Algorithm homework

Author Haihan Gao

Num PB1803080

链表表示的加权合并

```
typedef struct Node{
    Node *father;
    int power;
    //int value;
}set;
set Set[numberofnode];
Node Make-set(int value){
    set* Node;
    Node=(set*)malloc(sizeof(set));
    Node->father=Node;
    //Node->value=value;
    Node->power=0;
    return *Node;
}
Node *Find-set(set *node){
    if(node->father==node)
        return node;
    else
        return Find-set(node->father);
}
void Union(set *node1,set *node2){
    set *Node1;
    set *Node2;
    Node1=Find-set(node1);
    Node2=Find-set(node2);
    if(Node1->power<Node2->power){
        Node1->father=Node2;
        Node2->power++;
    }
    else{
        Node2->father=Node1;
        Node1->power++;
    }
    return ;
}
```

0-1背包问题

用一个数组weight[1..n]存储各种物品的质量，value[1...n]存储各种物品的价值，W代表背包可以存放物品的总质量，假设V[n_1,W_1]代表前n_1个物体，在总质量不超过W_1的条件下可以达到的价值最大值，显然，存在如下结论

V[n_1,W_1]=0 when n_1=0 or W_1=0 or W_1<0

目标问题是求V[n,W]

对于第n_1件物品，存在两种情况

该物品在背包里，则转换为求V[n_1-1,W_1-wieght[n_1]]+value[n_1]

不在背包中，则转化为求V[n_1-1,W_1]

求最大值，就是求max(V[n_1-1,W_1-wieght[n_1]]+value[n_1],V[n_1-1,W_1])

```
int max(int a,int b){
    return (a>b)?a:b;
}
int value(int i,int j){
    if(i<=0)
        return 0;
    else if(j<=0)
        return 0;
    else
        return v[i,j]=max(v[i-1,j],v[i-1,j-weight[i]]+value[i]);

}
```

Analyze the Time complexity of the Algorithm

利用二维数组计算V[n,W]，对于每个点计算的时间复杂度为O(1),那么总的时间复杂度为O(nW)