프로그래밍 실습 1

```
#include<stdio.h>
#include<Windows.h>

#define MAX_VERTICES 8
#define MAX_EDGES 12

#define TRUE 1
#define FALSE 0
#define INF 1000L

typedef struct
{
	int dist[MAX_VERTICES];
	char edge[MAX_VERTICES][2];
}dist;

int weight[MAX_VERTICES][MAX_VERTICES];

int selected[MAX_VERTICES];
dist dists;

int get_min_vertex(int n)
{
	int v,i;

	for(i=0;i<n;i++)
	{
		if(!selected[i])
		{
			v = i;
			break;
		}
	}

for(i=0;i<n;i++)
	{
		if(!selected[i] &&(dists.dist[i] < dists.dist[v]))
		{
			v = i;
		}
	}

	return (v);
}

void prim(int s,int n)
{
	int i,u,v;

	for(u=0;u<n;u++)
	{
		dists.dist[u]=INF;
		selected[u] = FALSE;
	}
```

```
	dists.dist[s] = 0;

	for(i=0;i<n;i++)
	{
		u=get_min_vertex(n);
		selected[u] = TRUE;
		if(dists.dist[u] == INF)
		{
			return;
		}
		if(u != s)
		{
			printf("(%c, %c) ",dists.edge[u][0],dists.edge[u][1]);
		}

		for(v=0;v<n;v++)
		{
			if(weight[u][v]!=INF)
			{
				if(!selected[v] && weight[u][v]<dists.dist[v])
				{
					dists.dist[v] = weight[u][v];
					dists.edge[v][0] = 65+u;
					dists.edge[v][1] = 65+v;
				}
			}
		}
	}
}

int main()
{
	int i,j;
	int u,v,temp;
	char u_temp,v_temp;

	for(i=0;i<MAX_VERTICES;i++)
	{
		for(j=0;j<MAX_VERTICES;j++)
		{
			if(i==j)
			{
				weight[i][j] = 0;
			}else
			{
				weight[i][j] = INF;
			}
		}
	}
	printf("정점의 개수와 간선의 개수를 그래프에 맞게 변경하셨나요₩n");
	printf("정점_1 정점_2 가중치를 다음 형태대로 입력하세요₩n");
	printf("예제) a b 10₩n₩n");
```

```
	for(i=0;i<MAX_EDGES;i++)
	{
		fflush(stdin);
		scanf("%c %c %d",&u_temp,&v_temp,&temp);
		u = (int)u_temp - 97;
		v = (int)v_temp - 97;
		weight[u][v] = temp;
		weight[v][u] = temp;
	}

	prim(0,MAX_VERTICES);

	system("PAUSE");
	return 0;
}
```

프로그래밍 실습 2

```
#include<stdio.h>
#include<conio.h>
#include<windows.h>

typedef struct node *tree_ptr;
typedef struct node
{
	tree_ptr left;
	int data;
	tree_ptr right;
}node;

void preorder(tree_ptr p)
{
	if(p!=NULL)
	{
		printf("%3d)",p->data);
		preorder(p->left);
		preorder(p->right);
	}
}

void inorder(tree_ptr p)
{
	if(p!=NULL)
	{
		inorder(p->left);
		printf("%3d)",p->data);
		inorder(p->right);
	}
}

void postorder(tree_ptr p)
{
	if(p!=NULL)
	{
		postorder(p->left);
		postorder(p->right);
		printf("%3d)",p->data);
	}
}

void insert(int x,tree_ptr *p)
{
	if(*p==NULL)
	{
		*p=(tree_ptr)malloc(sizeof(node));
		(*p)->data=x;
		(*p)->right=NULL;
		(*p)->left=NULL;
	}else if(x<(*p)->data)
	{
		insert(x,&((*p)->left));
	}else
	{
		insert(x,&((*p)->right));
```

```
	}
}

void main()
{
	tree_ptr prn=NULL;
	char ch;
	int x;

	printf("₩n### Select number ###₩n");
	printf("1. 전위 탐방(preorder)			search₩n");
	printf("2. 중위 탐방(inorder)			search₩n");
	printf("3. 후위 탐방(postorder)		search₩n");
	printf("4. insert₩n");
	printf("5. ######## end ########₩n");
	printf("먼저 4번을 선택해서 이진 트리를 만들고 그 이진 트리에서 탐방 하세요.₩n");
	printf("4번의 입력 형태는 4 5와 같은 형태로 계속 같은 방법을 이용해서 이진 트리를 만드세요.₩n");
	do
	{
		scanf("%c", &ch);

		if(ch == '5')
		{
			break;
		}

		switch(ch)
		{
		case '1':
			preorder(prn);
			break;
		case '2':
			inorder(prn);
			break;
		case '3':
			postorder(prn);
			break;
		case '4':
			scanf("%d",&x);
			insert(x,&prn);
			break;
		}
	}while(1);

	printf("######## end ########₩n");
	system("PAUSE");
}
```