<Assignment #3>

201420907

소프트웨어학과

안우일

Ⅰ. 문제 파악 및 조건 분석

(1) 문제 파악

> 1. 문자열에 들어 있는 회문(palindrome)의 개수를 확인하는 프로그램을 작성하시오.
>
> - 회문 (palindrome) 정의: madam과 같이 앞으로 읽으나 뒤로 읽으나 똑같은 단어를 뜻함

- 회문은 위에서 적혀있듯이 앞으로 읽을 때와 역으로 읽을 때가 같은 말이 되는 문자열이다.
- 요번 과제에서 작성해야 할 프로그램은 회문의 개수를 출력해주어야 하는 프로그램이다.

(ex) 'MADAM' 입력 시 회문은 'ADA' , 'MADAM' 로서 2개이므로 화면에 '2' 출력

'HAAH' 입력 시 회문은 'AA' , 'HAAH'로서 2개이므로 화면에 '2' 출력

'AAA' 입력 시 회문은 'AA', 'AA' , 'AAA'로서 3개이므로 화면에 '3' 출력

(2) 문제 조건

> - 조건:
> (1) 대소문자 구별 할 것, (2) 입력 문자열 최대 개수 255개,

- 자료형 중 char형은 문자를 ASCII코드값으로 저장하기 때문에 사용자로부터 값을 입력 받을 때 대소문자를 구별 할 수 있게 된다. 따라서 조건 (1)을 만족시키기 위해서는 char형으로 값을 입력 받아야 할 것이다.

- 입력 문자열 최대 개수가 정해진 것으로 보아 문자열을 하나의 입력으로 받기보다는 array(배열)을 통해 array의 길이를 255로 설정한 다음, 사용자로부터 값을 입력 받아야 할 것이다.

Ⅱ. 문제를 푸는 데 사용할 알고리즘(Manacher's Algorithm)

(1) 알고리즘의 정의(Manacher's Algorithm)

배열 S = 입력 받은 문자열(char형 배열로 선언하여 받아줌)

배열 A = 문자열의 i번째를 중심으로 한 회문의 최대 반지름의 길이(0<i<strlen(S)) (int형 배열로 선언)

(i=0일 때, A[i]는 당연히 0의 값을 가지므로, R의 값과 p의 값은 모두 0부터 시작한다.)

1. i는 0부터 $n-1 (n=|S|)$까지 진행된다
2. $j<i$인 모든 j에 대해 $R=\max(j+A[j])$이라하고, 또한 그러한 j를 p라 하자. 즉, $R=p+A[p]$
3. $i \leq R$인지 여부에 따라 $A[i]$의 초기값이 정해진다
   - $i>R$이라면, $A[i]$의 초기값은 0이다.
   - $i \leq R$이라면, i는 p를 중심으로 한 팔린드롬에 속한다는 이야기이다. 이때 p를 중심으로 i의 대칭점 i'을 구한다. (즉, $i'=2*p-i$) $A[i]$의 초기값은 $\min(R-i, A[i'])$으로 둔다.
4. $A[i]$의 초기값에서부터, $S[i-A[i]]$과 $S[i+A[i]]$가 같을 때까지 $A[i]$를 증가시키고, 그 다음 i로 넘어간다.

- (2)에 대한 부가설명-

R이라는 지점은 i-1까지 회문을 Search했을 때, Search된 회문들의 마지막 점들 중 가장 문자열의 끝점에 있는 지점이다.

이때의 중심점은 p라고 정의해 놓았다.

- (3)에 대한 부가설명-

위에 있는 R의 정의에 따라 만약 i라는 점이 R의 값보다 작다면, 중심점 p, 반지름 R으로 하는 회문 안에 속해있다는 말이다. 이 때, i와 같은 문자를 가진 i`이라는 대칭점이 있을 것이다. 그 점의 값은 (2*p-i)가 된다.

- (4)에 대한 부가설명-

S[i-A[i]]와 S[i+A[i]]는 각각 문자열의 i-A[i]번째와 i+A[i]번째에 있는 char이다.

여기서, i는 대칭점을 의미하고, A[i]는 대칭점에서의 거리(회문의 반지름)에 해당되므로

S[i-A[i]]와 S[i+A[i]]가 달라지기 전까지 계속해서 초기 A[i]값으로부터 A[i]값을 증가시키면

회문의 반지름을 구할 수 있다.

(2) 알고리즘을 통해 구해지는 회문(Palindrome)의 개수

-회문(Palindrome)의 개수는 반지름 배열 A[]의 요소들을 모두 더한 값이다.

(ex) 문자열 배열 S = 'MADAM' 일 때,

M	A	D	A	M

반지름 배열 A는

0	0	2	0	0

따라서 문자열 'MADAM'의 회문(Palindrome)의 개수는 (0+0+2+0+0)=2이다.

(3) Manacher's Algorithm의 한계 및 해결방안(예외사항)

① 한계

Manacher's Algorithm은 i번째 문자열을 중심으로하여 반지름이 같은 회문만을 찾아 낼 수 있다. 즉, 반지름을 R이라고 하면, 이 알고리즘을 통해 찾아낼 수 있는 회문의 길이는 (2R+1) 이므로 홀수의 길이를 가지는 회문밖에 찾아 낼 수 없고, 짝수의 길이를 가지는 회문은 찾아낼 수 없다. 따라서 짝수의 길이를 가지는 회문을 찾아내기 위해서는 어떤 해결방안이 필요하다.

② 해결방안

문자열의 사이사이에 '$' , '#'와 같은 의미 없는 문자를 추가하여 알고리즘을 통해 회문의 개수를 구하면 된다. 예를 들어 문자열 S = 'MADAM'을 문자열 S`='#M#A#D#A#M#'와 같이 재설정하여 배열A`('#'을 추가한 문자열의 가장 긴 회문의 반지름 값)을 구해주면 된다. 여기서 유의해야 할 점이 두 가지 있다. 우선, 원래 배열의 길이를 n이라고 하면, (n+1)개의 '#'을 넣어주어야 한다. 따라서 원래 문자열의 2배인 배열을 선언해주고 여기에 값을 넣어주어야 한다. 두 번째로 A`의 index에 저장되어 있는 변수의 값들은 '#'을 추가한 문자열의 변수들을 다 더한 값이므로 '#'에 의한 회문들의 개수가 더해져 있다. 따라서 원래의 문자열의 회문의 개수를 구해주기 위해서는 A`의 요소의 값들을 각각 2로 나눈 몫의 값들을 더해주어야 한다.

(ex) 문자열 배열 S`= 'MADAM' 일 때,

#	M	#	A	#	D	#	A	#	M	#

반지름 배열 A`는

0	1	0	1	0	5	0	1	0	1	0

반지름 배열 A`의 값들을 2로 몫으로 나눈 값은

0	0	0	0	0	2	0	0	0	0	0

이 되어서 이 알고리즘에 의한 문자열 'MADAM'의 회문의 개수는 (0*10+2*1)=2인데, 실제로도 'ADA','MADAM'으로 2개이므로 '#'을 넣지 않고 알고리즘을 돌려 구한 회문의 개수와 같음을 알 수 있다.

(ex) 문자열 배열 S`= 'HAAH' 일 때,

#	H	#	A	#	A	#	H	#

반지름 배열 A`는

0	1	0	1	5	1	0	1	0

반지름 배열 A`의 값들을 2로 몫으로 나눈 값은

0	0	0	0	2	0	0	0	0

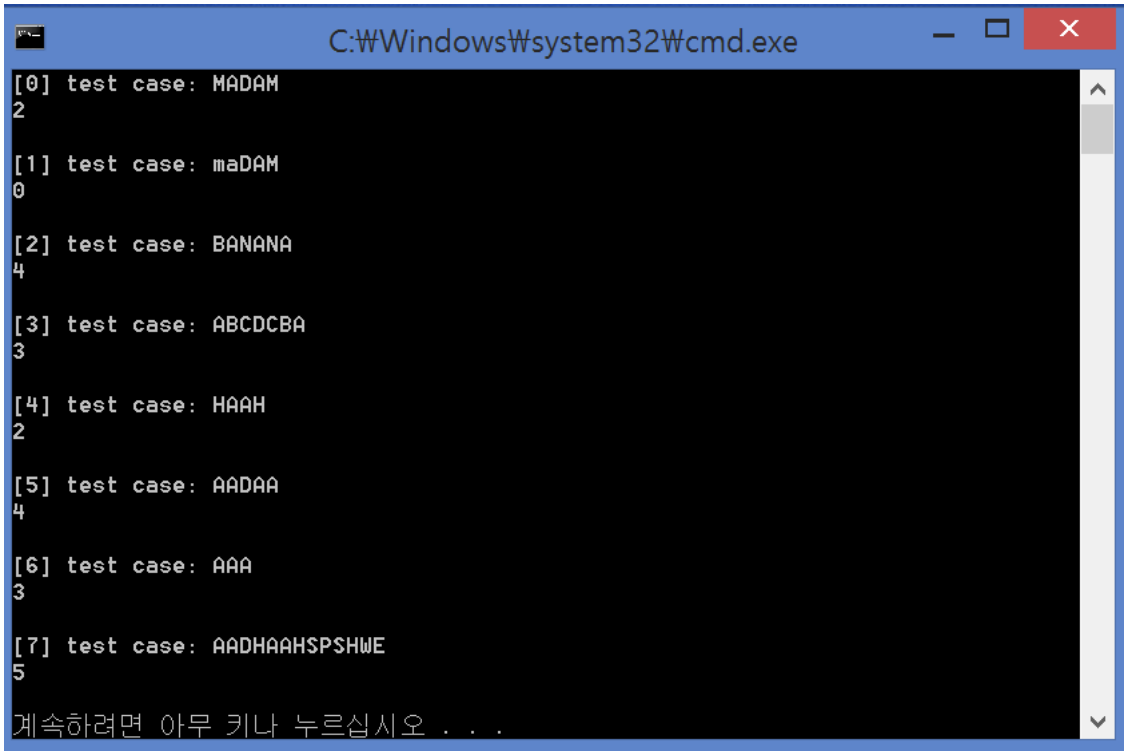
이 되어서 이 알고리즘에 의한 문자열 'HAAH'의 회문의 개수는 (0*8+2*1)=2인데, 실제로도 'AA','HAAH'으로 2개이므로 짝수개의 문자열을 가지는 회문의 개수도 구해진다는 것을 알 수 있다.

Ⅲ. 알고리즘을 C코드로 구현하기

[코드]

```
int checkPalindrome(char* buff)
{
    int ret = 0;
    // write code here
    char twice_buff[611] = { 0 }; //길이가 (2n+1)인 배열 선언
    int rad[610] = { 0 }; //반지름의 길이
    int maxR = 0, p = 0;
    int n = strlen(buff); //문자열의 길이

    //짝수의 회문도 구하기 위해 사이사이에 '#' 추가함
    for (int i = 0; i <= strlen(buff); i++)
    {
        twice_buff[2 * i] = '#';
        twice_buff[2 * i + 1] = buff[i];
    }

    //'#'을 추가한 문자열의 길이
    int N = strlen(twice_buff);

    //rad[i] 배열의 값을 각각 구하는 반복문
    for (int i = 1; i <= N; i++)
    {
        if (i <= maxR)
            rad[i] = min(rad[2 * p - i], maxR - i);
        else
            rad[i] = 0;

        while (i - rad[i] - 1 >= 0 && i + rad[i] + 1 <= N && twice_buff[i - rad[i] - 1] == twice_buff[i + rad[i] + 1])
            rad[i]++;

        if (maxR < i + rad[i])
            maxR = i + rad[i], p = i;
    }

    //실제 문자열의 rad값을 구해주기 위해 2로 나눠줌
    for (int i = 0; i <= N; i++)
    {
        rad[i] /= 2;
    }

    //0-(N-1)의 rad[i]값을 모두 더해줌
    for (int k = 0; k < N; k++)
        ret += rad[k];

    printf("%d₩n₩n", ret);
    return ret;
}

int main(void)
{
    char buff[255];
    int nTest;

    FILE *fp = freopen("input.txt", "rt", stdin);
    if (fp == NULL) {
        fprintf(stderr, "File not found₩n");
        return 0;
    }

    scanf("%d", &nTest);

    for (int i = 0; i < nTest; i++) {
        scanf("%s", buff);
        printf("[%d] test case: %s₩n", i, buff);
        checkPalindrome(buff);
    }

    fclose(fp);

    return 1;
}
```

[출력]



문자열이 'MADAM' 일 때 »» 'ADA' , 'MADAM'이므로 '2' 출력

문자열이 'maDAM' 일 때 »» 소문자와 대문자는 다른 문자임. 따라서 회문은 없으므로 '0' 출력

문자열이 'BANANA' 일 때 »» 'ANA' , 'NAN', 'ANANA', 'ANA'이므로 '4' 출력

문자열이 'ABCDCBA' 일 때 »» 'CDC' , 'BCDCB', 'ABCDCBA' 이므로 '3' 출력

문자열이 'HAAH' 일 때 »» 'AA' , 'HAAH'이므로 '2' 출력

문자열이 'AADAA' 일 때 »» 'AA' , 'ADA' , 'AADAA' ,' AA'이므로 '4' 출력

문자열이 'AAA' 일 때 »» 'AA' , 'AAA', 'AA'이므로 '3' 출력

문자열이 'AADHAAHSPSHWE' 일 때 »» 'AA' , 'AA', 'HAAH', 'SPS', 'HSPSH'이므로 '5' 출력

-출처-

2p 알고리즘 : https://algospot.com/wiki/read/Manacher's_algorithm