자열은 다양한 문제에서 연습이 가능합니다.

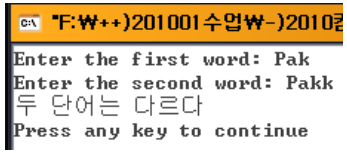
실제 프로그래밍에서 문자열을 다루는 연습은 아주 중요합니다.

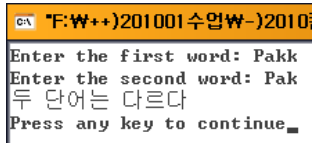
아래의 challenge 문제들은 수업뒤에서 연습문제로 사용될 수도 있습니다.

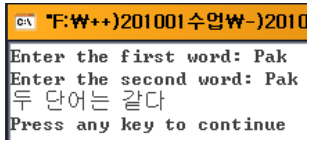
■ **challenge6a**(난이도 상) 두 개의 단어를 입력받아서 같은가 다른가를 판별하는 프로그램을 작성하라. 아래의 실행결과를 모두 실행시켜보아 프로그램의 완성도를 확인하라.









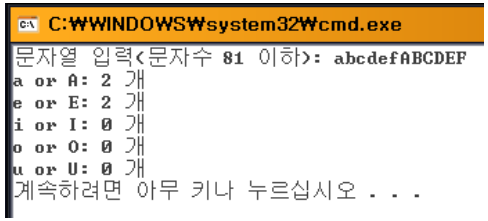


■ **challenge13b**(난이도 상) 하나의 단어를 입력 받아서 이 단어 안에 포함된 숫자의 합을 출력하는 프로그램을 작성하라. 단어 안에 숫자가 연이어 나올 경우 하나의 수로 간주한다.

실행예:

```
Enter a word: ab22c3d5
글자 안의 수의 합은 30
```

■ **challenge6c**(난이도 중) 주어진 문자열에서 모음-a(A), e(E), i(I), o(O), u(U)-의 개수를 세는 프로그램을 작성하라. 소문자 대문자를 함께 카운트한다.

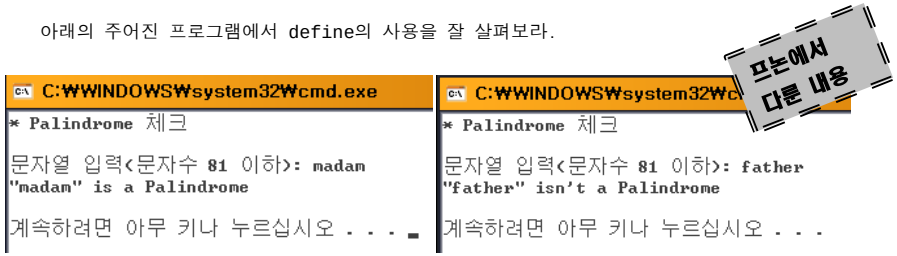


아래의 주어진 프로그램에서 define의 사용을 잘 살펴보라.

```
#include <stdio.h>

#define MAX_STRING 81

int main()
{
    char str[MAX_STRING];
    int countA = 0, countE = 0, countI = 0, countO = 0, countU = 0;
    int i = 0;

    printf("문자열 입력 (문자수 %d 이하): ", MAX_STRING);
    scanf("%s", str);

    // 여기에 코드를 넣는다




}
```

■ **challenge6d (난이도 중)**주어진 단어가 palindrome인지를 판별하는 프로그램을 작성하라. palindrome은 madam나 abccba처럼 앞에서부터 읽으나 뒤에서부터 읽으나 동일한 단어를 의미한다.

아래의 주어진 프로그램에서 define의 사용을 잘 살펴보라.



프논에서 다룬 내용

```
#define MAX_STRING 81
#define BOOL int
#define TRUE 1
#define FALSE 0

// Palindrome 검사
BOOL isPalindrome(char str[]);

int main(void)
{
	char str[MAX_STRING];

	printf("* Palindrome 체크\n\n");
	printf("문자열 입력(문자수 %d 이하): ", MAX_STRING);
	scanf("%s", str);

	if (isPalindrome(str))		// Palindrome 검사
		printf("\"%s\" is a Palindrome\n\n", str); // “를 출력하려면 \”를 써야
	else
		printf("\"%s\" isn't a Palindrome\n\n", str);

	return 0;
}


BOOL isPalindrome(char s[])
{
	//문자열 s를(사실은 s는 포인터이지만, 이는 2학기때 배운다)
	//체크해서 palindrome이면 TRUE값을 palindrome아니면 FALSE값을 return한다.









}
```

LAB 6 - 문자열(추가)

컴프1에서 다룬 내용

■ **LAB6_3 문자열의 함수 매개변수 전달**을 살펴보자.

```
void printUpperCase(char s[])
{
	int i;
	for (i = 0; s[i] != '\0'; i++)
		if (s[i] >= 'A' && s[i] <= 'Z')
			printf("%c", s[i]);
	printf("\n");
}
```

이때 일반 배열과는 달리 문자열의 경우는 그 **크기**를 같이 전달 할 필요없다.
그 끝이 '\0'으로 끝나는 성질을 이용하면 되기 때문이다.
아래의 실행결과를 갖도록 함수 strLength를 추가하여 프로그램을 완성하라.

```
C:₩WINDOWS₩system32₩cmd.exe
Enter a string:VeryGood
길이는 8
대문자만 출력하면
VG
계속하려면 아무 키나 누르십시오 . . .
```

```
#include <stdio.h>

void printUpperCase(char s[]);
int main(void)
{
	char str[81];

	printf("Enter a string:");
	scanf("%s", str);

	printf("길이는 %d\n", strLength(str)); // strLength 호출

	printf("대문자만 출력하면\n");
	printUpperCase(str); // printUpperCase 호출
}
```

HW 6 - 문자열(문자열의 함수매개변수 전달)

■ **HW6_4**

컴프1에서 다룬 내용

인생은 ATTITUDE에 달렸다! - 진대제 전 정보통신부 장관

그가 말하는 `알파벳으로 보는100점짜리 인생의 조건`은 이렇다.
`A 는1, B는2, C는3.....Z는26`같은 방식으로A에서Z까지 점수를 매긴다.
이 공식을 각 단어에 적용하면 된다.
이를테면 행운을 뜻하는 단어
`LUCK`의 경우L(12) +U(21) +C(3) +K(11)`이므로 합계는 47점이 된다.
진 장관은 이렇게 계산한 각 단어의 점수를 프리젠테이션 화면으로
보여주면서 LOVE(사랑)는 낙제점을 면한54점,
돈(MONEY)은 72점
지식(KNOWLEDGE)은96점 밖에 안 된다고 설명한다.
또 열심히 일하는 경우(HARDWORK)도 98점,운도 좋고 돈도 많다는 의미의
`FORTUNE`역시 99점에 불과 하다고 강조한다.
그러나 `자세 몸가짐`을 의미하는 `***ATTITUDE***`는 100점으로
`결국 인생은마음 먹기에 달려있는 것 같다`는게 진대제 전 장관의 강의 요지다.

*휴식,스트레스도 100점
그는 공교롭게도 정신적 중압감을 나타내는`스트레스(***STRESS***)와 휴식을
의미하는 숙어(***TAKEAREST***)역시 100점` 이라며 `인생에는 적당한
스트레스와 휴식도 필요한것 같다`고 덧 붙인다

위의 글에서 설명된대로 단어를 입력받아 그 점수를 계산하는 프로그램을 작성하라.
대문자 소문자 모두 작동하도록 하라.

힌트:
처음에는 대문자에만 작동하도록 만든후,
이를 성공하면 소문자도 다룰수 있도록 수정하여 완성한다.

```
C:₩WINDOWS₩system32₩cmd.exe
단어를 입력하세요<빈칸없이>:Love
점수는 54
계속하려면 아무 키나 누르십시오 . . .
```

```
C:₩WINDOWS₩system32₩cmd.exe
단어를 입력하세요<빈칸없이>:Attitude
점수는 100
계속하려면 아무 키나 누르십시오 . . .
```

```
#include <stdio.h>
int main(void)
{
	char str[20];

	printf("단어를 입력하세요(빈칸없이):");
	scanf("%s", str);

	printf("점수는 %d\n", calculatePoint(str));
}

// calculatePoint 함수의 정의
```

HW 6_Z : 프논 학기말 (문자열 관련)

■ 2020 프논 학기말 고사

□ 문제 5

문제(느슨한 패스워드 체킹)

프논이 교수는 apple 이라는 패스워드를 쓴다.

로그인시 패스워드를 입력할 때 자꾸 apple 뒤에 뭔가를 붙이는 이상한 습관이 생겼다.
패스워드 입력시 앞부분만 맞으면 로그인이 성공 하도록 <느슨한 패스워드 체킹 시스템>으로 바꾸려한다.

즉 apple 을 넣으면 당연히 로그인 성공이지만 apple1, 혹은 appleapple 이라고 넣어도 로그인 성공이 되게 하고 싶다.

이러한 느슨한패스워드체킹을 테스트하기 위해 패스워드를 3 번 입력하게 하고 각 패스워드에 대해서 성공하면 1 을
실패하면 0 을 출력한다.

예를 들어, 세 번 모두 맞는 패스워드이면 111 을,
처음 시도만 맞으면 100, 다 틀리면 000 을 출력한다.

[주의]

변경하지말라 등 지시사항이 없는 경우 코드를 수정해도 좋음
변수 추가 가능

실행예 1:
입력) apple apple1 applebb <- 세 개의 입력 모두 패스워드로 성공
출력) 111

실행예 2:
입력) appleee appl bple <- 첫번째 시도만 패스워드 입력 성공
출력) 100

```
#include <stdio.h>
int checking(char w0[], char w[])
{
}
int main(void)
{
        char w0[81] = "apple";
        char w1[81], w2[81], w3[81];

        scanf("%s %s %s", w1, w2, w3);

        printf("%d", checking(w0, w1));
        printf("%d", checking(w0, w2));
        printf("%d", checking(w0, w3));
}
```


□ 문제 6

문제(단어속단어여부)(난이도 상)

is_word_in_word 함수는 다음과 같은 prototype 을 갖는다.

```
int is_word_in_word(char w1[], int start, char w2[]);
```

위의 함수는 단어 w2 가 단어 w1 의 start 인덱스위치에 있으면 1 을, 아니면 0 을 반환한다.

예를 들어 w1 가 ababcd 이고 w2 가 abc 일 때,
start 가 2 이면 위의 함수는 1 을 반환하고
그 외에는 0 을 반환한다.

주어진 프로그램의 main 함수는

문자열 s1 와 s2 를 입력으로 받아서
start 를 0 부터 (w1 의 길이 - 1)까지 반복적으로 is_word_in_word 를 호출하여 그 반환값(0 또는 1)을 출력한다.

```
즉 s1 가 ababcd, s2 가 abc 라면

start 가 0 일때: 0
start 가 1 일때: 0
start 가 2 일때: 1   <- s1 에서 인덱스가 2 인 위치에 s2 가 존재하므로
start 가 3 일때: 0
start 가 4 일때: 0
start 가 5 일때: 0

그러므로 001000 을 출력한다.Copy
```

is_word_in_word 함수를 정의하여 아래와 같은 실행예가 되도록 프로그램하라.

[주의]

main 함수는 그대로두고(변경시 감점)
변경하지말라 등 지시사항이 없는 경우 코드를 수정해도 좋음
변수 추가 가능

실행예
입력) ababcd abc
출력) 001000

실행예
입력) cabbcc cc
출력) 000010

실행예
입력)abcabc abc
출력)100100

실행예
입력) abcdef xx
출력) 000000

```
#include <stdio.h>

// 문자열 w2 가 문자열 w1 의 start 위치에 있으면 1 을, 아니면 0 을 반환하는 함수
int is_word_in_word(char w1[], int start, char w2[])
```

```
{



}


int main(void) // main은 변경하지 말고 사용한다. 변경시 감점
{
        char s1[20], s2[20];
        int i, j;

        scanf("%s", s1);
        scanf("%s", s2);

        for (i = 0; s1[i] != '₩0'; i++)
                printf("%d", is_word_in_word(s1, i, s2));
}
```

■ **2018 프논 학기말 고사**

□ 스트링 접합

알파벳 순으로 나열된 두개의 스트링 a, b을 입력받아서 이를 하나로 합쳐서 c라는 새로운 스트링을 만드는 함수 mergeString을 정의하라. c도 알파벳 순으로 나열되어 있어야한다.
즉,
예1) a: "ADEFX" b: "BGHYZ" → c: "ABDEFGHXYZ"

예2) a: "ABC" b: "XYZ" → c: "ABCXYZ"

예3) a: "ABCX" b: "ACYZ" → c: "AABCCXYZ"

```
int mergeString(char c[], char a[], char b[])
{




}
int main(void)
{
        char word1[10], word2[10];
        char mergedWord[20];

        scanf("%s", word1);
        scanf("%s", word2);
        mergeString(mergedWord, word1, word2);
        printf("%s₩n", mergedWord);
}
```

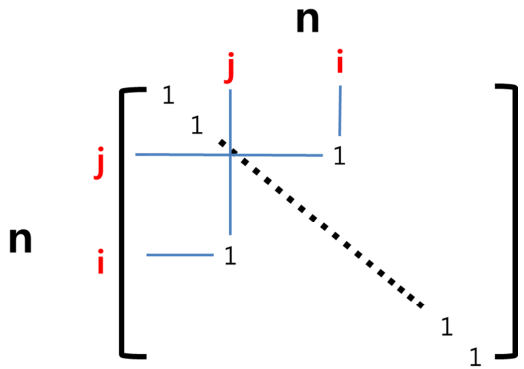
```
C:₩WINDOWS₩sy
ABC
BXYZ
ABBCXYZ
계속하려면 아무 키느
```

Project 04(sns: 2차원 배열, 행렬곱)

본 과제는 여러분이 일상에서 자주 접하는 cyworld, facebook, twitter와 같은 SNS에 대한 것이다. 이런 SNS 시스템의 주요 기능 중의 하나는 친구를 추천하고 추천된 친구가 2촌 혹은 3촌이라고 알려주는 것이다. 여기서 2촌이라는 것은 내가 지금 추천된 친구와 1촌은 아니지만 내 1촌 중에서 한 명 이상이 추천된 친구와 1촌 관계를 맺고 있음을 의미한다. 즉, 추천된 친구는 내 1촌이 아니지만 내 1촌의 1촌이다.

이번 과제는 SNS를 가정하고 그 시스템의 이름을 DDSNS라고 하자. 그리고 DDSNS 상에서 모든 가입자의 1촌의 1촌을 계산하는 프로그램을 목표로 한다. 이번 과제를 해내기 위해서는 2차원 배열과 2차원 배열을 파라미터로 함수를 호출하는 방법을 잘 숙지하고 있어야 한다. 먼저 주어지는 것은 모든 사용자와 모든 사용자의 1촌 정보이다. 이 정보를 표현하기 위해 이번 과제에서는 2차원 배열을 사용한다. 만약 현재 DDSNS의 사용자가 100명이라면 100x100 크기의 2차원 배열을 생각한다. 그리고 사용자는 영/숫자로 조합된 id 대신에 0~99 범위의 숫자를 id로 가진다고 생각하자.

그리고 모든 사용자의 1촌을 표현하기 위해 2차원 배열 link_data를 다음과 같이 0과 1로 채운다. i라는 사용자



가 j라는 사용자와 1촌이라면 link_data[i][j]과 link[j][i]를 1로 하고 1촌이 아니면 link_data[i][j]와 link_data[j][i]는 0으로 한다. 여기서 i와 j는 사용자를 나타내는 정보이므로 0~99 범위의 숫자가 된다. 이렇게 link_data를 채우면 DDSNS의 link_data는 0 혹은 1로 채워진 2차원 배열이 된다. 참고로 link_data[i][i]는 1로 한다. (즉, 나와 나는 1촌 관계라고 하자.) 따라서 link_data는 아래와 같은 모양을 하게 될 것이다.

이 행렬을 역으로 해석하자. 만약 i라는 사용자가 j 사용자와 1촌 관계이면 link_data[i][j] 및 link_data[j][i] 값은 1이 된다. 만약 i 사용자가 j 사용자와 1촌은 아니지만 2촌 관계라면 어떤 공통 1촌 k를 가진다는 것이다. 즉 link_data[i][j]는 0이지만, link_data[i][k], link_data [k][i], link_data [j][k], link_data [k][j]가 1이 된다. 이 예제에서는 link_data[i][k]와 link_data[k][j]가 중요하다.

만약 2촌까지도 위의 방식으로 표현해주는 행렬을 link_data2라고 하고 i와 j가 2촌 관계라면 link_data2[i][j]는 1이 된다. 이것을 잘 음미하면 link_data[i][k]와 link_data[k][j]가 모두 1인 k가 존재하면 link_data2[i][j]가 1이 됨을 알 수가 있다. 즉, i와 j의 공통 1촌 k가 전체 사용자 중에서 1명 이상 존재하면 된다는 것이다. 이것을 좀더 formal하게 수식으로 작성하면,

$$\text{link\_data2[i][j]} = (\text{link\_data[i][0]} \wedge \text{link\_data[0][j]}) \vee (\text{link\_data[i][1]} \wedge \text{link\_data[1][j]}) \vee \ldots \vee (\text{link\_data[i][n-1]} \wedge \text{link\_data[n-1][j]})$$

위 수식에서 ∧는 AND 연산을, ∨ 는 OR 연산을 의미한다.

위의 내용은 결국 boolean matrix의 logical multiplication과 연관이 있다. Boolean matrix는 matrix의 value가 true (1) 혹은 false (0) 인 matrix를 의미하며, boolean matrix의 논리곱 연산은 일반 matrix의 곱 연산과 유사한데 차이는 + 대신에 OR 연산을 그리고 * 대신에 AND 연산을 한 것이다. 즉, 전체 사용자의 2촌까지를 계산

해서는 link_data와 link_data를 논리곱 연산을 수행하면 된다. 아래의 수식을 좀더 주의 깊게 보자. 이제부터 +는 OR로 *는 AND로 읽자. 그리고 a를 원소로 가지는 matrix를 link_data로 생각하면 R을 원소로 가지는 matrix는 2촌까지를 표현하는 matrix가 된다.

$$\begin{bmatrix} a_{00} & a_{01} & \ldots & a_{0(n-1)} \\ a_{10} & a_{11} & \ldots & a_{1(n-1)} \\ \ldots & \ldots & \ldots & \ldots \\ a_{(n-1)0} & a_{(n-1)1} & \ldots & a_{(n-1)(n-1)} \end{bmatrix} \times \begin{bmatrix} a_{00} & a_{01} & \ldots & a_{0(n-1)} \\ a_{10} & a_{11} & \ldots & a_{1(n-1)} \\ \ldots & \ldots & \ldots & \ldots \\ a_{(n-1)0} & a_{(n-1)1} & \ldots & a_{(n-1)(n-1)} \end{bmatrix}$$

$$= \begin{bmatrix} R_{00} & R_{01} & \ldots & R_{0(n-1)} \\ R_{10} & R_{11} & \ldots & R_{1(n-1)} \\ \ldots & \ldots & \ldots & \ldots \\ \ldots & \ldots & \ldots & \ldots \\ R_{(n-1)0} & R_{(n-1)1} & \ldots & R_{(n-1)(n-1)} \end{bmatrix}$$

$$R_{00} = a_{00} \times a_{00} + a_{01} \times a_{10} + \ldots + a_{0(n-1)} \times a_{(n-1)0}$$

위의 matrix에서 R00은 아래와 같이 계산이 된다.
이것을 일반화하면 아래와 같이 구해진다.

$$R_{ij} = a_{i0} \times a_{0j} + a_{i1} \times a_{1j} + \ldots + a_{i(n-1)} \times a_{(n-1)j}$$

Rij는 a_{ik}와 a_{kj}가 모두 1인 k가 1개 이상 존재하면 Rij는 1이 된다. 즉, i와 k가 1촌이고 k와 j가 1촌을 만족시키는 k가 존재하면 i와 j가 2촌임을 알 수가 있고, link_data matrix를 두 번 곱하면 1촌의 1촌까지 표시해주는 matrix를 구할 수 있음을 알 수가 있다.

이번 과제는 이것을 활용하여 2촌까지의 인맥관계를 구하는 프로그램을 작성하는 것이다. 즉, n * n 크기의 boolean martix를 논리곱하는 프로그램을 작성하는 것이다. 이 번에도 역시 main 함수는 주어지며, 여러분들은 나머지 코드를 작성해야 한다.

□ 손으로 풀어보기**(생략 가능)**

0 과 1, 1 과 2, 2 와 4, 3 과 4 가 각각 1 촌이라고 하고

최초 1 촌 관계를 보이는 배열(Friends matrix)의 모습을 그린 후

2 촌까지의 관계를 그려보라(행렬 곱 사용)

■ **Project4_0 (일촌의 일촌 #1)**

사용자 5명을 가정하자. 각각 0부터 4까지 번호를 붙인다.

1촌 친구관계는 다음과 같다고 가정할 때,

0과 1, 1과 2, 2와 4, 3과 4

2촌까지의 인맥관계를 구하는 프로그램을 작성하라.

```
#include <stdlib.h>
#include <stdio.h>
#include <time.h>

#define NUM_OF_MEMBERS 5

// 함수 원형
void print_links(int data[][NUM_OF_MEMBERS] ); //2차원 배열 출력
void matrix_multiplication(int data[][NUM_OF_MEMBERS], int result[][NUM_OF_MEMBERS] );
                        //2차원 배열 논리곱 data X data → result

int main( void )
{
    // 0과 1, 1과 2, 2와 4, 3과 4가 각각 1촌임을 보이도록 배열을 초기화하라.
    int link_data[NUM_OF_MEMBERS][NUM_OF_MEMBERS] = {




    }

    // 2촌 관계를 넣을 배열을 0으로 초기화한다.
    int link_result[NUM_OF_MEMBERS][NUM_OF_MEMBERS] = {0};

    printf("=============================================================₩n");
    printf("Friends matrix₩n");
    printf("=============================================================₩n");
    print_links(link_data); // 최초 1촌 관계 출력

    matrix_multiplication(link_data, link_result); // 2촌 관계를 link_result에 넣는다

    printf("=============================================================₩n");
    printf("Friends of friends matrix₩n");
    printf("=============================================================₩n");

    print_links(link_result); // 2촌까지의 관계 출력
}

void print_links(int data[][NUM_OF_MEMBERS])
{
    // 정의하라
}

void matrix_multiplication(int data[][NUM_OF_MEMBERS], int result[][NUM_OF_MEMBERS])
{
    // 정의하라
}
```

■ Project4_1(일촌의 일촌 #2)

sns 가입자 15명에 대해서 random하게 1촌을 설정해서(최대 4명으로) 1촌, 2촌의 인맥관계를 보여주는 프로그램을 작성하라. print_links와 matrix_multiplication는 앞 문제에서 정의한 것을 그대로 쓰면 되고 별도로 더 코드를 추가하지 않는다.

main함수를 이해하고 이전에 정의한 함수의 정의부분을 채워넣고 프로그램을 실행시켜본다.

```
#include <stdlib.h>
#include <stdio.h>
#include <time.h>

#define NUM_OF_MEMBERS 15 // sns 가입자의 수

// 함수 원형
void print_links(int data[][NUM_OF_MEMBERS] );
void matrix_multiplication(int data[][NUM_OF_MEMBERS], int result[][NUM_OF_MEMBERS] );

int main( void )
{
	int link_data[NUM_OF_MEMBERS][NUM_OF_MEMBERS] = {0}; //최초 link matrix 1촌의 matrix
	int link_result[NUM_OF_MEMBERS][NUM_OF_MEMBERS] = {0}; //2촌까지의 관계가 표시된 matrix

	int i = 0;
	int j = 0;
	int num_of_steps=0;
	int ALL_ONES=0;

	srand(100); // 같은 결과를 얻기위해서 일단 씨드를 100으로 설정(나중에는 주석처리해서 실행)
	// srand( (unsigned int)time(NULL) );

	// link_data 값 넣기: 자신은 모두 1촌이다
	for(i = 0; i<NUM_OF_MEMBERS; i++ )
	{
		link_data[i][i] = 1; //i와 i의 관계는 1촌. 즉 1로 표시.
	}

	// link_data 값 넣기: random하게 수를 발생시켜서 1촌 친구를 설정한다.
	for(i = 0; i<NUM_OF_MEMBERS; i++ ) //각 user마다 대략 4명의 친구가 있다고 가정.
	{
		j=0;
		while ( j<2 ) //i가 두 개의 링크를 연결하고 i가 아닌 다른 user가 i와 연결
		{
			int new_link = rand()%NUM_OF_MEMBERS;
			if( new_link != i )
			{
				link_data[i][new_link] = 1; //i와 new_link가 1촌이면
				link_data[new_link][i] = 1; //new_linke와 i도 1촌.
				j++;
			}
		}
	}
	printf("=================================================================₩n");
	printf("Friends matrix₩n");
	printf("=================================================================₩n");
	print_links(link_data); //최초 1촌 관계 출력

	matrix_multiplication(link_data, link_result);

	printf("=================================================================₩n");
	printf("Friends of friends matrix₩n");
	printf("=================================================================₩n");
	print_links(link_result); // 계산된 2촌까지의 관계 출력
}
```

```
void print_links(int data[][NUM_OF_MEMBERS]) /
{
	//앞의 프로젝트에서 정의

void matrix_multiplication(int data[][NUM_OF_MEMBERS], int result[][NUM_OF_MEMBERS])
{
	//앞의 프로젝트에서 정의
}
```

이 프로그램의 결과는 다음과 같다. srand함수 사용시 seed를 100으로 고정하였으므로 항상 같은 결과가 나온다.

아래 결과에서 Friend matrix의 A[0][14](혹은 A[14][0])가 0으로 표시되어있으므로 사용자 0과 14는 1촌이 아니다.

하지만, A[0][5]과 A[5][14]이 1이기 때문에 즉 사용자 0과 5이 1촌이고 5과 14가 1촌이기 때문에 결과적으로 0과 14는 2촌 관계가 있는 것이다. 이는 Friends of friends matrix의 A[0][14]가 1로 표현된다.

```
=================================================================
Friends matrix
=================================================================
  1   1   0   0   1   1   0   0   0   1   0   0   0   0   0
  1   1   0   0   0   1   0   0   0   1   1   0   1   0   0   0
  0   0   1   0   1   0   0   0   0   0   1   0   0   0   0   0
  0   0   0   1   0   0   1   1   1   0   0   0   0   1   0
  1   0   1   0   1   0   0   0   0   0   0   1   0   0   0   0
  1   0   0   0   0   1   0   0   0   0   0   1   0   0   1   1
  0   0   0   1   0   0   1   0   1   1   0   0   0   0   0   0
  0   0   0   1   0   0   0   0   1   0   0   0   1   0   0   1
  0   1   0   1   0   0   1   0   1   0   0   0   0   1   0   0
  1   1   1   0   0   0   1   0   0   1   0   1   0   0   0   0
  0   0   0   0   1   1   0   0   0   0   0   1   0   1   0   1
  0   1   0   0   0   0   0   1   0   1   0   1   1   0   0   0
  0   0   0   0   0   0   0   0   1   0   1   1   1   1   0   0
  0   0   0   1   0   1   0   0   0   0   0   0   1   1   1   1
  0   0   0   0   0   1   0   1   0   0   1   0   0   1   1   1
=================================================================
Friends of friends matrix
=================================================================
  1   1   1   0   1   1   1   0   1   1   1   1   0   1   1   1
  1   1   1   1   1   1   1   1   1   1   0   1   1   0   0   0
  1   1   1   0   1   0   1   0   0   1   1   1   0   0   0   0
  0   1   0   1   0   1   1   1   1   1   0   1   1   1   1   1
  1   1   1   0   1   1   0   0   0   1   1   0   1   0   1   1
  1   1   0   1   1   1   0   1   0   1   1   0   1   1   1   1
  1   1   1   1   0   0   1   1   1   1   0   1   1   1   0   0
  0   1   0   1   0   1   1   1   1   1   1   1   1   1   1   1
  1   1   0   1   0   0   1   1   1   1   1   1   1   1   1   0
  1   1   1   1   1   1   1   1   1   1   0   1   1   0   0   0
  1   0   1   0   1   1   0   1   1   0   1   1   1   1   1   1
  1   1   1   1   0   0   1   1   1   1   1   1   1   1   1   1
  0   1   0   1   1   1   1   1   1   1   1   1   1   1   1   1
  1   0   0   1   0   1   1   1   1   0   1   1   1   1   1   1
  1   0   0   1   1   1   0   1   0   0   1   1   1   1   1   1
계속하려면 아무 키나 누르십시오 . . .
```