《大话数据结构 溢彩加强版》勘误

加粗大字体必需修改，小字体属于是无伤大雅的小错，可以不修改。

带有**的是第二、三次印刷后错误。

带有****的是第四次印刷后错误。

带有*****的是第五次印刷后错误。

带有******的是第六次印刷后错误。

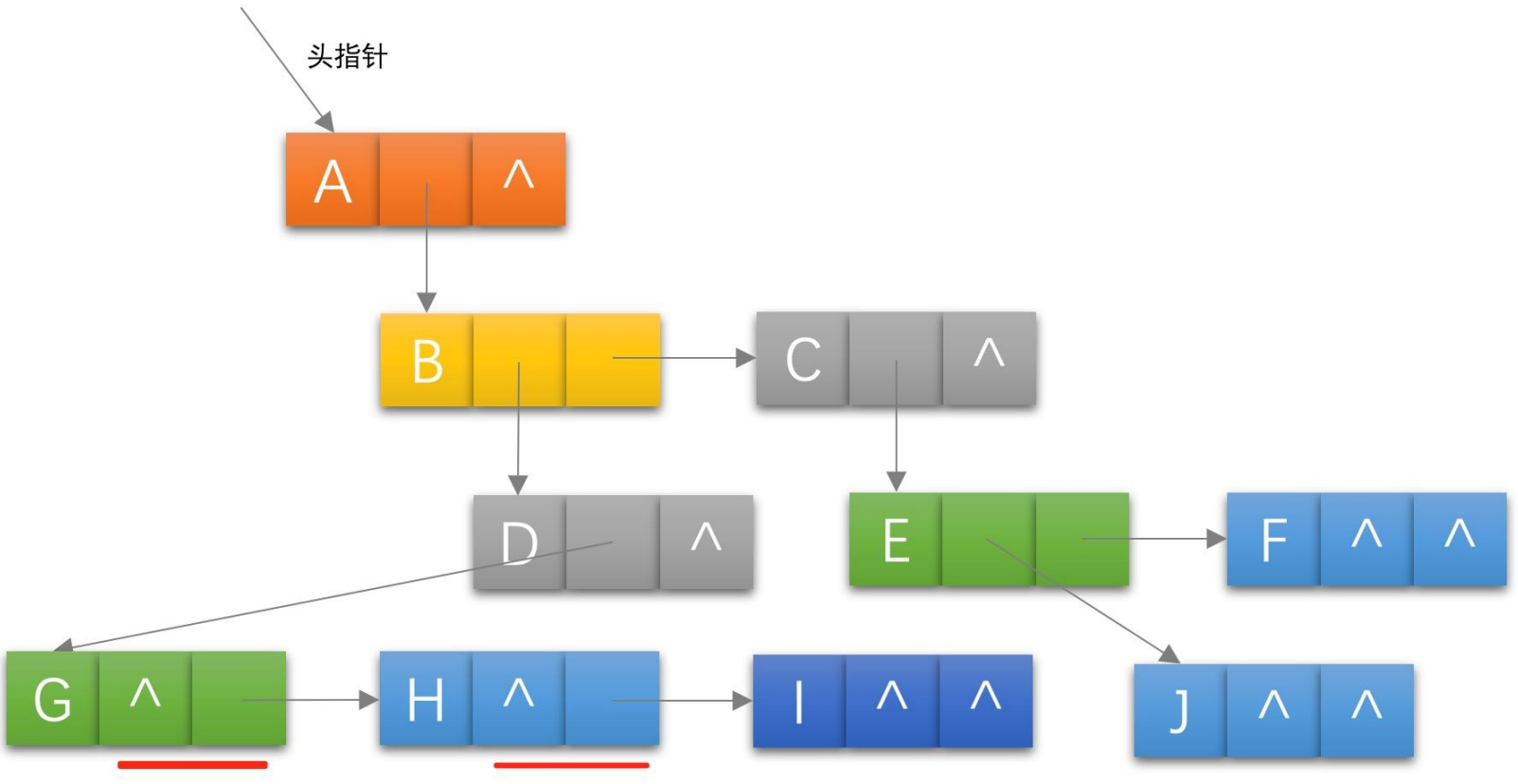
带有*******的是第七次印刷后错误。

带有********的是第八次印刷后错误。

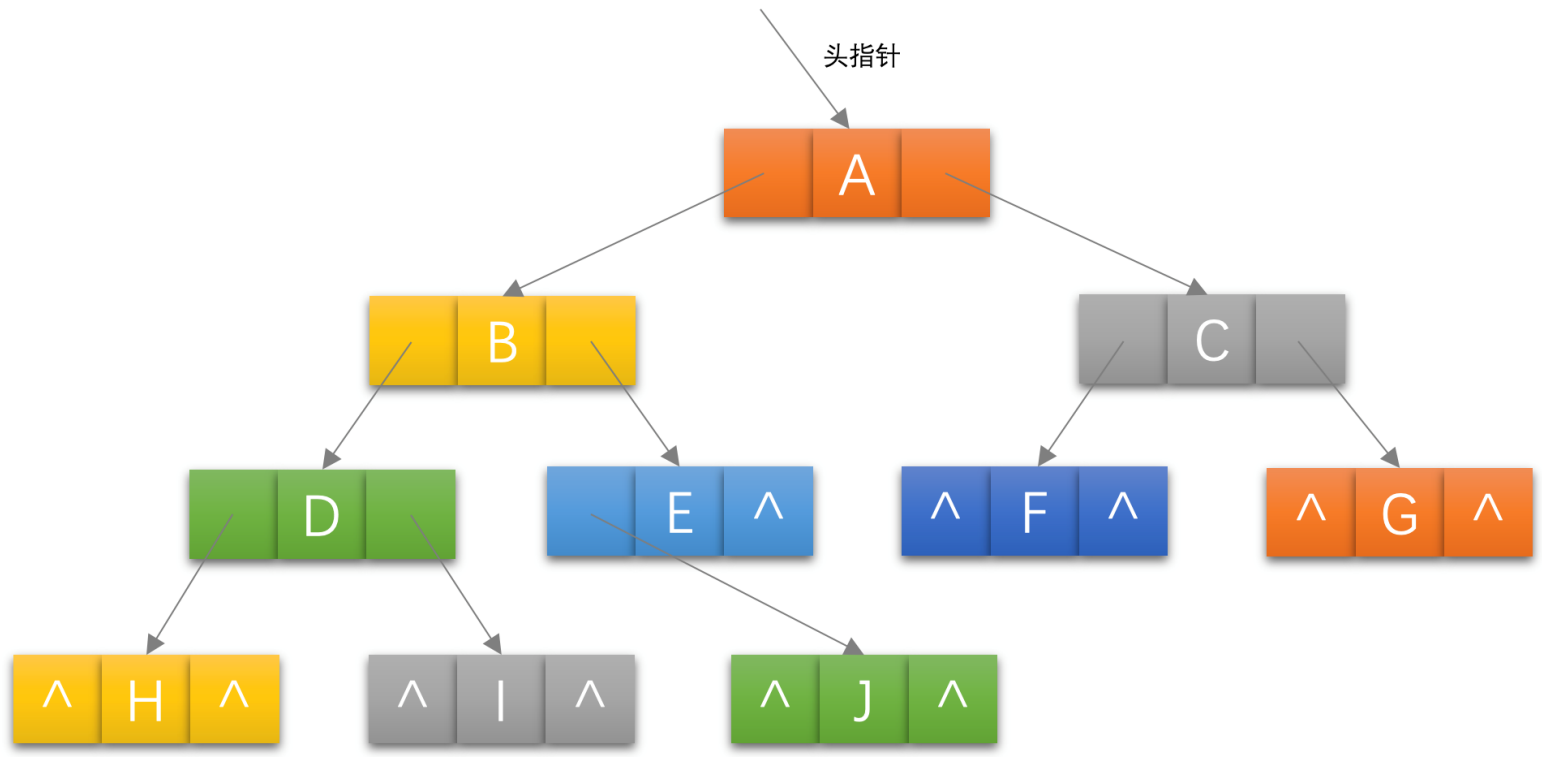
给您添麻烦了，真心抱歉！

第一次印刷后的所有勘误（涵盖二、三、四次印刷）

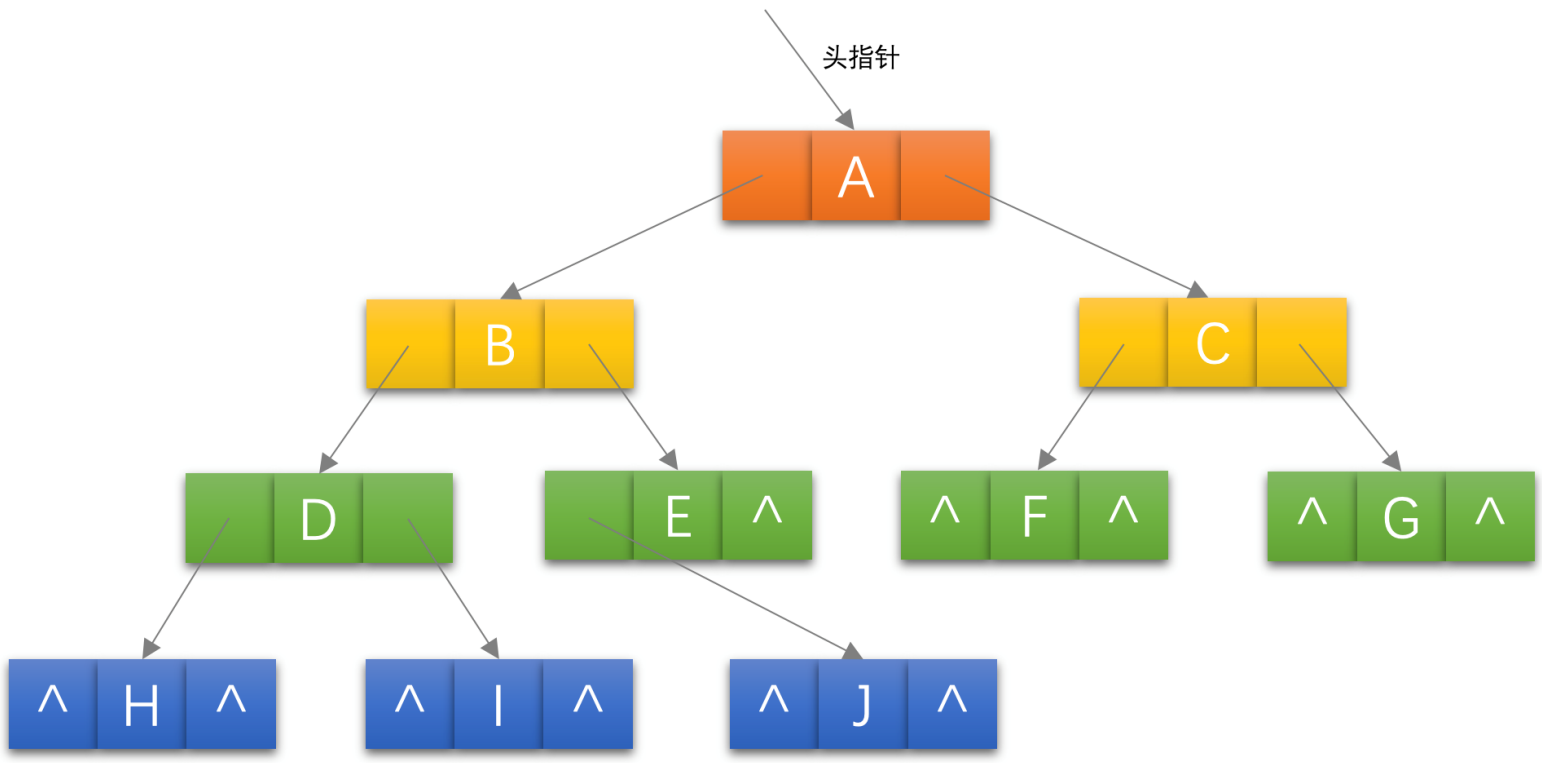
- 目录，3.4.3，“数据长度与线性表长度的区别”，将“数据”改为“数组”
- **P33，第二行，“平均的查找时间为 n/2”，将“n/2”改为“(n+1)/2”
- P44，3.4.3 “数据长度与线性表长度的区别”，将“数据”改为“数组”
- P46，中间段，“注意，这里我们是把指针*e 的值给修改成 L->data[i-1]”，其中“L->data[i-1]”改成“L.data[i-1]”
- *******P49，第三段，“由于元素插入到第 i 个位置，或删除第 i 个元素…… ”，插入“需要移动 n-i+1 个元素，”
- P63，最下面一段第一行，“乙的游标 2”，把“游”改为“下”
- **P67，图中文字，“失去了链式存储结构随机存取特性”，”链式“改成“顺序”。**
- P70，上图代码第三行注释，“赋值给 reaA->next”其中少了一个字母 r，应该是“赋值给 rearA->next”
- P79，4.4.1，第一段第二行，“线性表是用数组来实现的，”，其中“线性表”改成“顺序表”。
- *******P107，中间表格第三行，“串中元素仅由一个字符组成”，其中“一个”两字去掉。
- **P112，第一行，“一开始就区配成功”，改成“匹配”。
- *****P132，第 2 个表，第 6 行最后一列，也就是 4、E、2、-1 这一行，将-1 改成 9。
- **P136，第一行，“或者找某个结点的兄弟”改为“或者找某个结点的某个孩子的兄弟”。增加“某个孩子的”。**
- **P137，改图中 G 结点与 H 结点的标记与连接错误**



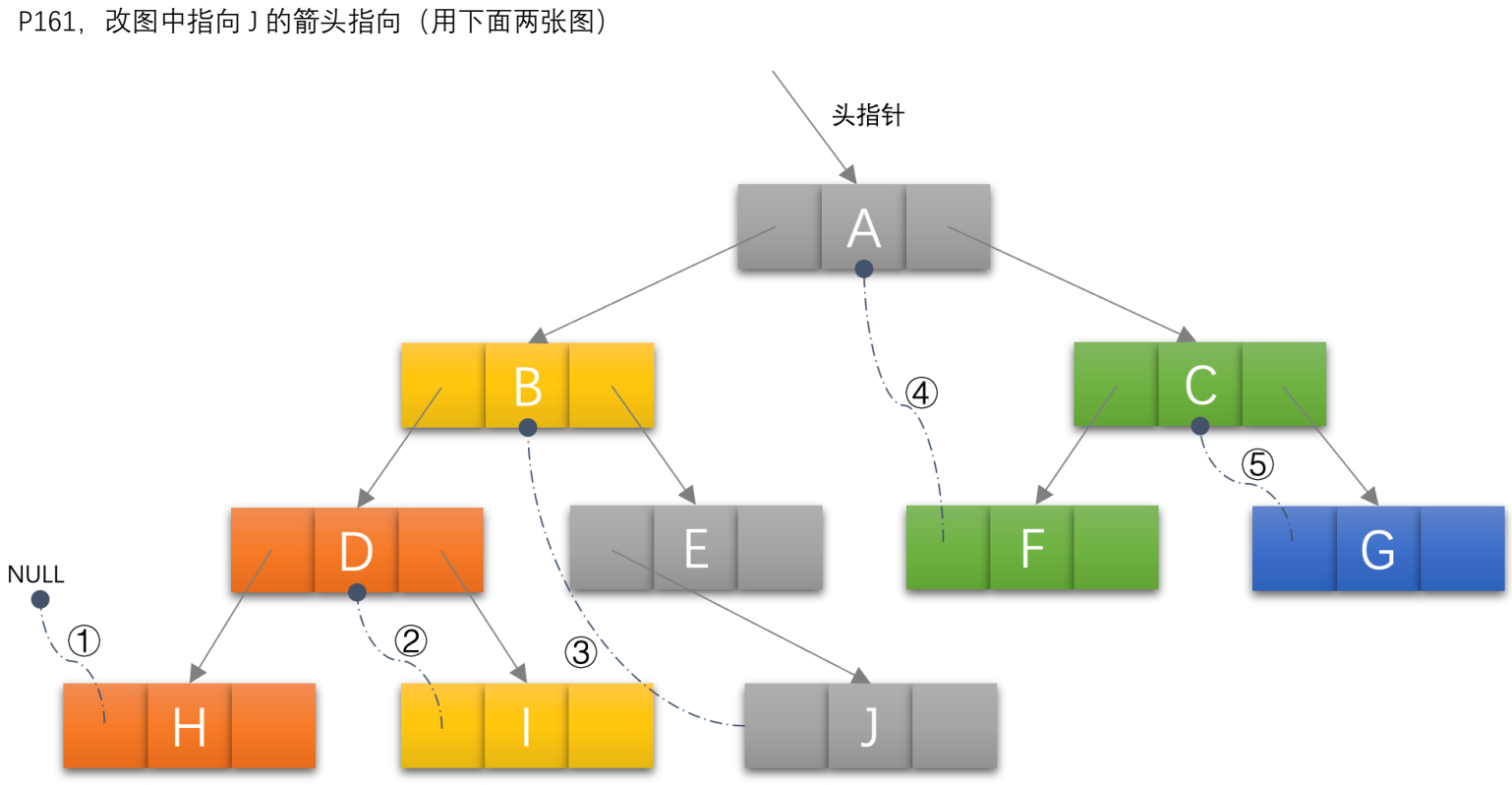
- P142，第五段，“（3）倒数两层”，改为“（3）倒数第二层”
- **P143，6.6.2 上面一段和下面最后一段，两处“通过数据归纳法的论证，”，改为“数学”。
- P145，6.7.1 最下面图，G 的颜色改为绿色。
- P147，改图中指向 J 的箭头指向

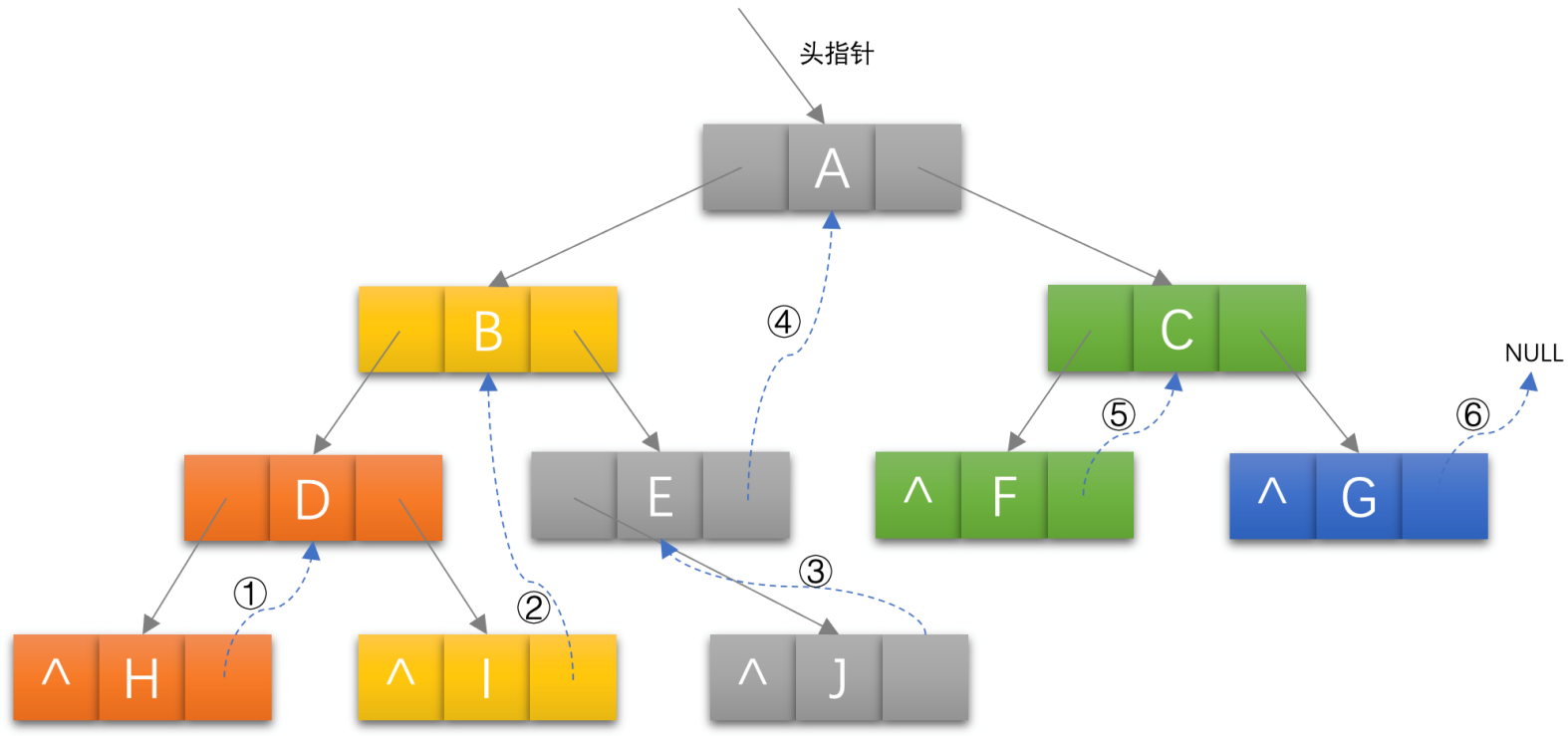


- ******P154，最后一段，“所有返回。打印结点 H，函数执行完毕，返回。”去掉那个“，”。
- P160，改图中指向 J 的箭头指向，用下图



- P161，改图中指向 J 的箭头指向（用下面两张图）





- **P161，第二段，“空心箭头实线为前驱，虚线黑箭头为后继”改为“黑点虚线为前驱，蓝箭头虚线为后继”**
- P162，代码注释去掉两个等号，改为「Link=0 表示指向左右孩子指针」和「Thread=1 表示指向前驱或后继的线索」
- ****P170，倒数第二段。“后序遍历：是先访问森林中第一棵树，后根遍历的方式遍历每棵子树，然后再访问根结点，再依次同样方式遍历除去第一棵树的剩余树构成的森林。比如下图三棵树的森林，后序遍历序列的结果就是 BCDAFEJHIG。可如果我们对下图的二叉树进行分析就会发现，森林的前序遍历和二叉树的前序遍历结果相同，森林的后序遍历和二叉树的中序遍历结果相同。”将“后序”或“后根”改成“中序”。
- ****P186，第二段第二行，v=vi,0,vi,1,⋯,vi,m=v’其中 0、1、m 要改成小下标 v=$v_{i,0}$,$v_{i,1}$,⋯,$v_{i,m}$=v’
- **P188， 上方图两图的 A 与 B 箭头画反**

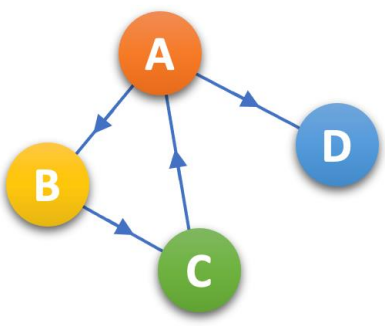


图1

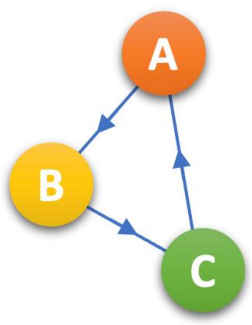
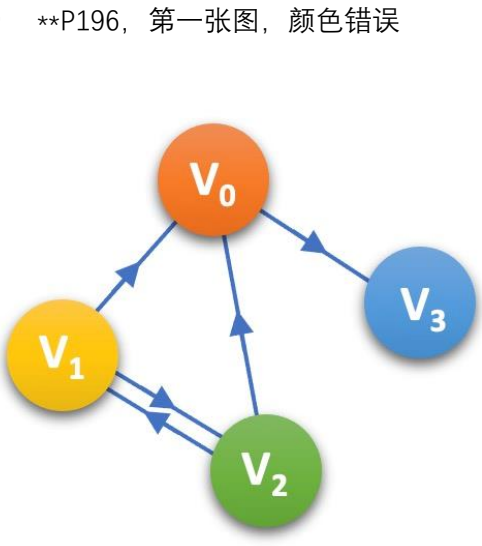
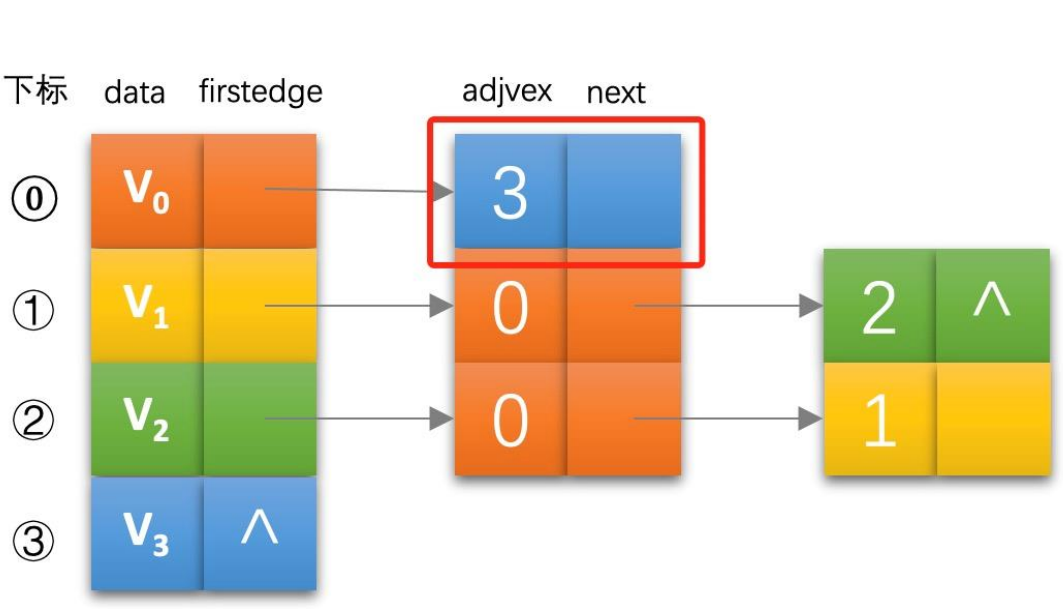


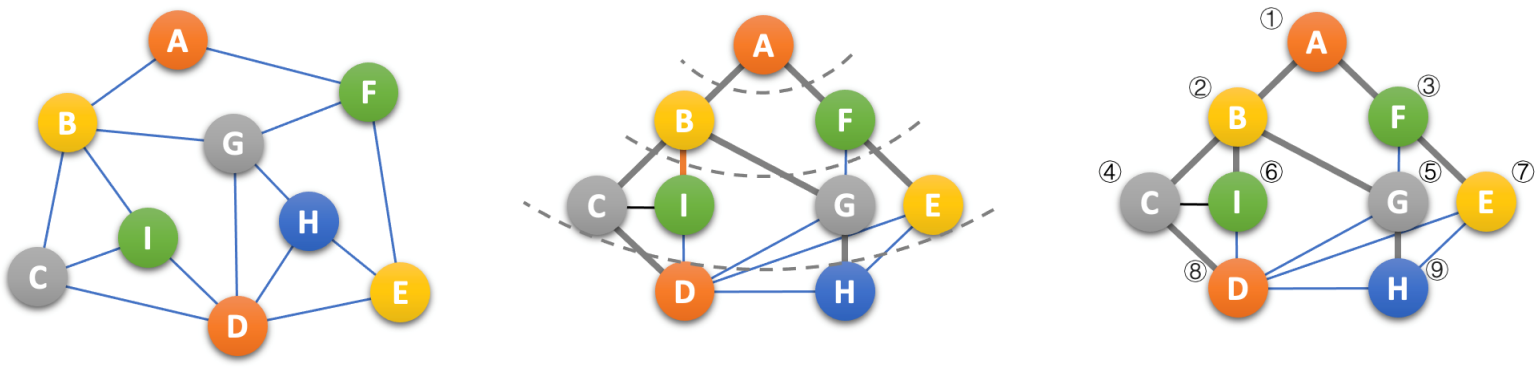
图2

- **P196，第一张图，颜色错误

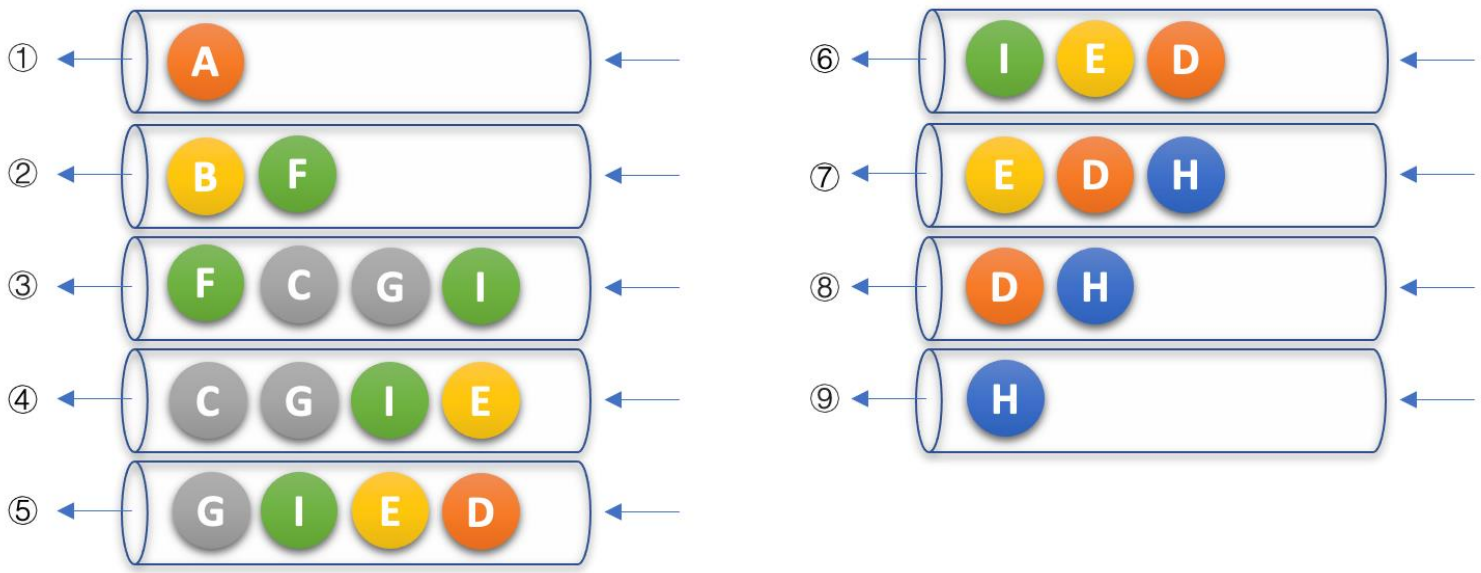




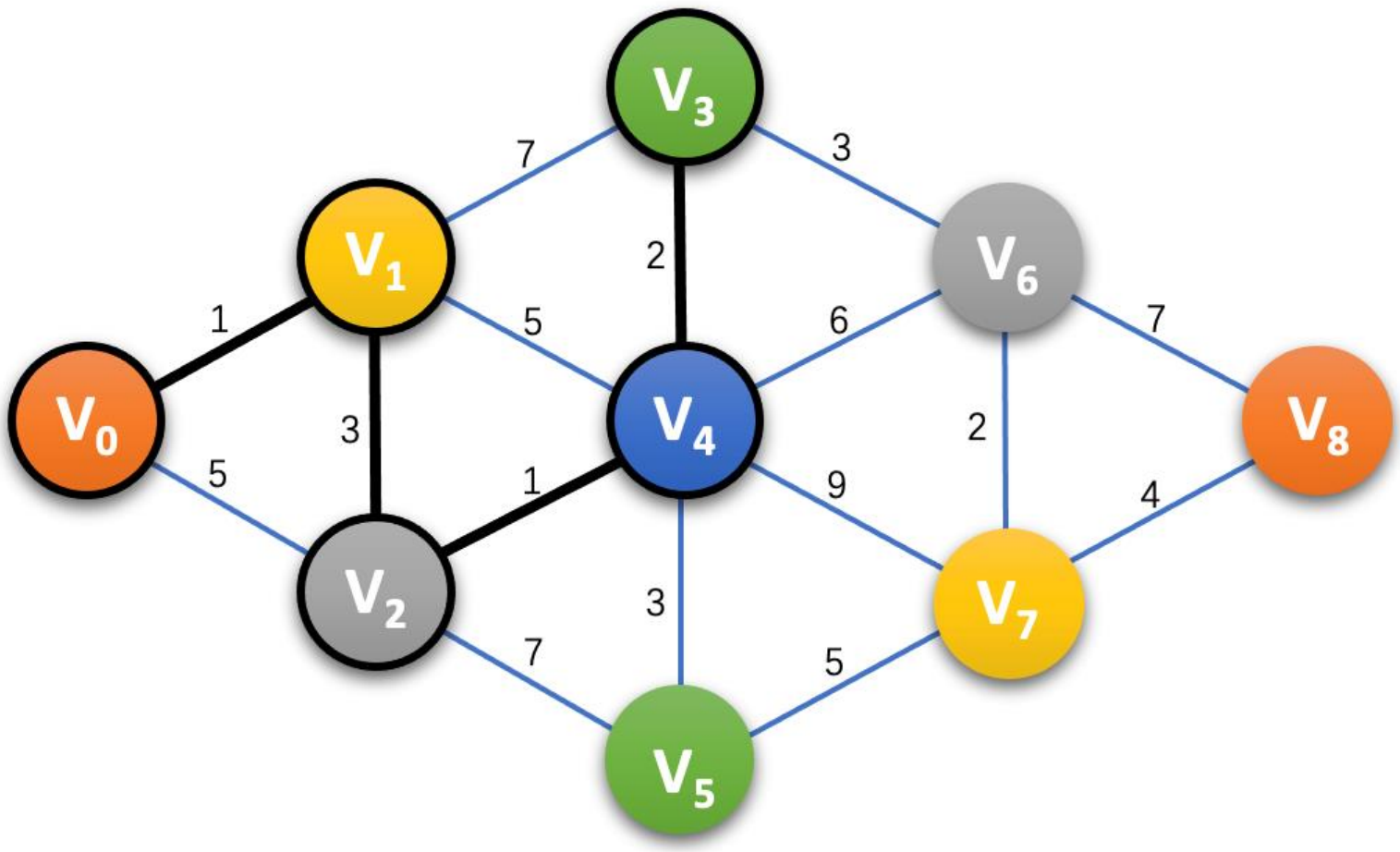
- **P206，下图第三图的错误，需 5 与 6 交换**



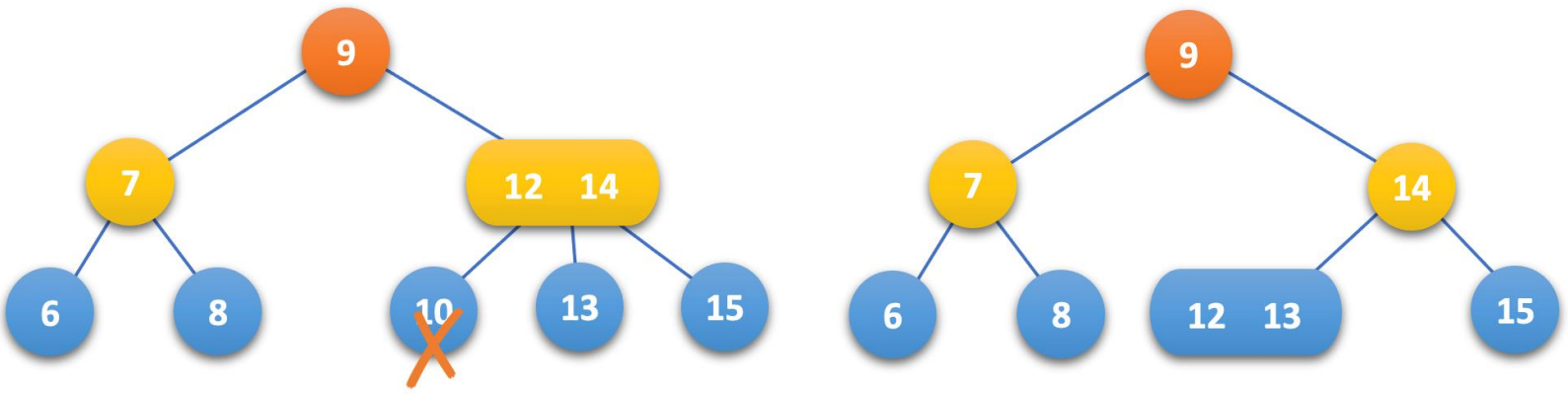
P206，下图错误，主要是 I 与 G 字母交换，原因在于字母顺序 G 在 I 的前面，循环时不可能在后面



- P221，图错误，应该更改为



- **P239，代码第 8 行注释，/* 事件最早发生时间数组 */ 其中“早”改为“晚”。
- **P240，第 2 行，“{27，2727，27” 其中“2727”中间应该有一个逗号，改为“{27，27，27，27”。
- **P247，下方图，⑤次关键字”应该指向-0.11 处
- *******P253，倒数第四行：如果再让你查单调“zoo”。“单调”应该是改为“单词”
- ****P256，第四行，“后面的 a[11]、a[12]均未赋值”中的“ 、a[12]均”删除。第五行，“a[11]=a[12]=a[10]=99”中的“=a[12]”删除。
- **P257，第二行，“新范围是第 m+1 个到第 high 个”，“m”改成“mid”
- *******P274，倒数第二行：“二者差大于了绝对值 1。”应该是改为“二者差的绝对值大于 1。”
- **P288,最下方图错，换成下图**



- P329，第五行，2^{t-k+1}改为 2^{t-k+1}，其实中-写成了_
- ********P347，第二幅代码图，去掉一个空行。主要是为了让下面讲解时行号正确。

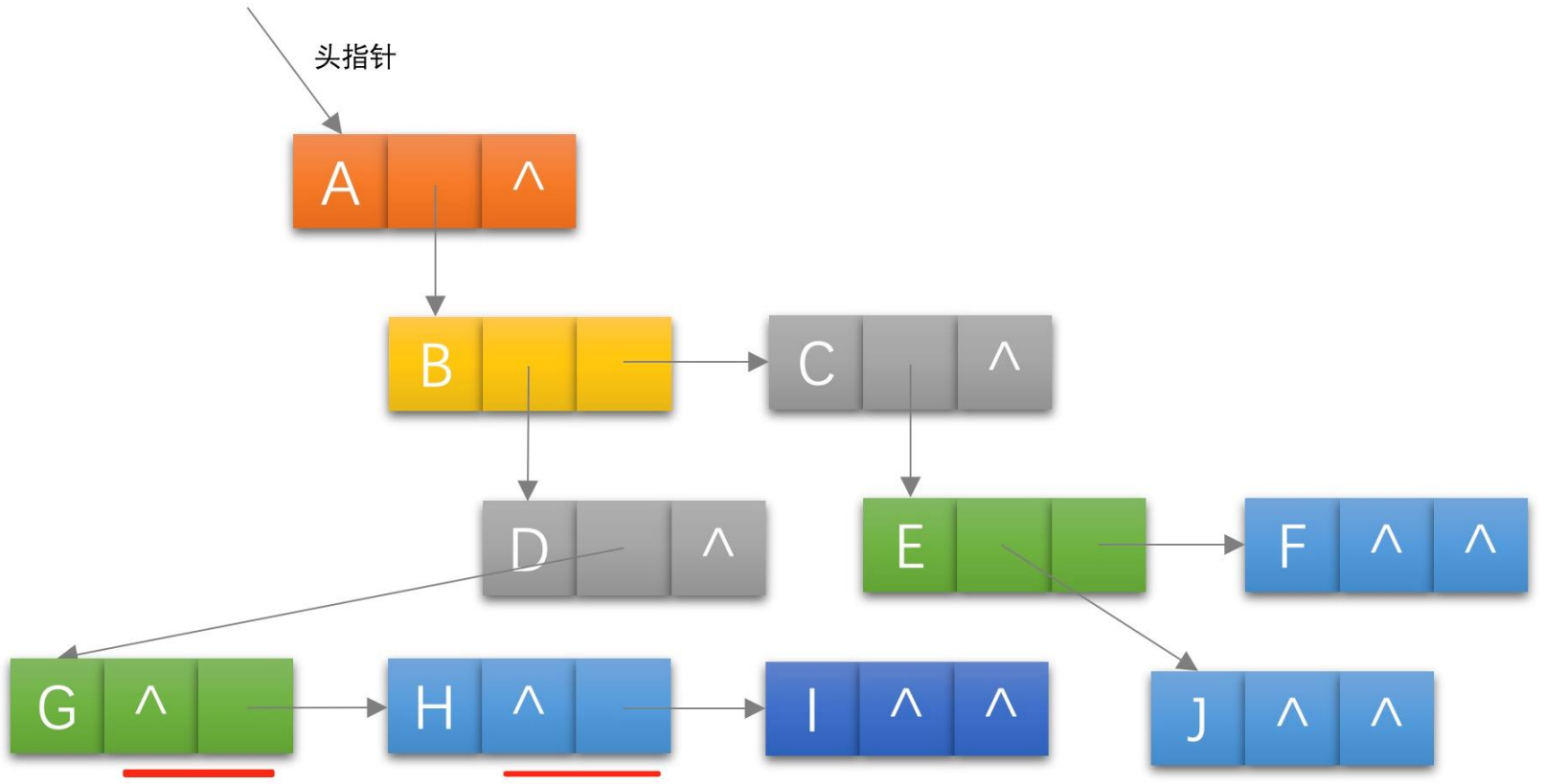
```
int Partition(SqList *L,int low,int high)
{/* 交换顺序表L中子表的记录，使枢轴记录到位，并返回其所在位置，此时在它之前(后)均不大(小)于它。*/
    int pivotkey;
        删除
    pivotkey=L->r[low];      /* 用子表的第一个记录作枢轴记录 */
    while(low<high)          /* 从表的两端交替地向中间扫描 */
    {
         while(low<high&&L->r[high]>=pivotkey)
            high--;
         swap(L,low,high);  /* 将比枢轴记录小的记录交换到低端 */
         while(low<high&&L->r[low]<=pivotkey)
            low++;
         swap(L,low,high);  /* 将比枢轴记录大的记录交换到高端 */
    }
    return low;              /* 返回枢轴所在位置 */
}
```

改成下面样式

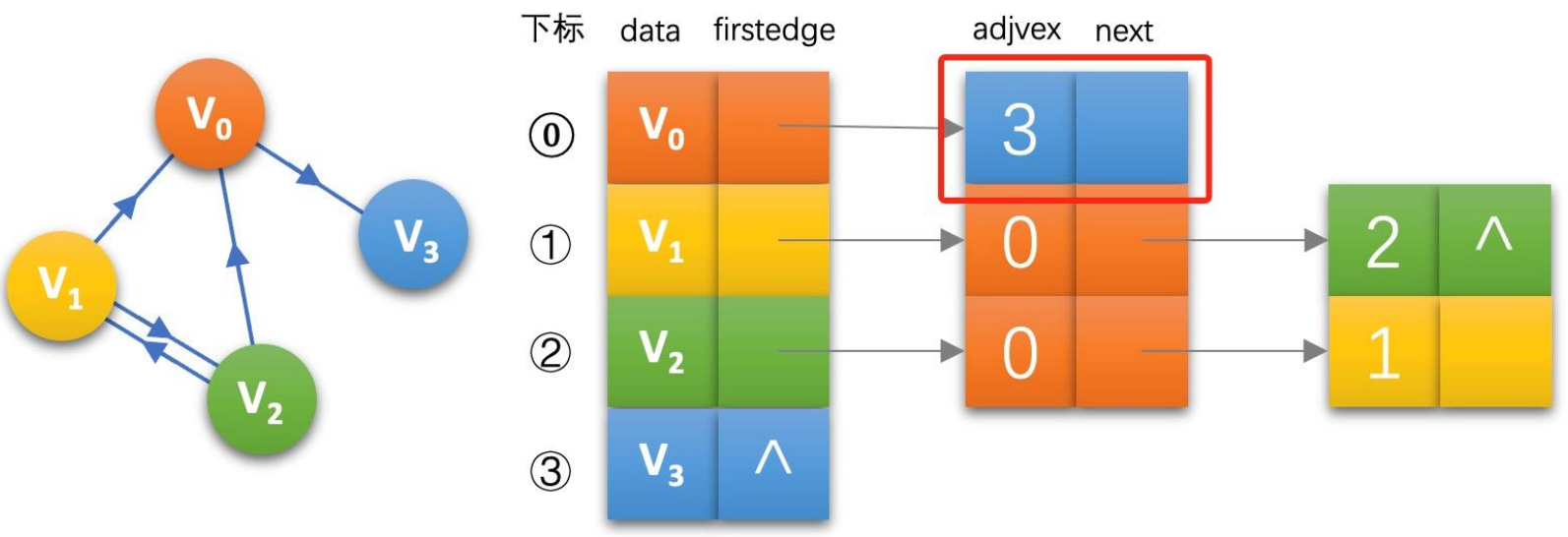
```
int Partition(SqList *L,int low,int high)
{/* 交换顺序表L中子表的记录，使枢轴记录到位，并返回其所在位置，此时在它之前(后)均不大(小)于它。*/
    int pivotkey;
    pivotkey=L->r[low];      /* 用子表的第一个记录作枢轴记录 */
    while(low<high)          /* 从表的两端交替地向中间扫描 */
    {
         while(low<high&&L->r[high]>=pivotkey)
            high--;
         swap(L,low,high);  /* 将比枢轴记录小的记录交换到低端 */
         while(low<high&&L->r[low]<=pivotkey)
            low++;
         swap(L,low,high);  /* 将比枢轴记录大的记录交换到高端 */
    }
    return low;              /* 返回枢轴所在位置 */
}
```

第二、三次印刷后勘误

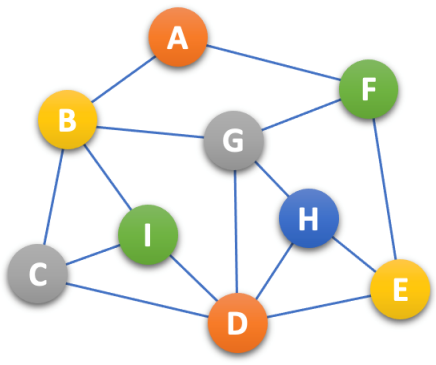
- **P33，第二行，“平均的查找时间为 n/2”，将“n/2”改为“(n+1)/2”
- *******P49，第三段，“由于元素插入到第 i 个位置，或删除第 i 个元素…… ”，插入“需要移动 n-i+1 个元素，”
- ****P70，上图代码第三行注释，“赋值给 reaA->next”其中少了一个字母 r，应该是“赋值给 rearA->next”
- *******P107，中间表格第三行，“串中元素仅由一个字符组成”，其中“一个”两字去掉。
- **P112，第一行，“一开始就区配成功”，改成“匹配”。
- *****P132，第 2 个表，第 6 行最后一列，也就是 4、E、2、-1 这一行，将-1 改成 9。
- **P137，改图中 G 结点与 H 结点的标记与连接错误

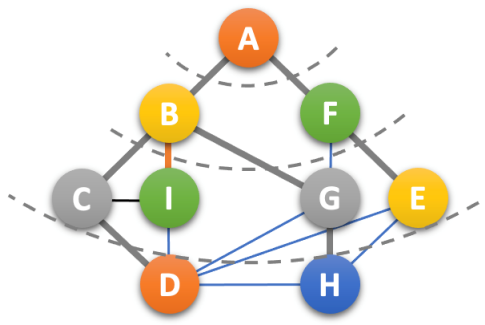


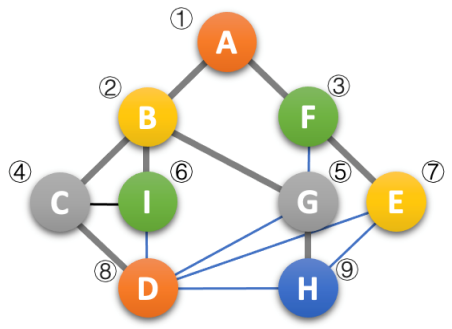
- **P143，6.6.2 上面一段和下面最后一段，两处“通过数据归纳法的论证，”，改为“数学”。
- ******P154，最后一段，“所有返回。打印结点 H，函数执行完毕，返回。”去掉那个“，”。
- ****P170，倒数第二段。“后序遍历：是先访问森林中第一棵树，后根遍历的方式遍历每棵子树，然后再访问根结点，再依次同样方式遍历除去第一棵树的剩余树构成的森林。比如下图三棵树的森林，后序遍历序列的结果就是 BCDAFEJHIG。可如果我们对下图的二叉树进行分析就会发现，森林的前序遍历和二叉树的前序遍历结果相同，森林的后序遍历和二叉树的中序遍历结果相同。”将“后序”或“后根”改成“中序”。
- ****P186，第二段第二行，v=vi,0,vi,1,⋯,vi,m=v’其中 0、1、m 要改成小下标 v=$v_{i,0}$,$v_{i,1}$,⋯,$v_{i,m}$=v’
- **P196，第一张图，颜色错误



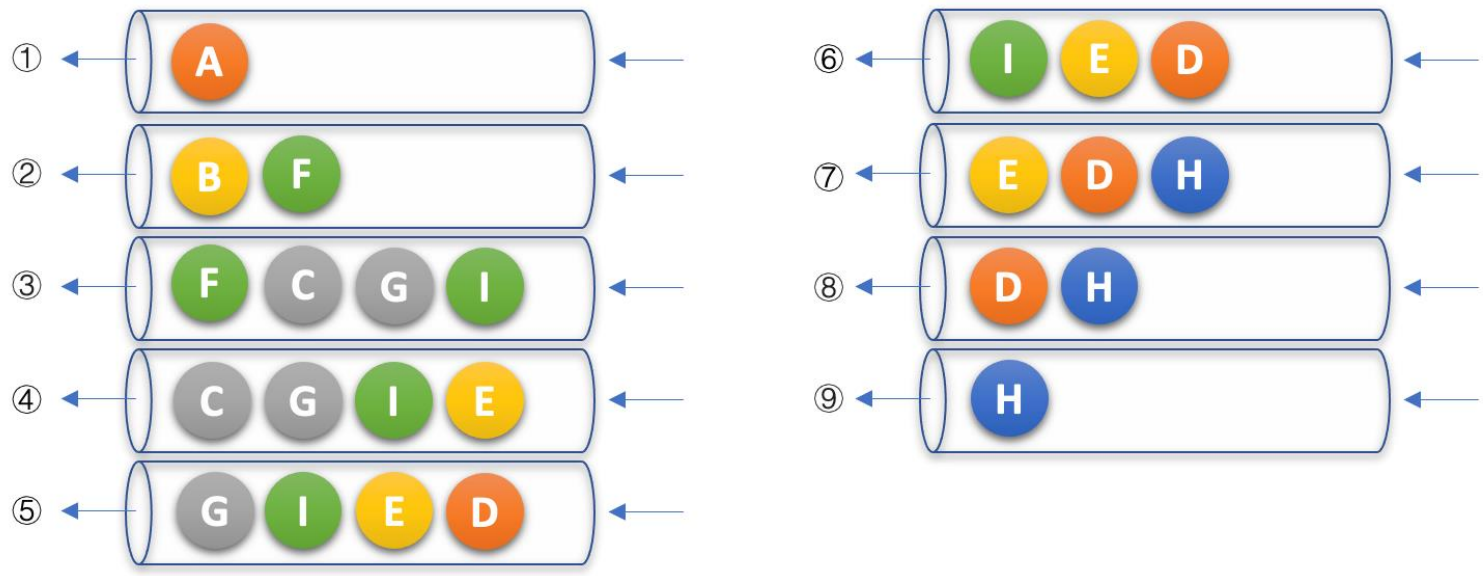
- **P206，下图第三图的错误，将编号 5 与编号 6 交换







- P206，下图错误，主要是 I 与 G 字母交换，原因在于字母顺序 G 在 I 的前面，按字母顺序循环时不可能到后面去



- **P239，代码第 8 行注释，/* 事件最早发生时间截图 */ 其中“早”改为“晚”。
- **P240，第 2 行，“{27，2727，27” 其中“2727”中间应该有一个逗号，改为“{27，27，27，27”。
- **P247，下方图，⑤次关键字”应该指向-0.11 处
- *******P253，倒数第四行：如果再让你查单调“zoo”。“单调”应该是改为“单词”
- ****P256，第四行，“后面的 a[11]、a[12]均未赋值”中的“ 、a[12]均”删除。第五行，“a[11]=a[12]=a[10]=99”中的“=a[12]”删除。
- **P257，第二行，“新范围是第 m+1 个到第 high 个”，“m”改成“mid”
- *******P274，倒数第二行：“二者差大于了绝对值 1。”应该是改为“二者差的绝对值大于 1。”
- ********P347，第二幅代码图，去掉一个空行。主要是为了让下面讲解时行号正确。

```
1  int Partition(SqList *L,int low,int high)
2  {/* 交换顺序表L中子表的记录，使枢轴记录到位，并返回其所在位置，此时在它之前(后)均不大(小)于它。*/
3      int pivotkey;
4          删除
5      pivotkey=L->r[low];      /* 用子表的第一个记录作枢轴记录 */
6      while(low<high)          /* 从表的两端交替地向中间扫描 */
7      {
8           while(low<high&&L->r[high]>=pivotkey)
9              high--;
10          swap(L,low,high);   /* 将比枢轴记录小的记录交换到低端 */
11          while(low<high&&L->r[low]<=pivotkey)
12             low++;
13          swap(L,low,high);   /* 将比枢轴记录大的记录交换到高端 */
14     }
15     return low;              /* 返回枢轴所在位置 */
16 }
```

改成下面样式

```
1  int Partition(SqList *L,int low,int high)
2  {/* 交换顺序表L中子表的记录，使枢轴记录到位，并返回其所在位置，此时在它之前(后)均不大(小)于它。*/
3      int pivotkey;
4      pivotkey=L->r[low];      /* 用子表的第一个记录作枢轴记录 */
5      while(low<high)          /* 从表的两端交替地向中间扫描 */
6      {
7           while(low<high&&L->r[high]>=pivotkey)
8              high--;
9           swap(L,low,high);   /* 将比枢轴记录小的记录交换到低端 */
10          while(low<high&&L->r[low]<=pivotkey)
11             low++;
12          swap(L,low,high);   /* 将比枢轴记录大的记录交换到高端 */
13     }
14     return low;              /* 返回枢轴所在位置 */
15 }
```

第四次印刷

- *******P49，第三段，“由于元素插入到第 i 个位置，或删除第 i 个元素…… ”，插入“需要移动 n-i+1 个元素，”
- ****P70，上图代码第三行注释，“赋值给 reaA->next”其中少了一个字母 r，应该是“赋值给 rearA->next”
- *******P107，中间表格第三行，“串中元素仅由一个字符组成”，其中“一个”两字去掉。
- *****P132，第 2 个表，第 6 行最后一列，也就是 4、E、2、-1 这一行，将-1 改成 9。
- ******P154，最后一段，“所有返回。打印结点 H，函数执行完毕，返回。”去掉那个“，”。
- ****P170，倒数第二段。“后序遍历：是先访问森林中第一棵树，后根遍历的方式遍历每棵子树，然后再访问根结点，再依次同样方式遍历除去第一棵树的剩余树构成的森林。比如下图三棵树的森林，后序遍历序列的结果就是 BCDAFEJHIG。可如果我们对下图的二叉树进行分析就会发现，森林的前序遍历和二叉树的前序遍历结果相同，森林的后序遍历和二叉树的中序遍历结果相同。”将“后序”或“后根”改成“中序”。
- ****P186，第二段第二行，v=vi,0,vi,1,…,vi,m=v’其中 0、1、m 要改成小下标 v=$v_{i,0}$,$v_{i,1}$,…,$v_{i,m}$=v’
- *******P253，倒数第四行：如果再让你查单调“zoo”。“单调”应该是改为“单词”
- ****P256，第四行，“后面的 a[11]、a[12]均未赋值”中的“ 、a[12]均”删除。第五行，“a[11]=a[12]=a[10]=99”中的“=a[12]”删除。
- *******P274，倒数第二行：“二者差大于了绝对值 1。”应该是改为“二者差的绝对值大于 1。”
- ********P347，第二幅代码图，去掉一个空行。主要是为了让下面讲解时行号正确。

```
int Partition(SqList *L,int low,int high)
{/* 交换顺序表L中子表的记录，使枢轴记录到位，并返回其所在位置，此时在它之前(后)均不大(小)于它。*/
    int pivotkey;
        删除
    pivotkey=L->r[low];      /* 用子表的第一个记录作枢轴记录 */
    while(low<high)          /* 从表的两端交替地向中间扫描 */
    {
         while(low<high&&L->r[high]>=pivotkey)
            high--;
         swap(L,low,high);   /* 将比枢轴记录小的记录交换到低端 */
         while(low<high&&L->r[low]<=pivotkey)
            low++;
         swap(L,low,high);   /* 将比枢轴记录大的记录交换到高端 */
    }
    return low;              /* 返回枢轴所在位置 */
}
```

改成下面样式

```
int Partition(SqList *L,int low,int high)
{/* 交换顺序表L中子表的记录，使枢轴记录到位，并返回其所在位置，此时在它之前(后)均不大(小)于它。*/
    int pivotkey;
    pivotkey=L->r[low];      /* 用子表的第一个记录作枢轴记录 */
    while(low<high)          /* 从表的两端交替地向中间扫描 */
    {
         while(low<high&&L->r[high]>=pivotkey)
            high--;
         swap(L,low,high);   /* 将比枢轴记录小的记录交换到低端 */
         while(low<high&&L->r[low]<=pivotkey)
            low++;
         swap(L,low,high);   /* 将比枢轴记录大的记录交换到高端 */
    }
    return low;              /* 返回枢轴所在位置 */
}
```

第五次印刷

- *******P49，第三段，“由于元素插入到第 i 个位置，或删除第 i 个元素…… ”，插入“需要移动 n-i+1 个元素，”
- *******P107，中间表格第三行，“串中元素仅由一个字符组成”，其中“一个”两字去掉。
- *****P132，第 2 个表，第 6 行最后一列，也就是 4、E、2、-1 这一行，将-1 改成 9。
- ******P154，最后一段，“所有返回。打印结点 H，函数执行完毕，返回。”去掉那个“，”。
- *******P253，倒数第四行：如果再让你查单调“zoo”。“单调”应该是改为“单词”
- *******P274，倒数第二行：“二者差大于了绝对值 1。”应该是改为“二者差的绝对值大于 1。”
- ********P347，第二幅代码图，去掉一个空行。主要是为了让下面讲解时行号正确。

```
int Partition(SqList *L,int low,int high)
{/* 交换顺序表L中子表的记录，使枢轴记录到位，并返回其所在位置，此时在它之前(后)均不大(小)于它。*/
    int pivotkey;
        删除
    pivotkey=L->r[low];      /* 用子表的第一个记录作枢轴记录 */
    while(low<high)          /* 从表的两端交替地向中间扫描 */
    {
         while(low<high&&L->r[high]>=pivotkey)
            high--;
         swap(L,low,high);   /* 将比枢轴记录小的记录交换到低端 */
         while(low<high&&L->r[low]<=pivotkey)
            low++;
         swap(L,low,high);   /* 将比枢轴记录大的记录交换到高端 */
    }
    return low;              /* 返回枢轴所在位置 */
}
```

改成下面样式

```c
int Partition(SqList *L,int low,int high)
{/* 交换顺序表L中子表的记录，使枢轴记录到位，并返回其所在位置，此时在它之前(后)均不大(小)于它。*/
    int pivotkey;
    pivotkey=L->r[low];      /* 用子表的第一个记录作枢轴记录 */
    while(low<high)          /* 从表的两端交替地向中间扫描 */
    {
         while(low<high&&L->r[high]>=pivotkey)
            high--;
         swap(L,low,high);  /* 将比枢轴记录小的记录交换到低端 */
         while(low<high&&L->r[low]<=pivotkey)
            low++;
         swap(L,low,high);  /* 将比枢轴记录大的记录交换到高端 */
    }
    return low;              /* 返回枢轴所在位置 */
}
```

第六次印刷

- *******P49，第三段，“由于元素插入到第 i 个位置，或删除第 i 个元素…… ”，插入“需要移动 n-i+1 个元素，”
- *******P107，中间表格第三行，“串中元素仅由一个字符组成”，其中“一个”两字去掉。
- ******P154，最后一段，“所有返回。打印结点 H，函数执行完毕，返回。”去掉那个“，”。
- *******P253，倒数第四行：如果再让你查单调“zoo”。“单调”应该是改为“单词”
- *******P274，倒数第二行：“二者差大于了绝对值 1。”应该是改为“二者差的绝对值大于 1。”
- ********P347，第二幅代码图，去掉一个空行。主要是为了让下面讲解时行号正确。

```c
int Partition(SqList *L,int low,int high)
{/* 交换顺序表L中子表的记录，使枢轴记录到位，并返回其所在位置，此时在它之前(后)均不大(小)于它。*/
    int pivotkey;
        删除
    pivotkey=L->r[low];      /* 用子表的第一个记录作枢轴记录 */
    while(low<high)          /* 从表的两端交替地向中间扫描 */
    {
         while(low<high&&L->r[high]>=pivotkey)
            high--;
         swap(L,low,high);  /* 将比枢轴记录小的记录交换到低端 */
         while(low<high&&L->r[low]<=pivotkey)
            low++;
         swap(L,low,high);  /* 将比枢轴记录大的记录交换到高端 */
    }
    return low;              /* 返回枢轴所在位置 */
}
```

改成下面样式

```c
int Partition(SqList *L,int low,int high)
{/* 交换顺序表L中子表的记录，使枢轴记录到位，并返回其所在位置，此时在它之前(后)均不大(小)于它。*/
    int pivotkey;
    pivotkey=L->r[low];      /* 用子表的第一个记录作枢轴记录 */
    while(low<high)          /* 从表的两端交替地向中间扫描 */
    {
         while(low<high&&L->r[high]>=pivotkey)
            high--;
         swap(L,low,high);  /* 将比枢轴记录小的记录交换到低端 */
         while(low<high&&L->r[low]<=pivotkey)
            low++;
         swap(L,low,high);  /* 将比枢轴记录大的记录交换到高端 */
    }
    return low;              /* 返回枢轴所在位置 */
}
```

第七次印刷

- *******P49，第三段，“由于元素插入到第 i 个位置，或删除第 i 个元素…… ”，插入“需要移动 n-i+1 个元素，”
- *******P107，中间表格第三行，“串中元素仅由一个字符组成”，其中“一个”两字去掉。
- *******P253，倒数第四行：如果再让你查单调“zoo”。“单调”应该是改为“单词”
- *******P274，倒数第二行：“二者差大于了绝对值 1。”应该是改为“二者差的绝对值大于 1。”
- ********P347，第二幅代码图，去掉一个空行。主要是为了让下面讲解时行号正确。

```c
int Partition(SqList *L,int low,int high)
{/* 交换顺序表L中子表的记录，使枢轴记录到位，并返回其所在位置，此时在它之前(后)均不大(小)于它。*/
    int pivotkey;
        删除
    pivotkey=L->r[low];      /* 用子表的第一个记录作枢轴记录 */
    while(low<high)          /* 从表的两端交替地向中间扫描 */
    {
         while(low<high&&L->r[high]>=pivotkey)
            high--;
         swap(L,low,high);   /* 将比枢轴记录小的记录交换到低端 */
         while(low<high&&L->r[low]<=pivotkey)
            low++;
         swap(L,low,high);   /* 将比枢轴记录大的记录交换到高端 */
    }
    return low;              /* 返回枢轴所在位置 */
}
```

改成下面样式

```c
int Partition(SqList *L,int low,int high)
{/* 交换顺序表L中子表的记录，使枢轴记录到位，并返回其所在位置，此时在它之前(后)均不大(小)于它。*/
    int pivotkey;
    pivotkey=L->r[low];      /* 用子表的第一个记录作枢轴记录 */
    while(low<high)          /* 从表的两端交替地向中间扫描 */
    {
         while(low<high&&L->r[high]>=pivotkey)
            high--;
         swap(L,low,high);   /* 将比枢轴记录小的记录交换到低端 */
         while(low<high&&L->r[low]<=pivotkey)
            low++;
         swap(L,low,high);   /* 将比枢轴记录大的记录交换到高端 */
    }
    return low;              /* 返回枢轴所在位置 */
}
```

第八次印刷

- ********P347，第二幅代码图，去掉一个空行。主要是为了让下面讲解时行号正确。

```c
int Partition(SqList *L,int low,int high)
{/* 交换顺序表L中子表的记录，使枢轴记录到位，并返回其所在位置，此时在它之前(后)均不大(小)于它。*/
    int pivotkey;
        删除
    pivotkey=L->r[low];      /* 用子表的第一个记录作枢轴记录 */
    while(low<high)          /* 从表的两端交替地向中间扫描 */
    {
         while(low<high&&L->r[high]>=pivotkey)
            high--;
         swap(L,low,high);   /* 将比枢轴记录小的记录交换到低端 */
         while(low<high&&L->r[low]<=pivotkey)
            low++;
         swap(L,low,high);   /* 将比枢轴记录大的记录交换到高端 */
    }
    return low;              /* 返回枢轴所在位置 */
}
```

改成下面样式

```c
int Partition(SqList *L,int low,int high)
{/* 交换顺序表L中子表的记录，使枢轴记录到位，并返回其所在位置，此时在它之前(后)均不大(小)于它。*/
    int pivotkey;
    pivotkey=L->r[low];      /* 用子表的第一个记录作枢轴记录 */
    while(low<high)          /* 从表的两端交替地向中间扫描 */
    {
         while(low<high&&L->r[high]>=pivotkey)
            high--;
         swap(L,low,high);   /* 将比枢轴记录小的记录交换到低端 */
         while(low<high&&L->r[low]<=pivotkey)
            low++;
         swap(L,low,high);   /* 将比枢轴记录大的记录交换到高端 */
    }
    return low;              /* 返回枢轴所在位置 */
}
```