Advanced C Programming – Homework #2

Prof. Cheolsoo Park

Assistant: Heesang Eom

컴퓨터정보공학부 2019202050 이강현

Submit date : 2019 / 12 / 09

문제 1

1. 과제 소개: 사용자에게 영어문장 혹은 모스부호를 입력받아 인코딩,디코딩 코드를 작성하는
2. 설계 내용: 문자들을 하나씩 입력받고 이를 알파벳인지 모스부호인지 먼저 구분한 후 조건문을 통해 문자들을 구분하고 바꿀 언어와 비교하여 출력할 것이다.

<의사 코드>

1. str 변수에 메모리를 malloc함수를 이용해 할당한다.

2.getchar함수와 반복문을 이용하여 문자단위로 입력받는다.

3.길이가 길어지면 메모리를 넘지 않도록 realloc함수를 이용해 추가로 메모리를 할당해준다.

4.입력이 모두 끝나면 알파벳을 입력받았는지 모스부호를 입력받았는지 조건문을 통해 구분한다.

5.알파벳이라면 공백은 그대로 공백으로 출력하고 알파벳이 저장된 배열과 아스키코드값을 비교연산하여 해당 알파벳을 출력해준다.

6. 문자열의 끝에 도달했을 때 반복문을 탈출한다.

7.모스부호를 입력받았다면 모스부호는 여러문자의 조합이 하나의 알파벳이므로 배열을 선언하고 공백의 유무로 알파벳을 구분한다.

8. 배열과 모스부호가 저장된 배열과 문자열비교를 후 알파벳을 참조하여 출력한다.

9.문자열의 끝에 도달했을 때 반복문을 탈출한다.

1. 코드 및 주석:

#include <stdio.h>//printf함수를 쓰기위한 헤더파일

#include <stdlib.h>//동적할당에 필요한 malloc함수와 realloc함수를 위한 헤더파일

#include <string.h>//문자열 비교를 위한 strcmp함수에 필요한 헤더파일

int main()//main함수 선언

{

int size = 1, len = 0;//동적할당에 필요한 size변수와 입력받는 str의 값들을 인덱싱하기 위한 len 변수

char \*str = (char \*)malloc(sizeof(char) \* size);//str변수에 size만큼 메모리를 할당

char Word;//str에 있는 값들을 복사하여 출력할 때 필요한 변수

char Alpabet[26] = {'A','B','C','D','E','F','G','H','I','J','K','L','M','N','O','P','Q','R','S','T','U','V','W','X','Y','Z' };//알파벳을 배열에 저장

char Morsecode[26][6] = {{".-"},{"-..."},{"-.-."},{"-.."},{"."},{"..-."},{"--."},{"...."},{".."},{".---"},{"-.-"},{".-.."},{"--"},{"-."},{"---"},{".--."},{"--.-"},{".-."},{"..."},{"-"},{"..-"},{"...-"},{".--"},{"-..-"},{ "-.--"},{"--.."}};//모스부호를 배열에 저장

int temp = 0, i = 0, j = 0, k = 0;//반복문에서 필요한 변수들

printf("In : ");//입력받는 것을 나타내기 위한 출력문

while (1)//글자의 길이를 모르기 때문에 무한 반복문을 사용

{

str[len] = getchar();//문자단위로 콘솔에서 입력받아 str에 저장

if (str[len] == '\n')//엔터를 입력받을 시

{

str[len] = NULL;//len번째 str의 값은 NULL값

break;//엔터를 입력받았기에 문장의 끝 따라서 반복문 탈출

}

if (size == len + 1) //입력받은 문자의 개수가 size와 같을 때 문자를 더 받기위해 메모리를 늘려줘야 함.

{

size += 3;//길이를 적당히 늘림

str = (char \*)realloc(str, sizeof(char)\*size);//size변수의 값 증가로 str이 좀더 메모리를 할당받음

}

len++;//다음 문자를 저장하기 위해 len값을 1증가

}

printf("Out : ");//출력되는 값을 나타내기 위한 출력문

if (97 <= str[0] && 122 >= str[0])//영어 소문자로 입력받기 때문에 해당 아스키코드로 입력 범위 지정

{

while (1)//무한 반복문

{

Word = str[j++];//Word 변수에 str값들을 하나씩 참조하기 위해 후위증가 연산

if (Word == ' ') //str에 의해 저장된 Word값이 공백이라면

{

printf(" ");//공백을 출력

}

else//공백이 아니라면

{

for (i = 0; i < 26; i++)//Word의 하나의 문자와 26개의 알파벳을 모두 비교하기 위해 반복

{

if (Word == Alpabet[i] + 32)//알파벳을 저장한 Alpabet배열은 모두 대문자이므로 32를 더해주어 소문자와 비교

{

printf("%s ", Morsecode[i]);//비교 후 일치하는 문자라면 해당 Morsecode를 출력

break;//반복문 탈출

}

}

}

if (j == len)//문자를 입력받을 때 증가된 len변수의 값과 Word변수에 str값들을 하나씩 참조하기 위해 사용했던 j변수가 같아지면 입력받은 문자 모두를 참고했다라는 뜻

break;//따라서 반복문 탈출

}

}

else//영어 소문자로 입력받지 않았다면 모스부호로 입력받았을 것

{

while (1)//무한 반복문

{

char arr[5] = { NULL };

/\*알파벳은 하나씩 입력받지만 모스부호는 여러개의 문자가

하나의 단어를 이루기 때문에 모스부호 하나를 참조하기 위해

Word가 아닌 arr배열을 이용. 길이가 5인 이유는 모스부호가 최대

문자4개조합을 넘지않고 널문자까지 포함시키기 위함\*/

for (i = 0; i < 5; i++)//길이가 5인 배열 모두 참고하기 위한 반복문

{

arr[i] = str[k++];//str변수속 문자들을 하나씩 배열에 저장

if (arr[i] == ' ')//공백일 때

{

if (i == 0)//첫문자라면

{

printf(" ");//공백 출력

}

else//첫 문자가 아니라면

arr[i] = NULL;//i번째 배열의 값은 NULL. 이는 모스부호를 입력할 때 영어단어의 구분을 공백 1개로 하기 때문이다.

break;//반복문 탈출

}

}

for (i = 0; i < 26; i++)//모스부호 26개들을 모두 비교하기 위한 반복문

{

if (!strcmp(arr,Morsecode[i]))//문자열들 간의 비교이므로 ==연산자가 아닌 문자열 비교 함수인 strcmp를 이용함

{

printf("%c", Alpabet[i]);//두 문자열이 동일할때 해당 모스부호를 알파벳으로 출력

break;//반복문 탈출

}

}

temp++;//모스부호 하나가 알파벳으로 출력되었을때 변수 1증가 즉, temp의 개수가 알파벳 개수

if (temp == len) //끝까지 참고했을 때

break;//반복문 탈출

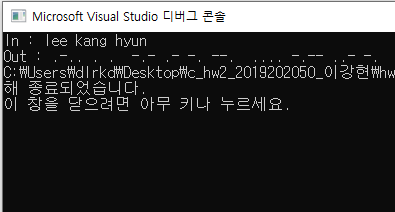
}

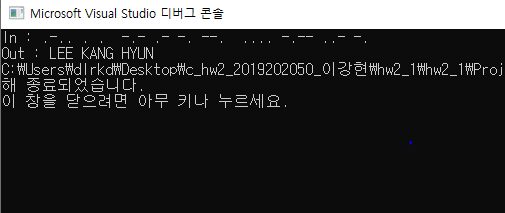
}

return 0;//종료

}

4. 실행 결과:





5. 고찰: 문자 단위로 입력을 받고 사용자가 길이를 마음대로 입력해도 받아낼 수 있었던 것은 동적할당 덕분이었던 것 같다. 배열의 길이를 [999]로 선언하고 받는 것보다 훨씬 깔끔하고 메모리의 낭비 없이 코드를 구현할 수 있었다.