자료구조 프로그래밍 과제 1

제출일: 2022-09-20
학번/이름: 2016121150 / 윤준영

문제 1. 문자열을 받아서 소문자를 모두 대문자로 바꾸어 출력하는 프로그램을 작성하시오.

PHW1 > C hw1_2016121150_1.c > ...

```c
// HW1 문제 1.
// 문자열을 받아서 소문자를 모두 대문자로
// 바꾸어 출력하는 프로그램을 작성하시오.
// 2016121150 윤준영

#include <stdio.h>

char upper(char i){
    // make upper letter a~z to A~Z
    // if i is 97~122 then -32
    if (i>=97 && i<=122) i = i-32;
    return i;
}

int main(){
    char s[100];
    int n = 0;
    printf("Input a string: ");
    scanf("%s", s);
    // make upper letter until end of string('\0')
    while (s[n]!='\0'){
        s[n] = upper(s[n]);
        n++;
    }
    printf("->%s\n", s);
    return 0;
}
```

upper 함수: 입력된 문자가 소문자일 경우 대문자로 바꿔주는 upper 함수를 작성하였다. upper 함수는 입력된 문자의 ASCII 코드를 통하여 소문자에 해당하는 97~122의 값을 가지면 해당 문자의 ASCII 코드 값을 32 빼줌으로써 대문자로 반환하는 함수이다.

main 함수: main 함수는 먼저 넉넉한 char 배열을 생성하고, 문자열을 입력받은 뒤, n=0부터 1씩 증가시켜 가면서 s[n]에 대해 upper함수를 적용한다. 만약 다음 문자 s[n]이 NULL('₩0')가 되면 반복을 종료하고, 문자열을 출력한다.

프로그램 실행 결과는 다음과 같다.

```
Input a string: adfa223x3299aasXX123
-> ADFA223X3299AASXX123
* 터미널이 작업에서 다시 사용됩니다. 닫으려면 아무 키나 누르세요.
```

숫자와 대문자는 upper함수에서 별다른 작업을 하지 않고 그대로 반환하기 때문에 입력된 문자열 그대로 출력되고, 소문자의 경우에는 ASCII 코드 값의 32를 빼준 대응하는 대문자로 출력된 것을 확인할 수 있다.

문제 2. 배열 세 개를 입력받아 처음 두 개의 문자열을 연결한 것을 세 번째 문자열에 저장하는 함수 concat를 작성하여 다음 프로그램을 완성하시오.

PHW1 > C hw1_2016121150_2.c > ...

```c
// HW1 문제 2.
// 배열 세개를 입력받아 처음 두 개의 문자열을
// 연결한 것을 세번째 문자열에 저장하는 함수
// concat 를 작성하여 다음 프로그램을 완성하시오.
// 2016121150 윤준영

#include <stdio.h>

char *concat(char c1[], char c2[]){
    int n=0, m=0;
    //count length of c1
    while (c1[n]!='\0') n++;
    //put c2 on the end of c1
    while (c2[m]!='\0'){
        c1[n+m] = c2[m];
        m++;
    }
    //declare end of string
    c1[n+m] = '\0';
    return c1;
}

main(){
    char s1[100];
    char s2[100];
    char *s3;
    scanf("%s", s1);
    scanf("%s", s2);
    s3 = concat(s1, s2);
    printf("-->%s\n", s3);
}
```

concat 함수: concat 함수는 두 문자열을 입력받고, 합쳐진 문자열을 반환하는 함수이다. 두 문자열을 입력받아 첫 문자열 뒤에 두 번째 문자열을 추가하는 방식으로 구현하였다. 첫 문자열 c1의 원소 c1[n]에서 n을 0에서 1씩 증가시켜 NULL('₩0')값이 나올 때까지의 n을 구하여 c1의 길이를 확인하고, c1의 뒤에 c2를 붙여넣었다. c2[m]에서 NULL('₩0')값이 나올 때까지 m을 0부터 1씩 증가시켜 얻은 문자를 c1[n+m]에 복사하여 두 문자열을 합쳤다. 그 후 c1의 마지막에 NULL('₩0')을 추가함으로써 문자열의 끝임을 알려주었다.

마지막에 NULL을 추가하지 않을 경우: 코드의 18줄을 주석처리하고 함수를 실행한 결과 다음과 같이 출력되는 것을 확인할 수 있었다.

```
abfdf
ffff
-->abfdfffff?崎
* 터미널이 작업에서 다시 사용됩니다. 닫으려면 아무 키나 누르세요.
```

이는 문자열 출력에서 문자열의 알 수 없기 때문에 문자열이 끝나도, 뒤에 이미 저장되어 있던 문자열이 그대로 출력된다는 것을 짐작할 수 있었다.

최종적으로 **프로그램 실행 결과**는 다음과 같다.

```
abfdf
ffff
-->abfdfffff
```
터미널이 작업에서 다시 사용됩니다. 닫으려면 아무 키나 누르세요.

마지막에 NULL을 추가하면 문자열의 끝을 인식해서 제대로 결과가 나오는 것을 확인할 수 있었다.

문제 3. 2x2 행렬 두 개를 받아서 두 행렬의 곱을 출력하는 프로그램을 작성하시오.

PHW1 > C hw1_2016121150_3.c > ...

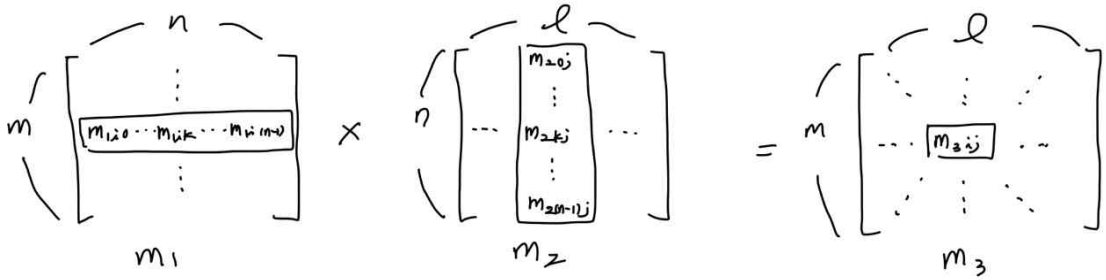
```c
// HW1 문제 3.
// 2x2 행렬 두 개를 받아서 두 행렬의 곱을
// 다음과 같이 출력하는 프로그램을 작성하시오.
// 2016121150 윤준영

#include <stdio.h>

void matgen(int a[], int m, int l) {
    for (int i=0; i<m; i++){
        for (int j=0; j<l; j++)
            scanf("%d", &(a[i*l+j]));
    }
}

void matprint(int a[], int m, int l) {
    for (int i=0; i<m; i++){
        for (int j=0; j<l; j++)
            printf("%d ", a[i*l+j]);
        printf("\n");
    }
}

void matmul(int m1[], int m2[], int m3[], int m, int n, int l)
{
    int i, j, k;
    for (i=0; i<m; i++)
        for (j=0; j<l; j++) {
            m3[i*l+j] = 0;
            for (k=0; k<n; k++)
            m3[i*l+j] += m1[i*n+k]*m2[k*l+j];
        }
}

int main(){
    int a[2][2];
    int b[2][2];
    int c[2][2];
    printf("Input elements of the first 2x2 matrix: ");
    matgen(&(a[0][0]), 2, 2);
    printf("Input elements of the second 2x2 matrix: ");
    matgen(&(b[0][0]), 2, 2);
    matmul(&(a[0][0]),&(b[0][0]),&(c[0][0]),2,2,2);
    matprint(&(c[0][0]), 2, 2);
    return 0;
}
```

행렬을 생성하는 matgen, 행렬을 출력해주는 matprint, 행렬곱을 해주는 matmul 함수를 작성하여 사용하였다.

matgen 함수: matgen 함수는 행렬의 첫 원소의 주소, 행렬의 크기 m, l을 입력받아 행렬을 생성하는 함수이다. 행을 의미하는 i가 0에서 m-1까지 증가하는 동안 각각 행의 열에 해당하는 j는 0에서 l까지 증가하여 scanf 함수를 이용하여 행렬의 원소를 입력받아 행렬을 생성한다.

matprint 함수: matprint함수는 matgen함수와 마찬가지로 행렬의 첫 원소의 주소, 행렬의 크기 m, l을 입력받아, i와 j가 각각 0에서 m, l까지 증가하며 행렬의 원소를 출력한다. 한 행 출력이 끝나면 '₩n'을 출력하여 줄바꿈을 수행한다.

matmul 함수: matmul함수는 세 행렬의 첫 원소의 주소를 받고, 행렬의 크기 m, n, l를 입력받는다. 첫 행렬은 m x n, 두 번째 행렬은 n x l, 세 번째 행렬은 m x l의 크기를 가진다. 행렬곱의 결과는 다음과 같이 나타낼 수 있다.



$$m_1[i][0]\times m_2[0][j]+m_1[i][1]\times m_2[1][j]+\ \cdots\ +m_1[i][k]\times m_2[k][j]+\ \cdots\ +m_1[i][n]\times m_2[n][j]=m_3[i][j]$$

세 번째 행렬이 행렬곱의 결과가 되는 함수이므로, for 문을 이용하여 m행, l열의 원소를 계산한다. 한 원소는 n회의 곱셈이 이루어져 합쳐지게 된다. 따라서 각 원소마다 m3[i*l+j]=0으로 초기화해 준 후 for 문을 이용하여 k=0에서 n-1까지 곱한 후 누적시킨다.

main 함수: 2x2 행렬인 a, b, c를 선언하고, a[0][0], b[0][0]의 주소와 matgen 함수를 이용하여 행렬 a와 b를 생성한다. 그 후 matmul 함수를 이용하여 행렬 a와 b의 행렬곱을 c로 저장한 후, matprint를 이용하여 행렬 c를 출력한다.

프로그램 실행 결과는 다음과 같다.

```
Input elements of the first 2x2 matrix: 1 3 2 4
Input elements of the second 2x2 matrix: 1 2 2 1
7 5
10 8
* 터미널이 작업에서 다시 사용됩니다. 달으려면 아무 키나 누르세요.
```