**K-Means Clustering**

컴퓨터공학과 20171646 박태윤

**(코드 실행 결과)**

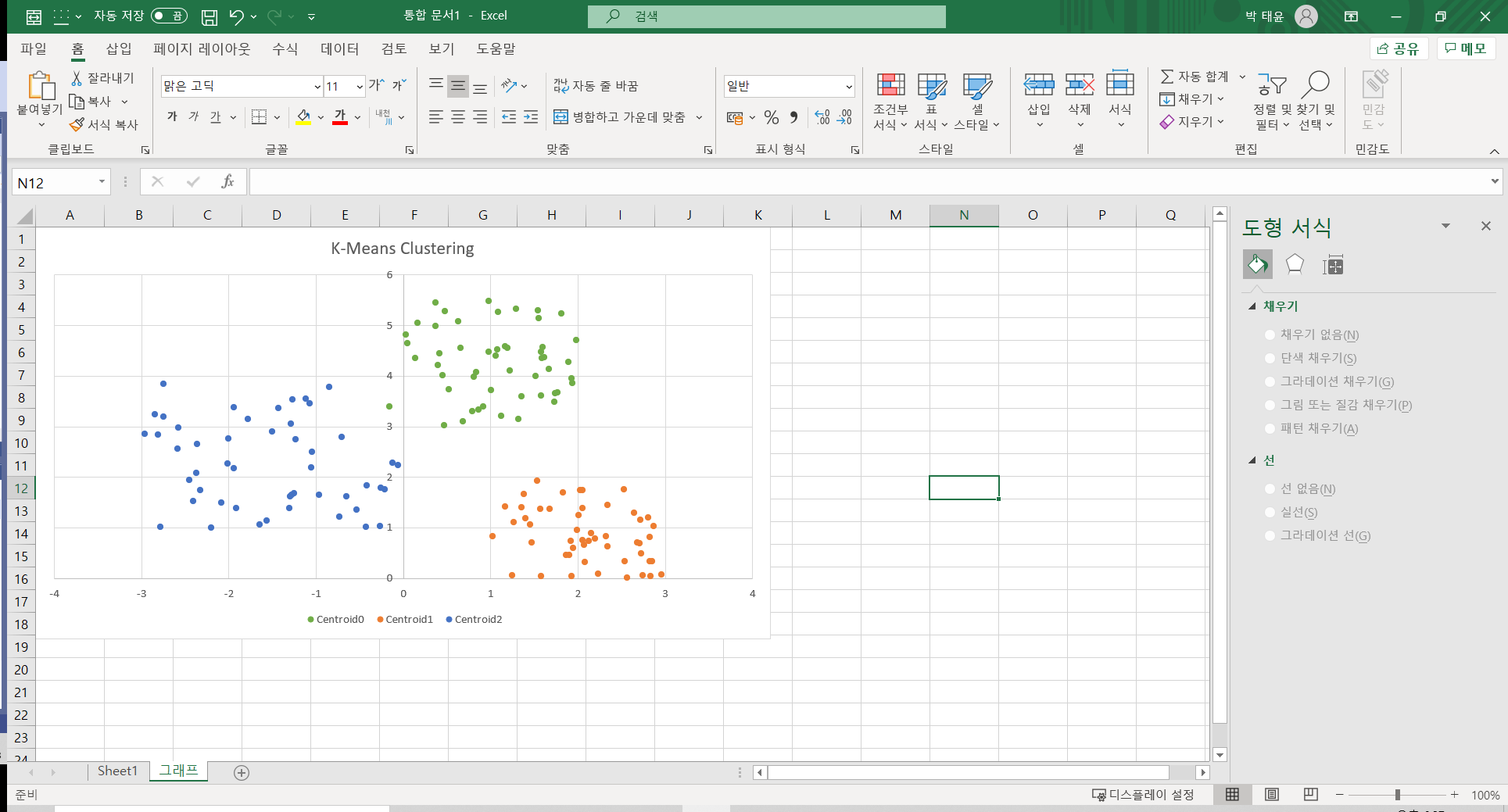
**-Input**

|  |  |
| --- | --- |
|  | 스크린샷, 노트북, 컴퓨터, 모니터이(가) 표시된 사진  자동 생성된 설명 |
|  |  |

**-output**

|  |  |  |
| --- | --- | --- |
|  |  |  |

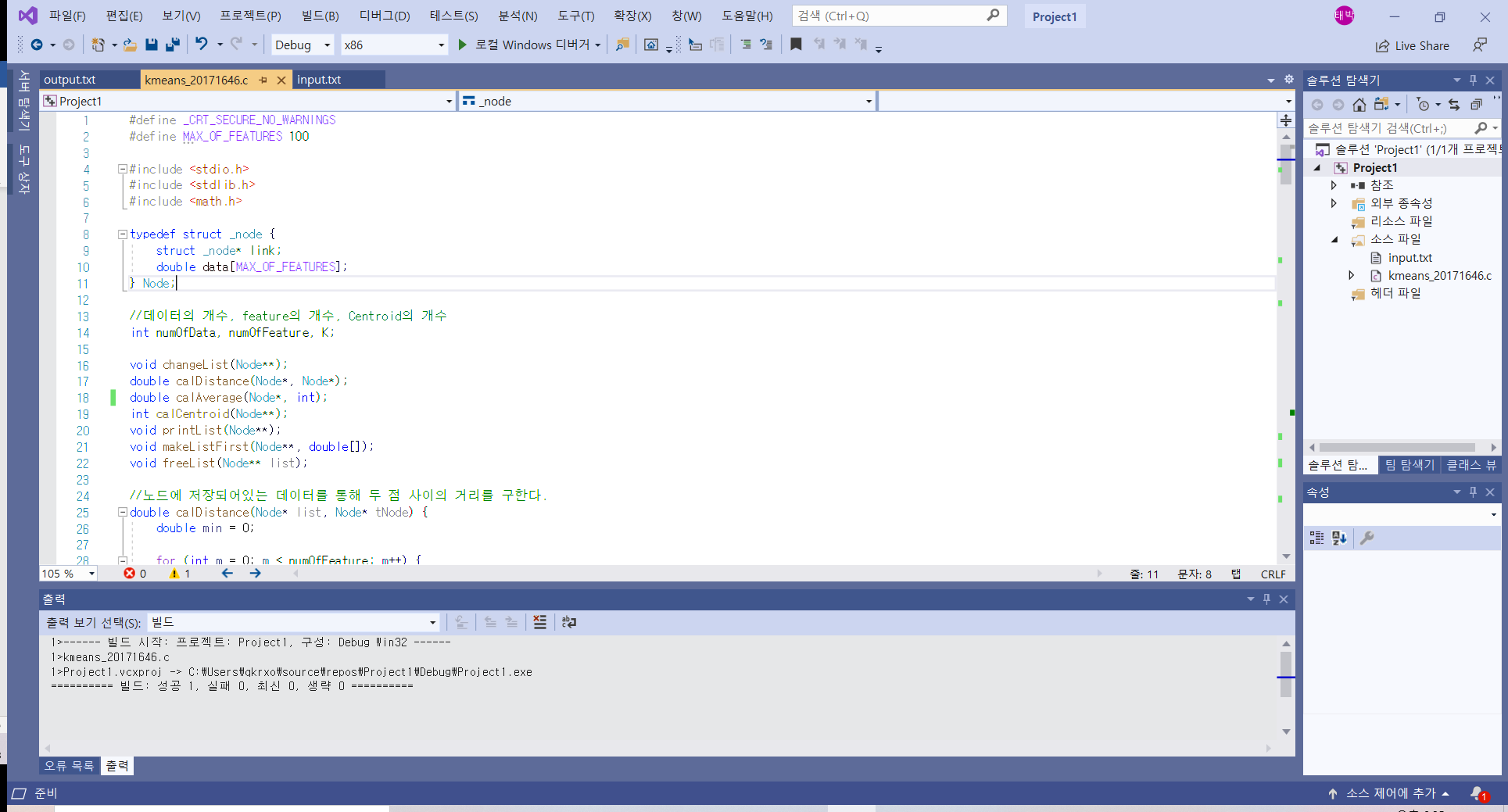
위의 그림과 같이 input.txt파일을 입력으로 받아 output.txt파일에 코드 실행 결과를 출력하였다.



출력 결과를 Centroid점을 기준으로 하여 그래프를 그려보았다. 데이터 군집들이 어느 정도 형성된 것을 확인할 수 있다.

**(코드 / 알고리즘 설명)**

K-Means Clustering은 주어진 데이터를 K개의 군집으로 묶는 알고리즘이다. 군집을 나타내는 Centroid라는 점이 존재하는데, 처음에 임의로 Centroid의 좌표를 설정해주고 각 데이터들마다 어떤 Centroid랑 가장 가까운지를 판별하여 군집을 만들어준다. 이후 같은 군집의 데이터들의 평균을 구하여 이 값을 이용하여 해당 Centroid를 업데이트 시켜준다. Centroid의 값이 바뀌면서 어떤 데이터와 가장 가까운 Centroid가 기존과 달라질 수 있다. 이런 경우 그 데이터가 속한 군집을 변경시켜가면서 계속 Centroid의 값을 군집 데이터들의 평균으로 업데이트 시켜준다. 이 과정을 더이상 Centroid들의 변화가 없거나 지정한 maximum Iteration에 도달할 때까지 반복한다.

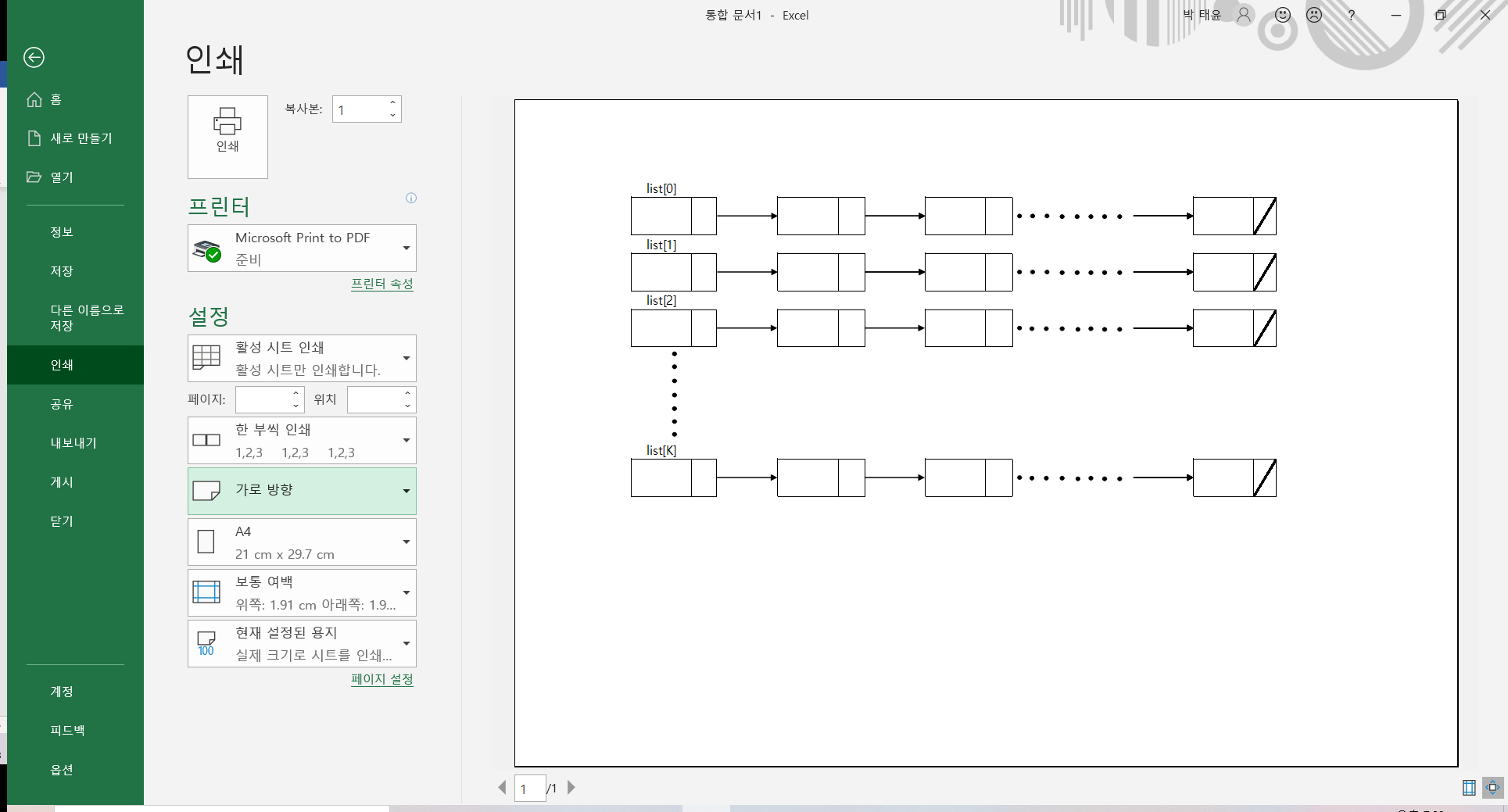


연결리스트를 만들기 위해 Node를 선언하였고, 노드 안에 들어가는 데이터는 double형태 MAX\_OF\_FEATURES(100)사이즈 배열로 선언하였다. input.txt파일 첫 줄에 쓰여져 있는 <데이터 개수> <데이터 feature 개수> <K값>을 받기 위해 전역변수로 numOfData, numOf Feature, K를 int type으로 선언하였다. 이후 changeList, calDistance, calAverage, calCentroid, printList, MakeListFirst, freeList 총 7개의 함수를 선언하였다.

(main 함수)

|  |  |
| --- | --- |
|  |  |

메인 함수는 다음과 같이 구현하였다. 랜덤seed값을 1000으로 설정하였으며 Node의 더블 포인터 형태로 list를 선언하였다. 이는 다음 그림과 같이 구현하기 위함이다.



각 행마다 첫 노드인 list[index]는 Centroid점의 좌표를, 그 이외의 노드에는 데이터 값으로 주어진 점들의 좌표를 가지고 있으며, 마지막 노드의 link는 NULL이다.

메인 함수에서 numOfData, numOfFeature, K값을 파일입력을 통해 받는다. 이후 위의 그림과 같이 list = (Node\*\*)malloc(sizeof(Node\*)\*K)를 하여 입력 받은 K만큼 공간 할당을 해준다. 이후 각 list[index]->data[]에 임의의 Centroid점을 설정하기 위해 실수형 난수를 넣어준다. 이후 각 list[index]에 데이터 값이 모두 들어갔다면 link를 NULL로 초기화 시켜준다. 그 다음double type 포인터형 변수 arr을 numOfFeature만큼 동적할당 시키고 arr에 input.txt에 있는 데이터를 한 줄씩(한 점의 좌표)입력시켜준다. 입력이 완료될 때 마다 makeListFirst함수를 호출하여 어느 Centroid점에 가장 가까운지를 판단하여 해당 list[index]에 연결시켜준다. 이후 Centroid점을 현재 연결되어 있는 점들의 평균 좌표로 이동시켜주는 calCentroid함수를 호출하는데, calCentroid가 0을 반환하면 반복문을 종료시키고 그렇지 않다면 50번 반복한다. calCentroid를 호출하면서 Centroid점의 좌표가 변경되었다면 모든 data점들을 살펴보면서 가장 가까운 Centroid점으로 연결을 바꿔주는 changeList함수 또한 호출한다. 반복문을 50번 반복했거나 calCentroid가 0을 반환하여 반복문이 종료되었다면 printList함수를 통해 output.txt에 결과를 출력해주고 동적할당을 진행한 arr배열의 메모리를 해제시키면서 freeList함수를 통해 list의 메모리를 모두 해제시켜준다.

(makeListFirst 함수)

|  |  |
| --- | --- |
| 스크린샷, 노트북, 컴퓨터, 모니터이(가) 표시된 사진  자동 생성된 설명 | 스크린샷, 노트북, 컴퓨터, 모니터이(가) 표시된 사진  자동 생성된 설명 |

list와 데이터로 주어진 한 점의 좌표를 가지고 있는 arr배열을 인자로 받는 함수이다. tempNode를 만들어서 이 노드에 arr에 저장되어 있는 데이터 값을 넣어준다. 이후 min을 처음에 calDistance함수를 사용하여 tempNode가 가리키는 점과 0번째 Centroid점의 거리로 초기화 시켜준다. 이후 for문을 통해 거리가 가장 짧은 Centroid점의 index를 골라준다. tmp변수는 반복문 첫 줄에서 (j = 1 ~ K)번째 Centroid점과 tempNode의 거리를 나타내는데, 현재 min보다 tmp가 작은 값을 나타낼 경우 index변수를 j로 초기화 시켜주고 min값을 tmp값으로 바꿔준다. 이러한 과정을 통해 가장 가까운 Centroid점을 고를 수 있다. Centroid점을 골랐으면 list[index]에 연결시켜주는데 이 때 list[index]에 아무것도 연결되어있지 않은 상태와 그렇지 않은 상태를 구분하여 연결시켜준다. 연결은 항상 list[index]바로 앞에 연결이 된다. 즉, 먼저 연결된 노드들은 makeListFirst를 반복하다보면 점점 뒤로 밀려나는 것을 의미한다.

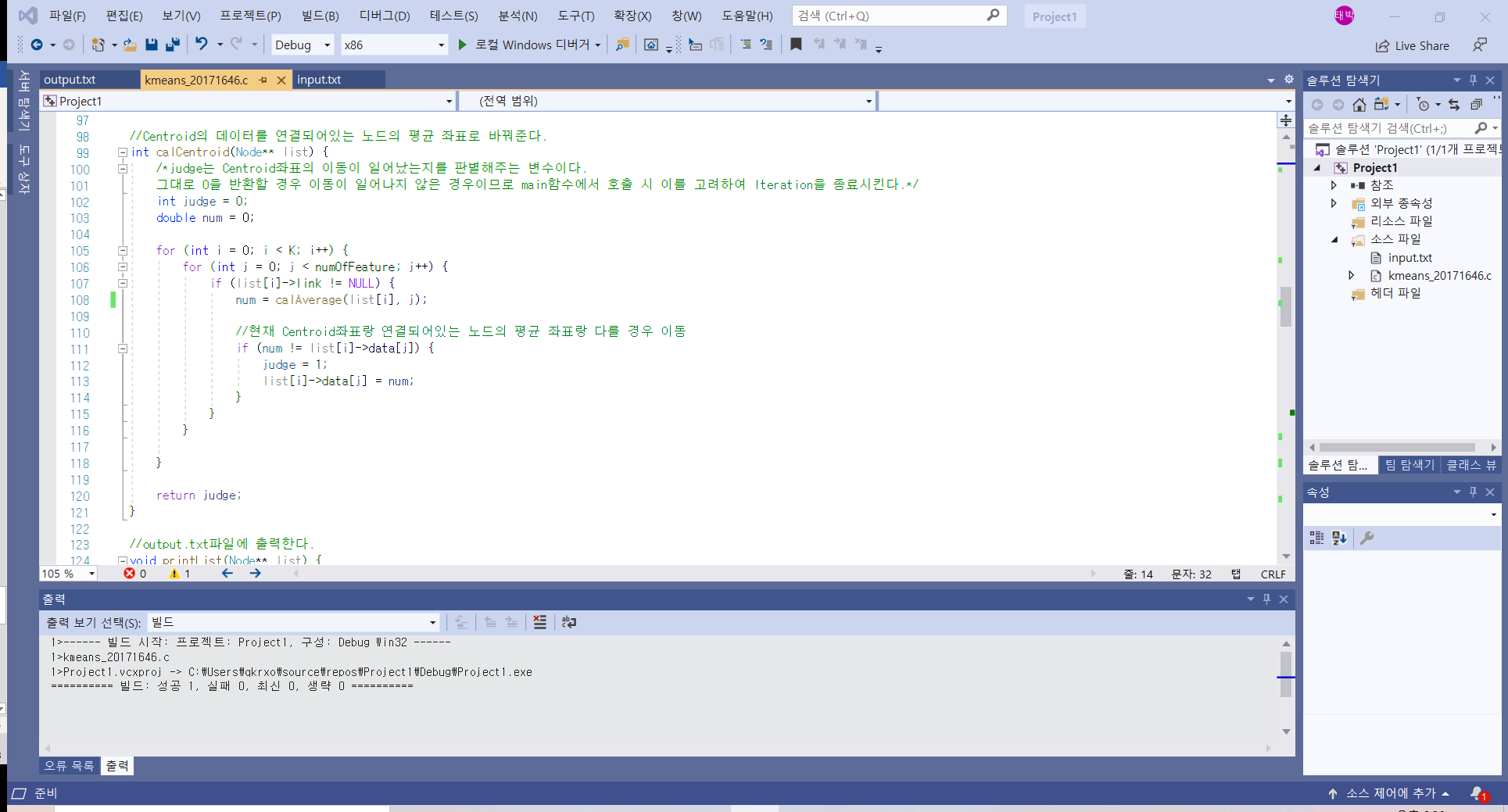
(calDistacne 함수)

스크린샷, 노트북, 컴퓨터, 모니터이(가) 표시된 사진

자동 생성된 설명

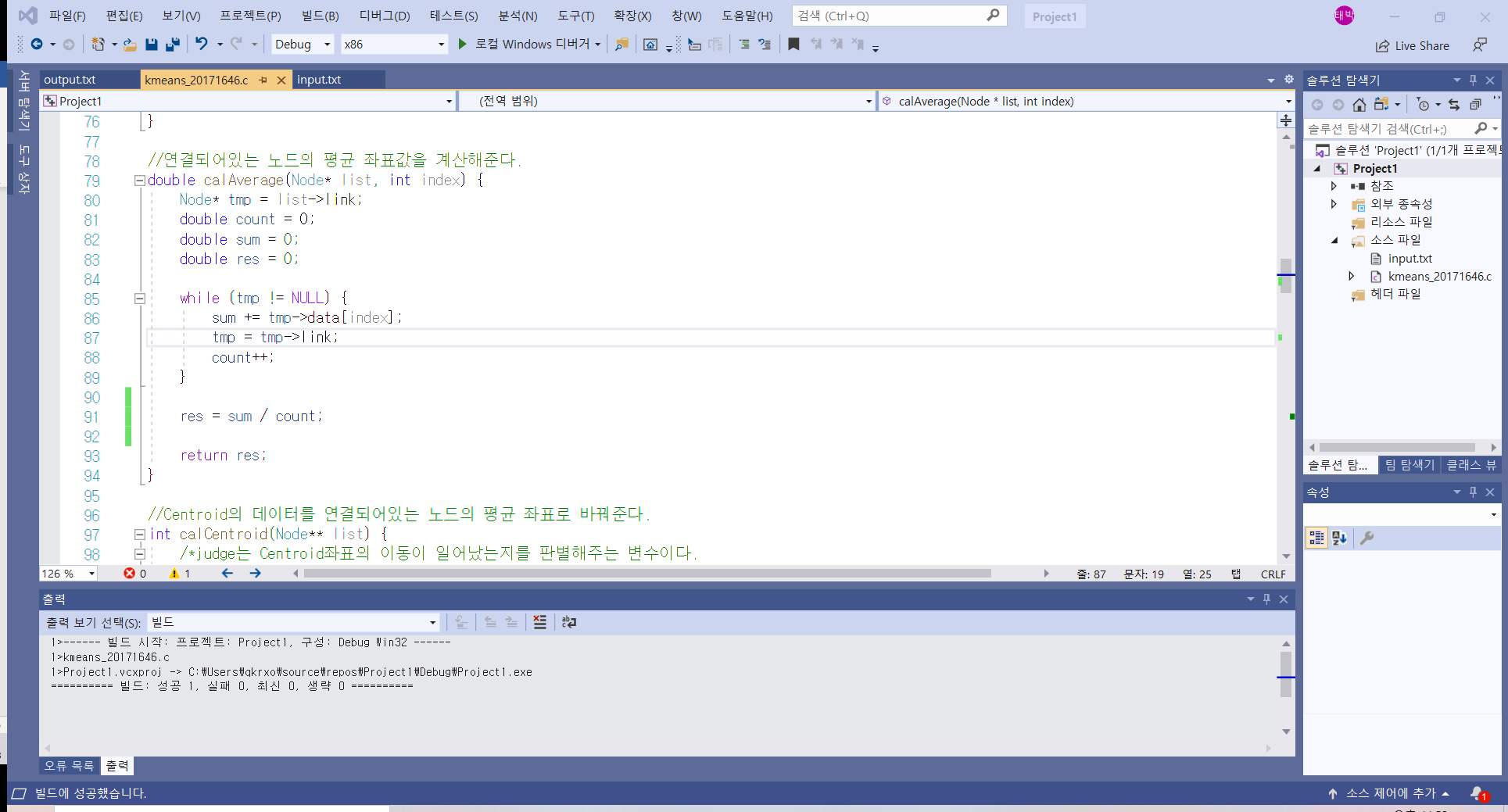
list와 tNode라는 parameter를 가지고 있다. Euclidean Distance를 사용하여 list에 저장되어 있는 점의 좌표와 tNode에 저장되어 있는 점의 좌표 사이의 거리를 return해준다.

(calCentroid 함수)



list[index]->data[] 즉, Centroid점의 좌표를 현재 연결되어있는 점들의 평균 좌표로 이동시켜주는 함수이다. 이중 for문이 사용되었는데 처음 for문은 K만큼, 안에 있는 for문은 numOfFeature만큼 반복한다. for문 안에 if문이 사용되었는데 현재 list[i]에 한 점이라도 연결되어 있는 경우에만 다음 코드를 실행하는 방식으로 구현하였다. if문 안에서 cal함수를 호출하여 return값을 num에 저장시켜주는데, cal함수는 연결되어있는 노드들의 좌표 평균값을 계산하여 반환해주는 함수이다. 반복문이 numOfFeature(=2)만큼 돌아가기 때문에 현재 입력의 경우에는 x와 y좌표의 평균값을 모두 구할 수 있다. 이후 num이랑 현재 Centroid점의 좌표값이랑 다른 경우에 judge변수의 값을 1로 바꿔주고 list[i]->data[j] = num을 통해 Centroid점을 이동시켜준다. 여기서 judge변수는 초기값은 0이였다가 Centroid점이 이동할 때에만 1로 값이 바뀌는 것으로 알 수 있는데, 현재 maximum Iteration이 50이고 Centroid점들의 이동이 더 없을 경우 50번 반복하는 과정을 중간에 종료시켜야하기 때문에 존재한다. 즉, calCentroid의 return값이 1인 경우는 한 점이라도 Centroid점의 이동이 발생한 것이고 0인 경우는 Centroid점의 이동이 발생하지 않은 것을 의미한다.

(CalAverage 함수)

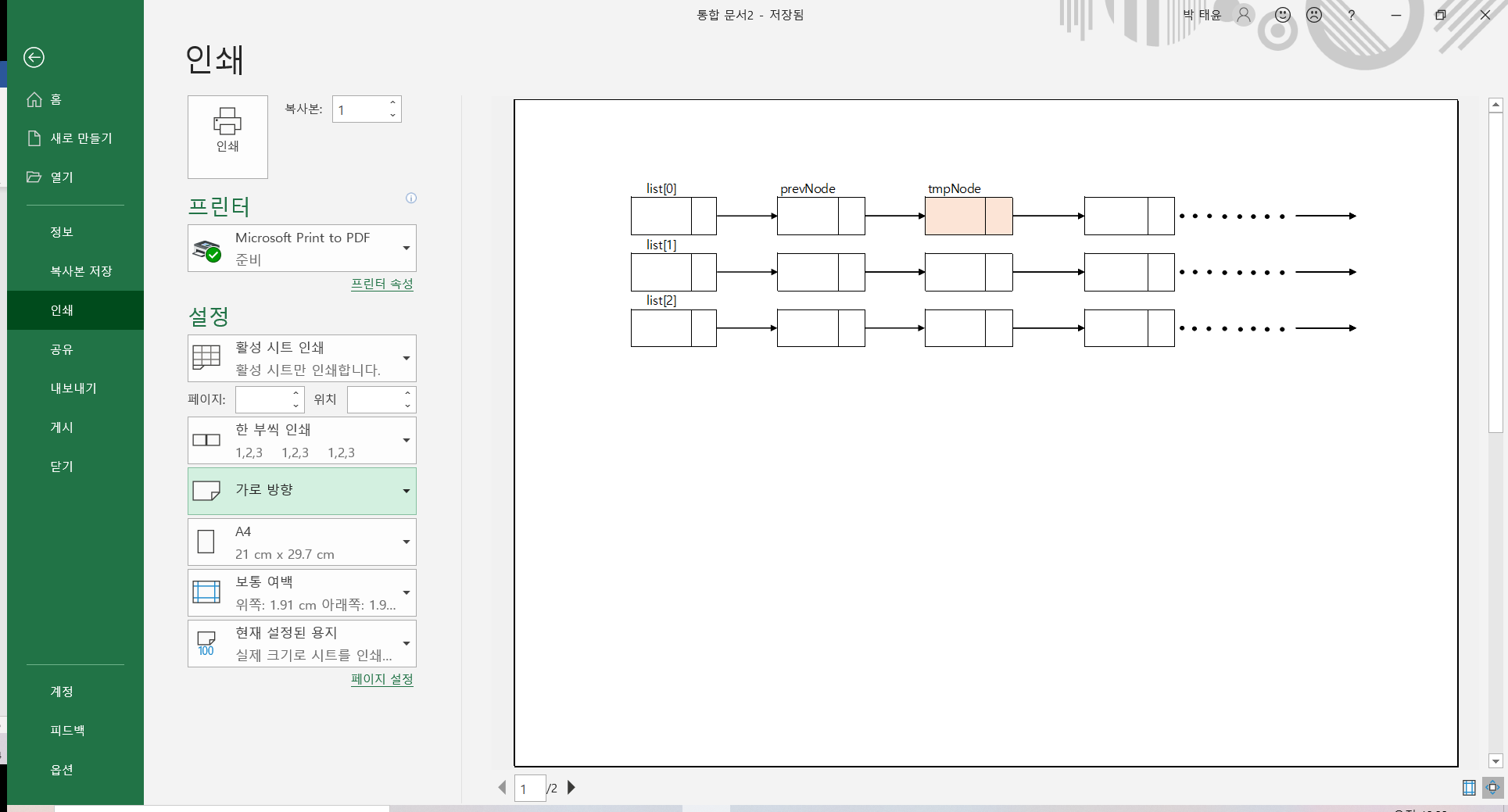


연결되어 있는 노드들 좌표의 평균을 구해주는 함수이다. list한 줄과 index를 parameter로 받으며 tmp를 통해 list에서 한 칸씩 이동하면서 sum에다 data값을 더해주고 count++도 해준다. 이 과정을 tmp가 NULL이 아닐 때까지, 즉 한 줄 끝에 도달할 때까지 반복하며 최종적으로 sum을 count로 나눈 res를 return해준다. index는 몇 번째 feature인지를 나타낸다.

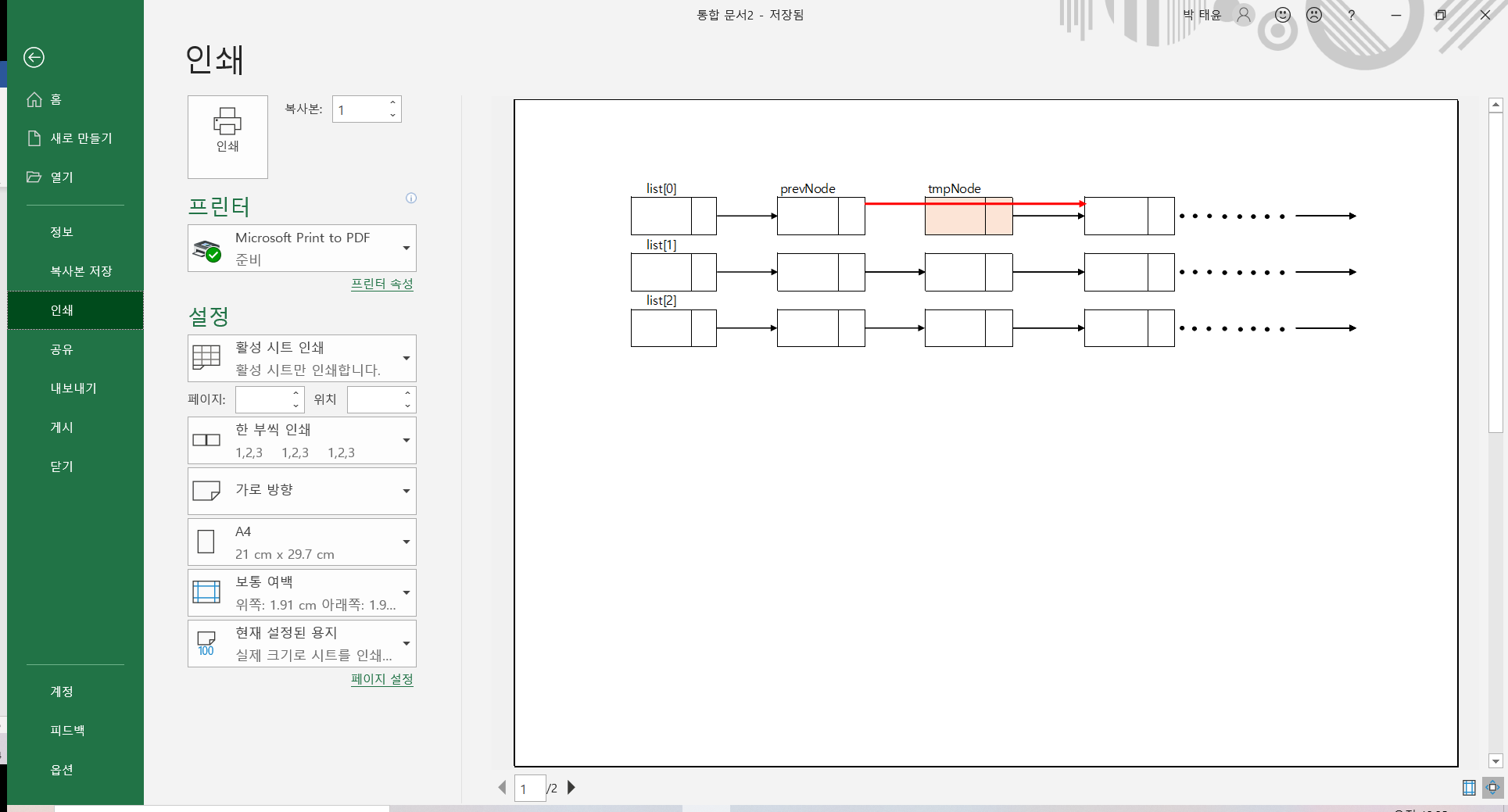
(changeList 함수)

|  |  |
| --- | --- |
| 텍스트이(가) 표시된 사진  자동 생성된 설명 | 스크린샷, 노트북, 컴퓨터, 모니터이(가) 표시된 사진  자동 생성된 설명 |

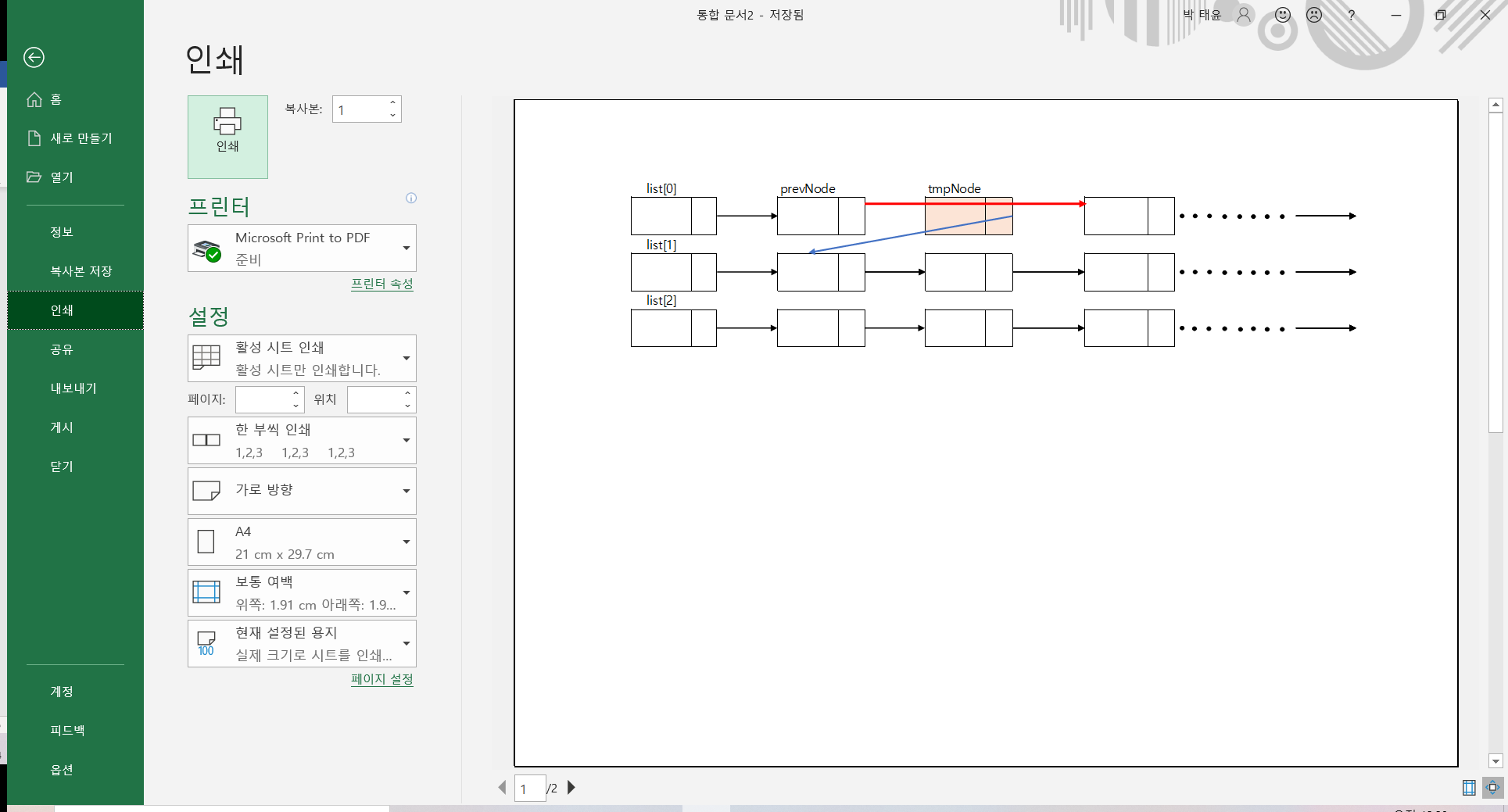
Centroid점의 좌표는 calCentroid함수에 의해 바뀐다. 이렇게 Centroid점의 좌표가 바뀌면 어떤 데이터 점이 예를 들어 원래 1번 Centroid에 가장 가까웠지만 calCentroid함수의 동작 이후 1번 보다 2번 Centroid가 더 가까운 점이 될 수도 있다. changeList는 이러한 경우를 고려하여 연결되어 있는 데이터 노드들의 연결을 바꾸어 주는 함수이다. Node\* 타입으로 tmpNode와 prevNode를 선언하였는데 이름 그대로 현재 보고 있는 노드와 그 전에 봤던 노드를 가리키는 변수이다. changeList함수는 for반복문과 그 안에 while문을 사용하여 구현하였다. for문은 K번 만큼 반복하며 for문 처음에 tmpNode와 prevNode의 위치를 각각 link[i]바로 앞에 연결되어 있는 노드, link[i]로 위치를 초기화 시켜주었다. index변수는 i로 값을 넣어주었다. while문은 tmpNode가 NULL이 아닐 때까지 반복을 하는데 이는 list[i]에 연결되어 있는 노드 중 마지막 노드에 도달할 때까지 반복한다는 의미이다. while문 처음에 min변수를 -1로 초기화 시켜주었는데, min변수는 tmpNode가 가리키고 있는 점과 가장 가까운 Centroid사이의 거리를 나타낸다. (두 점 사이의 거리는 음수가 될 수 없으므로 -1로 초기화를 시켜준다.) 이후 가장 가까운 Centroid의 index를 구하는 과정을 거친다. 이 과정을 거치게 되면 index에는 거리가 가장 가까운 Centroid의 인덱스 값이 들어가 있는데, 만약 i와 index가 같은 값이라면 현재 연결되어 있는 Centroid가 가장 가까운 Centroid인 것이기 때문에 이동이 필요하지 않다. 반대로 i랑 index가 다르다면 노드의 이동이 필요한 경우이기 때문에 if…else문을 통해 구분하여 처리해주었다. 그 과정은 다음과 같다.



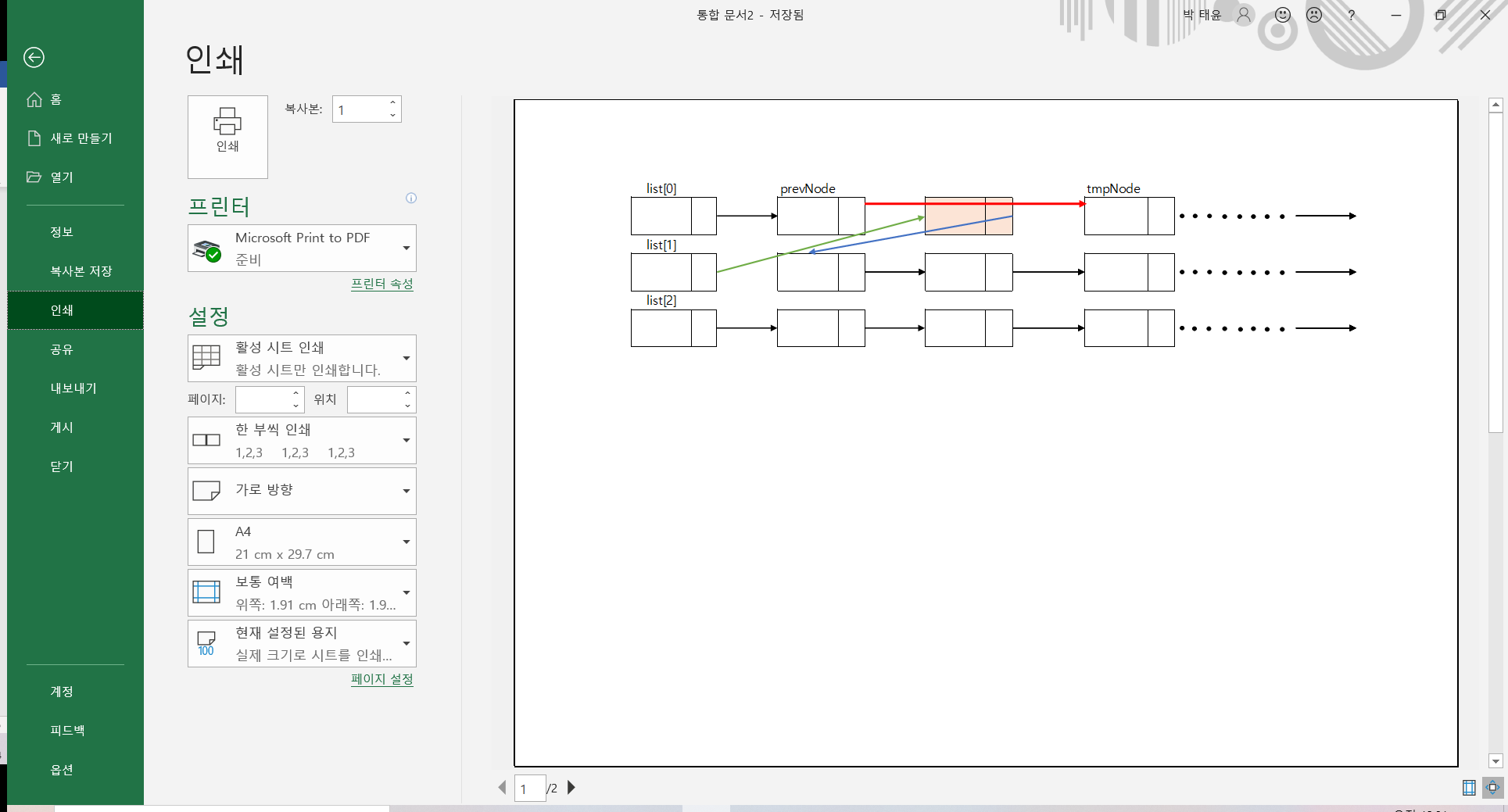
색칠된 노드가 list[0]에 연결이 되어있지만 현재 index변수가 1을 가리키고 있을 때, 즉 list[1]로 연결을 바꿔줘야한다면,



처음에 그림과 같이 prevNode의 link를 tmpNode->link로 바꿔준다.



이후 tmpNode의 link를 list[index]->link로 바꿔주는데, 현재 index는 1이기 때문에 위와 같다.



list[index]의 link를 tmpNode로 연결시키고 tmpNode를 prevNode->link로 이동시킨 후의 모습이다.

이동이 필요하지 않은 경우는 다음에 연결되어 있는 노드를 가지고 어느 Centroid랑 가장 가까운지를 판별해야 하기 때문에 prevNode = tmpNode // tmpNode = tmpNode->link를 하여 해당 list에서 한 칸 씩 이동시킨다.

이렇게 changeList함수랑 앞에서 소개한 calCentroid함수를 메인함수에서 반복해가면서 군집 분류를 정할 수 있다.

(printList 함수)

스크린샷, 컴퓨터, 노트북, 모니터이(가) 표시된 사진

자동 생성된 설명

군집 분류가 끝난 list를 output.txt에 출력해주는 함수이다. Node\* temp를 선언해주고 이후 K 반복하는 for반복문을 시작한다. 군집 번호를 먼저 출력해주고 temp는 Centroid점의 좌표는 출력하지 않기 때문에 list[i]->link를 가리키게 한다. 이후 while문에서 temp가 NULL이 아닐 때까지, 즉 한 군집에 연결되어 있는 마지막 노드에 도달할 때까지 출력을 진행한다. temp가 NULL에 도달하면 다음 군집을 for문을 통해 볼 수 있다.

(freeList 함수)

스크린샷, 노트북, 컴퓨터, 모니터이(가) 표시된 사진

자동 생성된 설명

동적 할당된 list의 메모리를 해제해주는 함수이다. delNode와 curNode를 선언하였다. for문은 K번 반복을 하며 curNode는 처음에 list[i]->link를 가리킨다. 이후 while문을 시작하는데 curNode가 list[i]에 연결되어 있는 노드 중 가장 마지막 노드를 가리킬 때까지 반복한다. delNode는 처음에 curNode와 같은 노드를 가리키고, curNode는 한 칸 이동시켜준다. 이후 delNode가 가리키는 노드의 메모리를 해제해준다. 이 과정을 반복한 뒤 while문을 빠져나오면 Centroid의 정보가 담긴 list[i]노드의 메모리를 해제해준다.