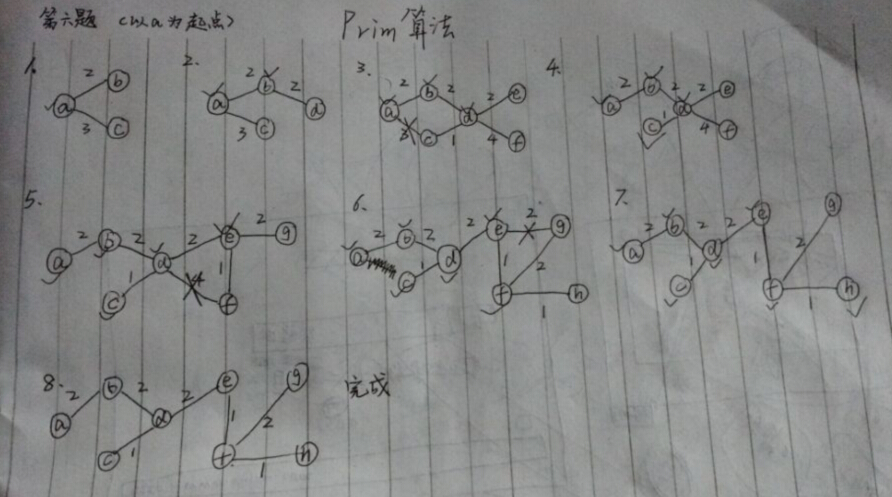
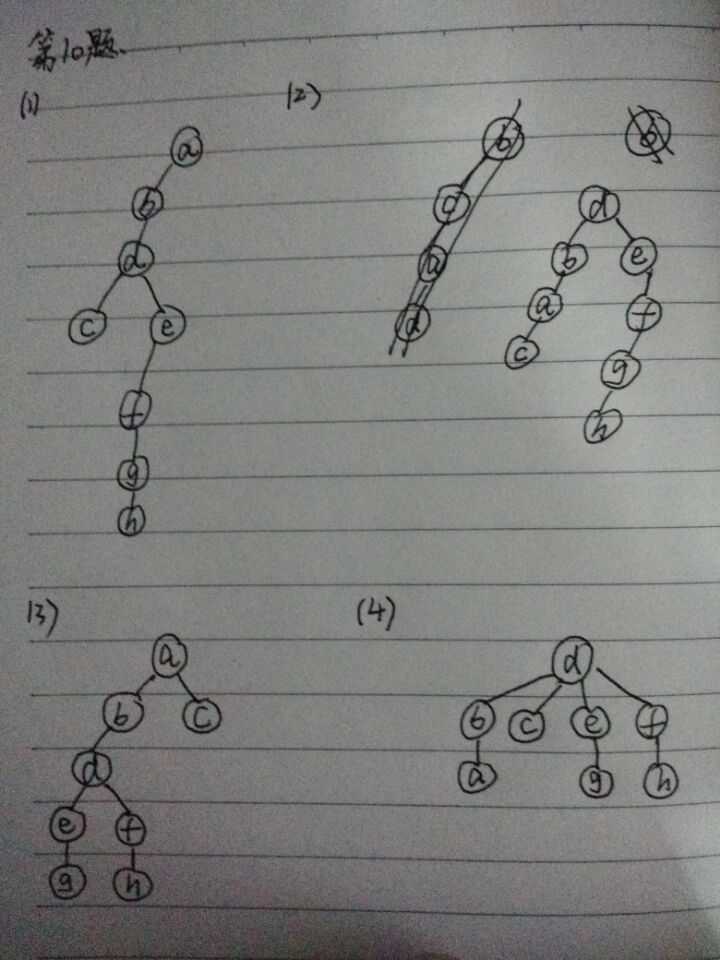
**本次作业：6，10，26，30，44**

第6题：



`

第10题：



第26题：

void floyd(int\*\* adj\_matrix,int\*\* min\_distance,int length){

for(int i=0;i<length;i++){

for(int j=0;j<length;j++){

min\_distance[i][j] == adj\_matrix[i][j];

}

}

for(int k=0;k<n;k++){

for(int i=0;i<n;i++){

for(int j=0;j<n;j++){

int new\_distance = min\_distance[i][k] + min\_distance[k][j];

min\_distance[i][j] = min\_distance[i][j] < new\_distance ? min\_distance[i][j] : new\_distance;

}

}

}

}

// 返回中心点的索引

int calculateCenterPoint(int\*\* adj\_matrix,int\*\* min\_distance,int length){

floyd(adj\_matrix,min\_distance,length);

int each\_max[n];

for(int i=0;i<n;i++){

each\_max[i] = min\_distance[i][0];

for(int j=1;j<n;j++){

each\_max[i] = each\_max[i] > min\_distance[i][j] ? each\_max[i] : min\_distance[i][j];

}

}

int min\_index = 0;

int min\_each = each\_max[0];

for(int i=1;i<n;i++){

if(each\_max[i] < min\_each){

min\_each = each\_max[i];

min\_index = i;

}

}

return min\_index;

}

第30题：

typedef struct Vertex{

char mark;

struct Vertex\* next;

}Vertex;

typedef struct Stack{

Vertex\* elementVertex;

struct Stack\* next;

}Stack;

// id从1开始，比索引大1

// 该函数实现了给定一个出发点遍历所有能到达的点，如果是强连通图，就可以完全遍历，如果是弱连通图或者不连通的，就需要在表里面搜没有标记过的点继续进行这个函数操作。

void DeepFirstSearch(Vertex\* table,int startId){

Stack\* top = NULL;

push(&top,table+startId-1);

(table+startId-1)->mark = 1;

for(top){

Vertex\* currentVertex = top->elementVertex;

while((currentVertex = currentVertex->next)!=NULL){

if(!((table+currentVertex.id-1)->mark)){

break;

}

}

// 如果找到未遍历的邻接点，就进栈并标记，否则就令栈顶出栈

if(currentVertex){

(table+currentVertex.id-1)->mark=1;

push(&top,table+currentVertex.id-1);

printf("(%d,%d)\n",top->next->id,top->id);

}else{

pop(&top,table);

}

}

}

第44题：

