자료구조\_과제3\_02

String Matching

본 과제는 C++을 기반으로 작성하였습니다.



목 차

1. Naive Algorithm

2. KMP Algorithm

자료구조

김성열교수님

소프트웨어학과

201811182

박원준

2019.04.04

**1. Naive Algorithm**

1) 코드

#include <vector>

#include <string>

#include <iostream>

using namespace std;

vector<int> answer;

int main(void)

{

string T, P; // T : text 배열, P : pattern 배열

cin >> T >> P;

for (int i = 0; i < T.length(); i++)

{

int idx = i;

for (int j = 0; j < P.length();)

{

if (T[idx] == P[j])

{

idx++; j++;

if (j == P.length())

{

answer.push\_back(i);

break;

}

continue;

}

else break;

}

}

int count = 0;

while (!answer.empty())

{

int idx = answer.back();

answer.pop\_back();

for (int i = 0; i < P.length(); i++)

cout << T[idx++];

cout << endl;

count++;

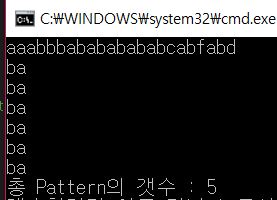
}

cout << "총 Pattern의 갯수 : " << count << endl;

return 0;

}

2) 실행 결과



**2. KMP Algorithm**

1) 코드

// 본 문제는 Pattern의 최대 길이를 100이라 가정함.(cache[100])

#include <vector>

#include <string>

#include <iostream>

using namespace std;

string T, P; // T : Text, P : Pattern

int cache[100];

vector<int> answer;

void FailureFunction()

{

for (int i = 1; i < P.length(); i++)

{

int j = cache[i - 1];

while (P[j + 1] != P[i] && j >= 0) j = cache[j];

// P[j+1] == P[i] 인 경우이면, 연속해서 패턴이 같다는 것이기 때문에, cache[i-1]보다 1 큰 값만 저장하면 된다.

// P[j+1] != P[i]인 경우이면, j의 다른 곳을 본다. 그곳이 -1일 경우 탐색이 끝나고 맨 처음 문자열을 비교하는 거고, 또 다른 인덱스 값이 있는 경우 반복의 여지가 있다.

if (P[i] == P[j + 1]) // 위의 경우를 통해 j값이 확정된 후, j == -1인 경우, 맨 처음 문자열이 같은지 비교를 통해 같으면 또 새로운 패턴이 생기는거고

// j == (다른 인덱스 값) 인 경우에는 어딘가 연속된 문자열이 있었다는 것이기 때문에, cache[i]에 해당 값을 저장하면서, 어딘가 연속된 패턴을 만든다.

cache[i] = j + 1;

}

}

void KMPAlgorithm()

{

int pos\_p = 0, pos\_t = 0;

for (int i = 0; i < T.length(); i++)

{

pos\_p = 0;

while (pos\_p < P.length() && pos\_t < T.length())

{

if (P[pos\_p] == T[pos\_t])

{

pos\_p++; pos\_t++;

}

else

{

if (pos\_p == 0)

pos\_t++;

else

pos\_p = cache[pos\_p - 1] + 1;

}

}

if (pos\_p >= P.length())

answer.push\_back(pos\_t - P.length());

}

}

int main(void)

{

fill(cache, cache + 100, -1);

cin >> T >> P;

FailureFunction();

KMPAlgorithm();

for (int e : answer)

cout << e << "번째 있습니다." << endl;

return 0;

}

2) 실행 결과

