计算非循环图的传递闭包算法

```
  //有向非循环图的传递闭包算法
void Graph_List::Tranclo( )
{
  int n=graphsize;
  int *BREACH=new int[n];
  vector< vector<int> >    reach; //数组reach的元素是集合类型
  int j=0;
  for (int i=0;i<n;i++)//初始化
  {
        BREACH[i]=0;
  }
  for (int i=n-1;i>=0;i--){//计算传递闭包
        vector<int> arr;
        BREACH[i]=1;
        arr.push_back(i);//初始化reach[i]={i}
        Edge *p = Head [ i ].adjacent;
        while(p != NULL){
              j=p-> VerAdj;
              if(BREACH[j]==0){
                    vector<int> r;
                    r=reach[n-j-1];//找到邻接点所存储的位置
                    for(int b=0;b<r.size();b++){
                          if(BREACH[r[b]]==0){
                                arr.push_back(r[b]);
                                BREACH[r[b]]=1;
                          }
                    }
              }
              p = p->link;
        }
        //记录顶点清零
        for(int c=0;c<arr.size();c++) {
              BREACH[arr[c]]=0;
        }

        reach.push_back(arr);
  }
  //打印可及顶点的集合
  for(int i=0;i<reach.size();i++){
        vector<int>    tem=reach[i];
        cout <<n - i-1 << "  的可及顶点集合:" ;
        for(int a=0;a<tem.size();a++){
              cout <<tem[a]<<",";
        }
        cout<<"\n";
  }
}
```