**자료구조 과제 – Kruskal’s Algorithm 프로그래밍 프로젝트**

정보대학 컴퓨터학과 2017320108

고재영

목차

1. 구현환경
2. 프로그램 소개 및 주의사항
3. 프로그램 사용방법, 설명 (콘솔 스크린샷 첨부)
4. 소스코드
5. 구현환경

OS: Windows 10

Language: C language

Tool: Visual Studio 17

1. 프로그램 소개

이 프로그램은 Minimum Spanning Tree (이하 MSP라고 축약)를 만드는 데에 사용할 수 있는 여러 알고리즘 중 Kruskal’s Algorithm을 이용하여 해결하는 프로그램입니다. 따라서, 해당 알고리즘 과정을 가시화하여 유저가 콘솔창을 통해 볼 수 있도록 할 것입니다.

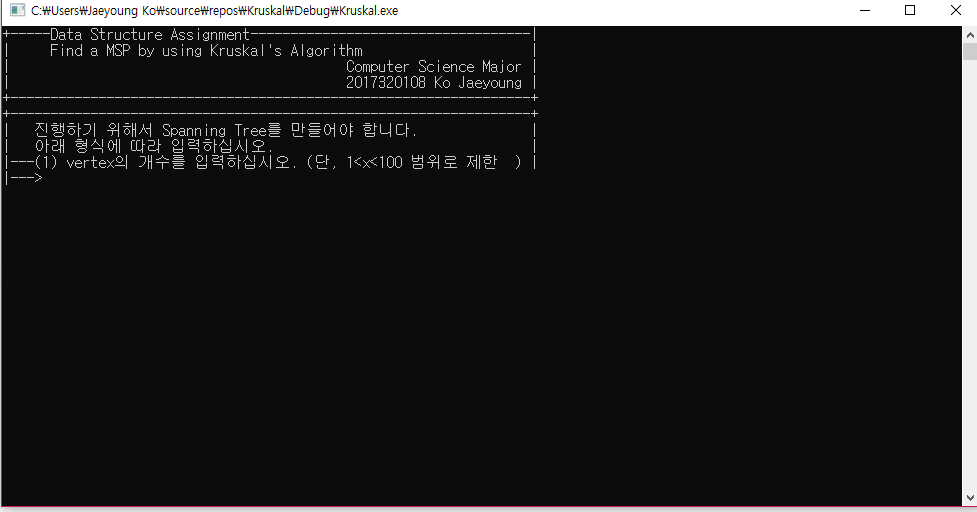
과정을 진행하기 위해서, 먼저 Spanning Tree가 주어져야 하기 때문에 유저로부터 Vertex와 두vertices를 연결하는 Edge, 그리고 해당 edge의 weight를 입력받을 것입니다. 유저는 반드시 입력하는 데에 있어서 콘솔창에서 안내하는 형식에 준수해야 하고, 이를 어김으로 인한 오류는 크게 핸들하지 않을 것입니다.

유저가 위의 입력들을 마친 이후에, 차례차례 취하는 edge를 하나씩 보여주고, 최종적으로 취해진 edge들의 weight값 총합까지 출력해줍니다. 따라서 유저는 MST가 해당 edge들로 이루어짐을 알 수 있습니다.

1. 프로그램 사용법 및 설명 (스크린샷 첨부)

지금부터 프로그램 사용 과정을 직접 시연하면서 설명해 드리겠습니다. 입력 값은 교재 309 페이지의 그림 6.22에 나와 있는 값들을 그대로 이용할 것입니다.

1. 시작화면



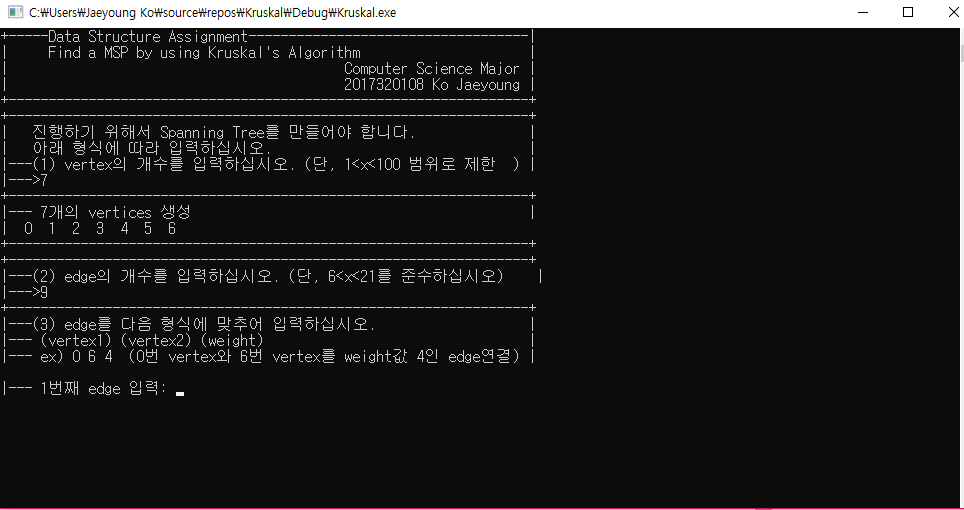
1. 유저 입력

유저 입력은 크게 세 가지입니다.

* Vertex의 개수 : vertex의 개수를 입력하면, 해당 개수만큼 자동으로

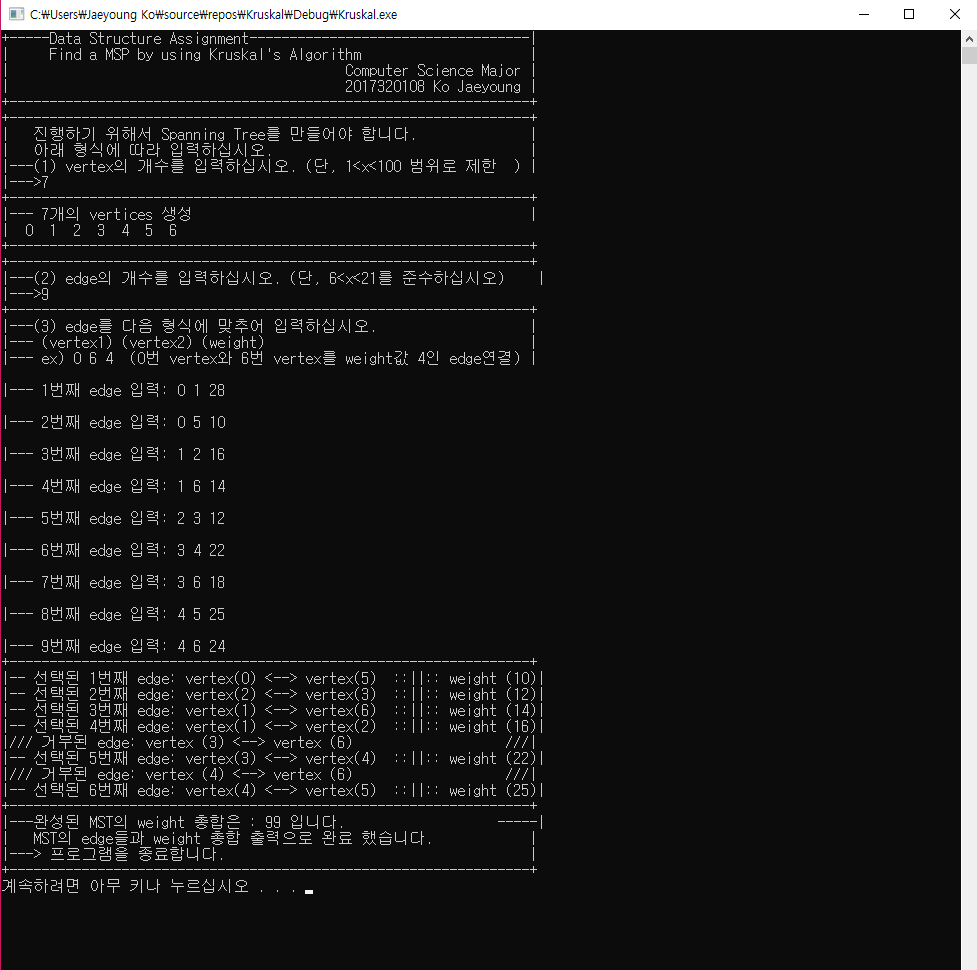
0번 vertex, 1번 vertex, … (vertex개수 - 1)번 vertex로 이름이 정수값으로 배정됩니다.

또한, max vertex 개수가 100으로 제한되어 있습니다. 이 규정을 벗어난 에러에 대해 핸들하지 않을 것입니다.



* Edge의 개수 : edge의 개수를 입력합니다. 단, 원활하게 MST를 만들기 위해서는 적어도 (vertex 개수 – 1)개 보다 크고, vertex C 2 개 보단 적은 수의 입력을 받아야 합니다. 이 규정을 벗어난 에러에 대해 핸들하지 않을 것입니다.
* 각 Edge 입력 : 입력 규정은 (vertex1) (vertex2) (weight) 입니다. 이를 어긋난 에러에 대해서 핸들하지 않습니다.

1. 최종 종료 화면



edge값들까지 다 입력하고 나면, kruskal algorithm이 수행되면서 그에 부합하는 edge들을 취합니다. 콘솔창으로는 알 수 없지만, weight값에 따라 edge들이 소팅되어서 그 순서대로 삽입을 하고, 위 콘솔에서 알 수 있듯이 cycle을 만들게 되는 edge는 거부된 edge라고 중간 과정에서 표시를 해줍니다.

7개의 vertex이기 때문에 다음과 같이 6개의 edge들이 선택되어 MST의 weight 총합 까지 99로 계산하여 출력합니다. 그 이후 프로그램이 정상적으로 마치면서 종료합니다.

1. 소스코드
2. #include <stdio.h>
3. #include <stdlib.h>
4. #define MAX\_VERTICES 100 //최대 vertex의 개수를 100으로 정함
5. #define MAX\_EDGES 4950 //최대 edge의 개수 정함
6. #define SWAP(a,b) {int temp; temp = a; a = b; b = temp;} //a와 b를 교환해주는 스왑함수
7. typedef struct Edge{ //Edge란 구조체 선언 - 2개의 vertex와 weight값
8. int v1;
9. int v2;
10. int weight;
11. }Edge;
12. int parent[MAX\_VERTICES]; //각 vertex의 패런트 노드
13. Edge e[MAX\_EDGES]; //edge에 관한 배열
14. void Init(int n) {
15. for (int i = 0; i < n; i++) {
16. parent[i] = -1; //패런트 노드 초기화
17. }
18. }
19. void Union(int p1, int p2) { //forest에서 두 트리를 연결 매개변수는 두 트리 각각의 루트노드
20. if (p1 < p2)
21. parent[p2] = p1;
22. else
23. parent[p1] = p2;
24. }
25. int Find(int v) { //vertex가 어떤 트리에 속하는 지, 해당 트리의 root 노드를 반환
26. while (parent[v] >= 0) {
27. v = parent[v];
28. }
29. return v;
30. }
31. void sort(int edges) { //Edge 리스트를 weight 순서대로 정렬시켜 줌
32. for (int i = edges; i > 1; i--) { //소팅의 알고리즘은 편하게 버블소팅으로 함.
33. for (int j = 1; j < i; j++){
34. if (e[j - 1].weight > e[j].weight){
35. SWAP(e[j - 1].weight, e[j].weight);
36. SWAP(e[j - 1].v1, e[j].v1);
37. SWAP(e[j - 1].v2, e[j].v2);
38. }
39. }
40. }
41. }
42. int CheckCycle(int v1, int v2) { //cycle 판별 함수 - 두 vertices에 대해 find통해 조상같으면 cycle임
43. if (Find(v1) == Find(v2)) { //두 vertex에 대해 같은 집합에 속해있는가?
44. //같은 집합에 속해있으면 cycle이 형성될 것임
45. return 0;
46. }
47. else {
48. //그렇지 않다면 같은 집합에 속해있지 않아 cycle 형성안해서 괜찮음.
49. Union(v1, v2);
50. return 1;
51. }
52. }
53. void Kruskal(int vertices, int edges) {
54. int weightcount = 0; //MST의 weight값 출력할거임
55. int edgecount = 0; //edgecount가 vertex개수 - 1이 되면 종료할 거임
56. printf("+-----------------------------------------------------------------+\n");
57. for (int i = 0; i < edges && edgecount != vertices - 1; i++) {
58. if (CheckCycle(e[i].v1, e[i].v2)) { //사이클 형성하지 않는다면
59. edgecount += 1;
60. weightcount += e[i].weight;
61. printf("|-- 선택된 %d번째 edge: vertex(%d) <--> vertex(%d) ::||:: weight (%d)|\n", edgecount, e[i].v1, e[i].v2, e[i].weight);
62. }
63. else { //사이클을 형성하면 거부
64. printf("|/// 거부된 edge: vertex (%d) <--> vertex (%d) ///|\n", e[i].v1, e[i].v2);
65. }
66. }
67. printf("+-----------------------------------------------------------------+\n");
68. printf("|---완성된 MST의 weight 총합은 : %d 입니다. -----|\n", weightcount);
69. printf("| MST의 edge들과 weight 총합 출력으로 완료 했습니다. |\n");
70. printf("|---> 프로그램을 종료합니다. |\n");
71. printf("+-----------------------------------------------------------------+\n");
72. }
73. void InitialScreen() {
74. printf("+-----Data Structure Assignment-----------------------------------|\n");
75. printf("| Find a MSP by using Kruskal's Algorithm |\n");
76. printf("| Computer Science Major |\n");
77. printf("| 2017320108 Ko Jaeyoung |\n");
78. printf("+-----------------------------------------------------------------+\n");
79. }
80. void InputScreenVertex() {
81. printf("+-----------------------------------------------------------------+\n");
82. printf("| 진행하기 위해서 Spanning Tree를 만들어야 합니다. |\n");
83. printf("| 아래 형식에 따라 입력하십시오. |\n");
84. printf("|---(1) vertex의 개수를 입력하십시오. (단, 1<x<100 범위로 제한 ) |\n");
85. printf("|--->");
86. }
87. void ShowVertex(int vertices) {
88. printf("+-----------------------------------------------------------------+\n");
89. printf("|--- %d개의 vertices 생성 |\n", vertices);
90. printf("| ");
91. for (int i = 0; i < vertices; i++) {
92. printf("%d ", i);
93. }
94. printf("\n+-----------------------------------------------------------------+\n");
95. }
96. void InputScreenEdge(int vertices) {
97. int maxedgenum = vertices \* (vertices - 1) / 2;
98. printf("+-----------------------------------------------------------------+\n");
99. printf("|---(2) edge의 개수를 입력하십시오. (단, %d<x<%d를 준수하십시오) |\n", vertices - 1, maxedgenum);
100. printf("|--->");
101. }
102. void InputScreenEdgeWeight() {
103. printf("+-----------------------------------------------------------------+\n");
104. printf("|---(3) edge를 다음 형식에 맞추어 입력하십시오. |\n");
105. printf("|--- (vertex1) (vertex2) (weight) |\n");
106. printf("|--- ex) 0 6 4 (0번 vertex와 6번 vertex를 weight값 4인 edge연결) |\n");
107. }
108. int main(){
109. int vertices, edges; //생성하는 vertex의 개수, edge의 개수
110. InitialScreen(); //처음 안내화면
111. InputScreenVertex(); //vertex 개수 입력
112. scanf\_s("%d", &vertices);
113. Init(vertices);
114. ShowVertex(vertices); //vertex 개수만큼 vertex이름이 정수로 붙여짐
115. InputScreenEdge(vertices); //edge 개수 입력 (단, vertex 개수에 따라 맥스값 제한)
116. scanf\_s("%d", &edges);
117. InputScreenEdgeWeight(); //edge 개수만큼 edge를 설정
118. for (int i = 0; i < edges; i++) {
119. printf("\n|--- %d번째 edge 입력: ", i+1);
120. scanf\_s("%d %d %d", &e[i].v1 , &e[i].v2, &e[i].weight);
121. }
122. sort(edges); //edge의 weight 순으로 정렬시켜줌
123. Kruskal(vertices, edges); //Kruskal 알고리즘을 통해 MSP완성
124. //완성 시 선택된 edge들 출력, 그리고 weight 총합 출력
125. system("pause");
126. return 0;
127. }