第 1 题部分代码参考如下：

```
#include <stdio.h>
//#include <malloc.h>
#include <stdlib.h>
#include <string.h>
#define TRUE 1
#define FALSE 0
#define OK 1
#define ERROR 0
#define OVERFLOW -2
typedef int Status;
typedef struct {
    unsigned int weight;
    unsigned int parent, lchild, rchild;
}HTNode, *HuffmanTree;                    //动态分配数组存储赫夫曼树
typedef char **HuffmanCode;               //动态分配数组存储赫夫曼编码表

void Select(HuffmanTree HT, int u, int &s1, int &s2) {
    int j;
    j=1;
    while(j<=u && HT[j].parent!=0) j++;
    s1=j;
    while(j<=u) {
      if(HT[j].parent==0 && HT[j].weight<HT[s1].weight) s1=j;
      j++;
    }
    HT[s1].parent = u+1;
    j = 1;
    while(j<=u && HT[j].parent!=0) j++;
    s2 = j;
    while(j<=u) {
      if(HT[j].parent==0 && HT[j].weight<HT[s2].weight) s2=j;
      j++;
    }
    if (s1>s2) {
      j=s1; s1=s2; s2=j;
    }
}

void HuffmanCoding(HuffmanTree &HT, HuffmanCode &HC, int *w, int n) {
  // 算法 6.12
  // w 存放 n 个字符的权值(均>0)，构造哈夫曼树 HT，
  // 并求出 n 个字符的哈夫曼编码 HC
  int i, j, m, s1, s2, start;
  char *cd;
  unsigned int c, f;
```

```
if (n<=1) return;
m = 2 * n - 1;
HT = (HuffmanTree)malloc((m+1) * sizeof(HTNode));   // 0 号单元未用
for (i=1; i<=n; i++) { //初始化
   HT[i].weight=w[i-1];
   _____;
   _____;
   _____;
}
for (i=n+1; i<=m; i++) { //初始化
   _____;
   _____;
   _____;
   _____;
}
printf("\n 哈夫曼树的构造过程如下所示：\n");
printf("HT 初态:\n   结点   weight   parent   lchild   rchild");
for (i=1; i<=m; i++)
   printf("\n%4d%8d%8d%8d%8d", i, HT[i].weight, HT[i].parent, HT[i].lchild, HT[i].rchild);
for (i=n+1; i<=m; i++) {   // 建哈夫曼树
   // 在 HT[1..i-1]中选择 parent 为 0 且 weight 最小的两个结点，
   // 其序号分别为 s1 和 s2。
   _____;
   HT[s1].parent = i;   _____;
   HT[i].lchild = s1;   _____;
   HT[i].weight =  _____;
   printf("\nselect: s1=%d     s2=%d\n", s1, s2);
   printf("   结点   weight   parent   lchild   rchild");
   for (j=1; j<=i; j++)
      printf("\n%4d%8d%8d%8d%8d", j, HT[j].weight, HT[j].parent, HT[j].lchild, HT[j].rchild);
}
printf("\n");

//--- 从叶子到根逆向求每个字符的哈夫曼编码 ---
HC = (HuffmanCode)malloc((_____)*sizeof(char *));   //分配 n 个字符编码的头指针向量
_____;      // 分配求编码的工作空间
_____;                                          // 编码结束符。
for (i=1; i<=n; ++i) {                          // 逐个字符求哈夫曼编码
   _____;                                          // 编码结束符位置
   for (c=i, f=HT[i].parent;_____; c=f,_____)
      // 从叶子到根逆向求编码
      if (HT[f].lchild==c)  _____;
      else  _____  = '1';
   HC[i] = (char *)malloc((_____)*sizeof(char));
         // 为第 i 个字符编码分配空间
   strcpy(HC[i],  _____);       // 从 cd 复制编码(串)到 HC
}
free(cd);     // 释放工作空间
```

```
    printf("输出各结点的赫夫曼编码：\n");
    for(i=1;i<=n;i++)
        printf("%2d    %s\n", i,  _____);
} // HuffmanCoding

void main() {
    int w[]={5, 29, 7, 8, 14, 23, 3, 11}, n=8;
    HuffmanTree HT;
    HuffmanCode HC;
    HuffmanCoding(HT,HC,w,n);
}
```