参考答案

一、单项选择题：1~10 小题，每小题 2 分，共 20 分。

1．B 【详解】程序每执行一次，x 就乘以 5，但是 x 又是小于 n 的,所以 5 的 a 次方小于等于 n,所以时间复杂度为 O($\log_5$（n）)，满足条件是 B。

2．D 【详解】该算法需要将 n 个元素依次插入到有序单链表中，需要两重循环插入每个元素，所以选 D。

3．C 【详解】根据树的性质，设分支数为 B，结点数为 n，则 B=n-1，而 B=2*1+3*2+4*3=20，n=21，所以选 C。

4．A 【详解】根据森林与二叉树的转换规则，二叉树的根是第一株树的根，而第一株树的其他结点为左子树，所以左子树的结点数为 n1-1，选 A。

5．D【详解】哈夫曼树的构造原理，所有结点度数要么为 0，要么为 2，而 D 选项的编码会出现度为 1 的结点，所以选 D。

6．B 【详解】根据关键路径定义，所以选 B。

7．C 【详解】简单环路也是简单路径，只是起点和终点重复，所以长度不会超过 n,选 C。

8．C 【详解】看序列数据分布，发现 1 与 15 交换，4 与 7 交换，满足希尔排序思想，所以选 C。

9．C【详解】比较次数与初态有关的是插入排序，快速排序，所以选 C。

10．D【详解】归并排序和气泡排序是稳定的，所以选 D。

二、填空题：11~15 小题，每小题 2 分，共 10 分。把符合题目要求的答案填写在下划线处。

11． +a*-bcd ； abc－d*+

【详解】中缀表达式转换成前缀表达式需要设置两个栈 R、Q，R 用于存储运算符，Q 用于存储转换后的表达式。对中缀表达式从右至左依次扫描，当遇到操作数时入栈 Q，扫描到操作符时，将操作符压入栈中，进栈的原则是保持栈顶操作符的优先级要高于栈中其他操作符的优先级；否则，将栈顶操作符依次出栈并入栈 Q，直到满足要求为止；遇到“)”进栈，当遇到“(”时，退 R 栈入 Q 直到“)”为止。

中缀表达式转换成后缀表达式要对中缀表达式从左至右依次扫描，由于操作数的顺序保持不变，当遇到操作数时直接输出，设立一个栈用以保存操作符，扫描到操作符时，将操作符压入栈中，进栈的原则是保持栈顶操作符的优先级要高于栈中其他操作符的优先级；否则，将栈顶操作符依次退栈并输出，直到满足要求为止；遇到“(”进栈，当遇到“)”时，退栈输出直到“)”为止。

12． n(n-1) ； n

【详解】图强连通的定义是任意两点之间都有一条路径，最多时每个顶点发出 n-1 条边，n 个顶点共有 n(n-1)条边。最少时每个结点至少要一条出路(单节点除外)，至少有 n 条边,正好可以组成一个环。

13． 63 ； 20

【详解】最多时满二叉树，即 2^6-1=63，最少时可以利用公式 $N_h=N_{h-1}+N_{h-2}+1$ 求得，其中 N_h 代表高度为 h 的 AVL 树最少结点树，N_{h-1} 代表 A V L 树的一棵子树，N_{h-2} 代表另一棵子树，N_0＝0，N_1＝1，N_2＝2，可以计算出最少为 20。

14． 242 ； 31

【详解】根据 B-树定义，3 阶 B 树最多 3^5-1=242，第 h 层最少包含结点数为 $2\lceil m/2\rceil^{h-1}$，所以高度为 5 最少关键字数 $N \geq 2\lceil 3/2\rceil^{5-1}-1=31$

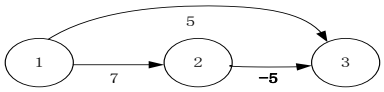
15． 4 ； 1

【详解】最多比较是树的高度 $\log_2 11$,取整为 4 。最少比较 1 次。

三、简答题：16~17 小题，每小题 10 分，共 20 分。

16.

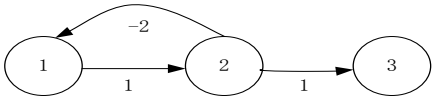
（1）例如：



以顶点 1 为源点，按 Dijkstra 算法，顶点 1 到顶点 3 的最短路径为（1，3），长度为 5；实际的最短路径为（1，2，3），长度为 2。 (3 分)

（2）例如，在公共交通费用网络中，对于某段路程，乘公交车单位给补助，而打车实报实销。如果补助费用大于打车费用，则这段路程的费用即为负权值。 (3 分)

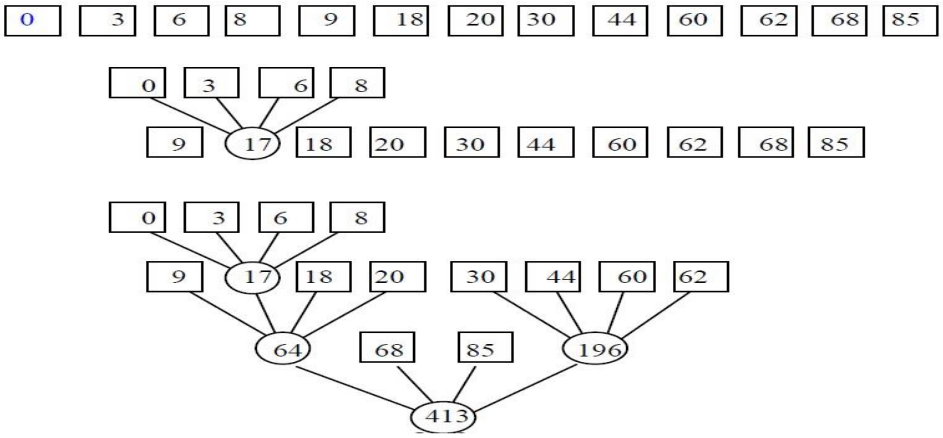
（3）例如：



顶点 1 到顶点 2 的路径长度可以任意小。 (4 分)

17.（1）初始归并段个数 n =12，外归并路数 k = 4，计算(n-1) % (k-1) = 11 % 3 =2≠0，必须补 k-1-2=1 个长度为 0 的空归并段，才能构造 k 路归并树。此时，归并树的内结点应有(n-1+1)/(k-1) = 12/3 = 4 个。 (2 分)

最佳归并树图示如下：



(4分)

（2）WPL = (3+6+8)*3 + (9+18+20+30+44+60+62)*2+(68+85)*1== 690 (2分)

访问外存的次数：690×2＝1380 （2分）

四、算法设计题：18~19 小题，每小题 10 分，共 20 分。按以下要求设计算法：⑴给出算法的基本设计思想；⑵根据设计思想，采用 C 或 C++或 JAVA 语言描述算法。⑶说明你所设计算法的时间和空间复杂度。

18. 分析：因为字符只有 9 种可能，所以可以申请一个 10 大小的数组来统计每个字符出现的次数（以空间换时间），统计完后，我们以原数组的元素值为下标，访问统计数组，直到遇到第一个访问数组元素值为 1 的元素，其所在位置的字符即为第一个只出现一次的字符。

（1）算法的基本设计思想：(2 分)

首先建立一个长度为 10 的计数数组（Hash 表），存放每个数字的出现次数，初始时各单元（桶）均为 0；

然后，扫描字符串，每碰到一个字符（数字），在 Hash 表中找到对应的项并把出现的次数增加一次。

最后，扫描计数数组（Hash 表），得到每个字符出现的次数了。

（2）算法描述(6 分)

```
void FindChar(char* str)
```

```
{   const int tableSize = 10;
unsigned int hashTable[tableSize];
if(str==NULL)
   return;
int hashTable [tableSize]={0};
for(int i=0;i<strlen(str);i++)
    hashTable [str[i]]++;
for(int j=0;j<strlen(str);j++)
{
    if(hashTable [str[j]]==1)
    {
       printf("%c\n",str[j]);
       return;
    }
}
if(j==strlen(str))
    printf("不存在这样的字符!\n");
}
```

（3）时间和空间复杂度(2 分)

算法的基本操作时扫描字符串的每个字符并对其出现次数进行计数，因此时间复杂度为Θ(n)，满足题目要求；

算法的辅助空间开销，取决于字符串对应的字母表的大小，与 n 无关，因此空间复杂度为 0(1)。

19. （1）算法的基本设计思想(2 分)

①如果 AVL 树的左子树的结点数为 a==k-1，则根结点的关键字即为第 k 小的关键字；否则，转②。

②如果 AVL 树的左子树的结点数 a 大于 k-1，则递归地其左子树上查找 AVL 树中第 k 小的关键字；否则转③。

③（AVL 树的左子树的结点数 a<k-1），则递归地其右子树上查找 AVL 树中第 k-a-1 小的关键字，即为 AVL 树中第 k 小的关键字。

（2）算法描述(6 分)

根据题目要求，算法的存储结构可定义如下：

```
typedef struct celltype{
     records key ;
     int size;
     struct celltype *lchild,*rchild ;
}AVLNode;
typedef AVLNode *AVLTree ;
```

算法实现如下：

```
AVLNode *FindKthKey(AVLTree root, int k)
{      if(root->size<k) return NULL;
       if(root->lchild->size==k-1)
           return root;
       else if(root->lchild->size>k-1)
           FindKthKey(root-lchild, k);
       else
```

```
        FindKthKey(root->rchild, k- root->lchild->size-1);
}
```

（3）时间和空间复杂度(2 分)

算法的本质就是递归地查找(Search)AVL树，因此时间和空间性能都取决于AVL树的高度，因此，时间复杂度为$\Theta(\log_2 n)$，满足题目要求；空间复杂度亦为$\Theta(\log_2 n)$。