```
//从起始顶点 v 开始深度优先遍历图的迭代算法
void Graph_List :: DDepthFirstSearch( const int v )
{
    int* visited = new int[graphsize] ; // 为辅助数组申请空间
    for( int k = 0 ; k < graphsize ; k++ ) // 辅助数组初始化
    visited[k] = 0 ;
    Stack S ;                    // 有关 Stack 类的定义参见第 3 章 顺序表
    S.Push( v ) ; // 将起始顶点压入栈
    int w;
    while( ! S.StackEmpty( ) ) // 当栈不空时
     {
          w = S.Pop( ) ;      // 弹出一个顶点 w 且可断定该顶点没有被访问过
          if(visited[w]==0)
           {
              cout << w << " " ;          // 输出该顶点
              visited[w] = 1 ; // 访问顶点 w
              Edge *p = Head [ w ].adjacent ;
              while( p != NULL )     // 当 w 没有邻接顶点时，k 值为-1。
              {
                   if( visited[p->VerAdj] == 0 ) // 若 k 未被访问过，将 k 压入栈。
                        S.Push( p->VerAdj ) ;
                   p=p->link;
               }
            }
         }
         delete[ ] visited;  // 释放辅助数组空间
     }
```