# Problem 1

```
#include<iostream>
#include<cstring>
using namespace std;
#define MAX 1000
typedef struct {
    int weight;
    int parent, lchild, rchild;
}HTNode, *HuffmanTree;
typedef char  **HuffmanCode;
void Select(HuffmanTree &HT, int n, int i, int s[], int vis[])
{
    int min = 65535;
    int j, m[2] = { 0 }, k = 0;
    while (1)
    {
        for (j = 1; j <= i; j++)
            if (HT[j].weight < min && vis[j] == 0)
            {
                min = HT[j].weight;
                m[k] = j;
            }
        vis[m[k]] = 1;
        min = 65535;
        k++;
        if (k == 2)
            break;
    }
    if (m[0] < m[1])
    {
        s[0] = m[0];
        s[1] = m[1];
    }
    else
    {
        s[0] = m[1];
        s[1] = m[0];
    }
}
void HuffmanCoding(HuffmanTree &HT, HuffmanCode &HC, int w[], int&
n, int &sum_w)
{
    int *vis = new int[2 * n - 1]{ 0 };
```

```
    if (n <= 1)
        return;
    int m;
    int s[2] = { 0,0 };
    m = 2 * n - 1;
    HT = new HTNode[m + 1];
    HuffmanTree p;
    int i;
    for (p = HT + 1, i = 1; i <= n; i++, p++, w++)
        *p = { *w,0,0,0 };
    for (; i <= m; i++)
        *p = { 0,0,0,0 };
    for (i = n + 1; i <= m; i++)
    {
        Select(HT, n, i - 1, s, vis);
        HT[s[0]].parent = i;
        HT[s[1]].parent = i;
        HT[i].lchild = s[0];
        HT[i].rchild = s[1];
        HT[i].weight = HT[s[0]].weight + HT[s[1]].weight;
    }
    HT[m].parent = 0;
    int start, c, f;
    HC = (HuffmanCode)malloc((n + 1) * sizeof(char *));
    char* cd = new char[n];
    cd[n - 1] = '\0';
    for (i = 1; i <= n; i++)
    {
        start = n - 1;
        for (c = i, f = HT[c].parent; f != 0; c = f, f =
HT[f].parent)
            if (HT[f].lchild == c)cd[--start] = '0';
            else cd[--start] = '1';
        HC[i] = new char[n - start];
        strcpy(HC[i], &cd[start]);
        sum_w = sum_w + HT[i].weight *strlen(HC[i]);
    }
    delete[]cd;
}
int main()
{
    int n, sum_w = 0;
    cin >> n;
    int *w = new int[MAX];
```

```
    HuffmanTree HT;
    HuffmanCode HC;
    for (int i = 0; i < n; i++)
        cin >> w[i];
    HuffmanCoding(HT, HC, w, n, sum_w);
    cout << sum_w << endl;
    return 0;
}
```

部分函数的另一种表达：

```
/*int Min(HuffmanTree &HT, int i)
{
    int min =65535;
    int j, flag;
    for (j = 1; j <= i; j++)
        if (HT[j].weight < min && HT[j].parent == 0)
        {
            min = HT[j].weight;
            flag = j;
        }
    HT[flag].parent = 1;
    return  flag;
}

void Select(HuffmanTree &HT, int i, int &s1, int &s2)
{
    s1 = Min(HT, i);
    s2 = Min(HT, i);
    int temp=0;
    if (s1>s2)
    {
        temp = s1;
        s1 = s2;
        s2 = temp;
    }
}
void HuffmanCoding(HuffmanTree &HT, HuffmanCode &HC, int w[], int& n, int &sum_w)
{
    if (n <= 1)
        return;
    int m;
    int s1, s2;
    m = 2 * n - 1;
    HT = new HTNode[m + 1];
    HuffmanTree p;
```

```
    int i;
    for (p = HT + 1, i = 1; i <= n; i++, p++, w++)
        *p = { *w,0,0,0 };
    for (; i <= m; i++)
        *p = { 0,0,0,0 };
    for (i = n + 1; i <= m; i++)
    {
        Select(HT,i-1,s1,s2);
        HT[s1].parent = i;
        HT[s2].parent = i;
        HT[i].lchild = s1;
        HT[i].rchild = s2;
        HT[i].weight = HT[s1].weight + HT[s2].weight;
    }
    HT[m].parent = 0;
    for (int j = n + 1; j <= m; j++)
        sum_w += HT[j].weight;
}*/
```

# Problem 2

```
#include<iostream>
#include<cstring>
using namespace std;
typedef struct {
    int weight;
    int parent, lchild, rchild;
}HTNode, *HuffmanTree;
typedef char  **HuffmanCode;
void InputHuffman(HuffmanTree &HT, int &n)
{
    int i, w, pa, lc;
    int m = 2 * n - 1;
    HT = new HTNode[m + 1];
    HuffmanTree p, q;
    for (p = HT + 1, i = 1; i <= m; i++, p++)
        *p = { 0,0,0,0 };
    for (q = p = HT + 1, i = 1; i <= m; i++, p++)
    {
        cin >> w >> pa >> lc;
        p->weight = w;
        p->parent = pa;
        if (i != m)
            if (lc == 0)
                (q + pa - 1)->lchild = i;
```

```
            else
                (q + pa - 1)->rchild = i;
        else
            break;
    }
}
void HuffmanCoding(HuffmanTree &HT, HuffmanCode &HC, int& n)
{
    int start, c, f, i;
    HC = (HuffmanCode)malloc((n + 1) * sizeof(char *));
    char* cd = new char[n];
    cd[n - 1] = '\0';
    for (i = 1; i <= n; i++)
    {
        start = n - 1;
        for (c = i, f = HT[c].parent; f != 0; c = f, f =
HT[f].parent)
            if (HT[f].lchild == c)cd[--start] = '0';
            else cd[--start] = '1';
        HC[i] = new char[n - start];
        strcpy(HC[i], &cd[start]);
        cout << i << " " << HC[i] << endl;
    }
    delete[]cd;
}
int main()
{
    int n;
    cin >> n;
    HuffmanTree HT;
    HuffmanCode HC;
    InputHuffman(HT, n);
    HuffmanCoding(HT, HC, n);
    return 0;
}
```

# Problem 3

```
#include<iostream>
#include<cstring>
#define MAX 1000
using namespace std;
typedef struct Code {
    int c_ASCII;
```

```
    char *str;
}*Cd;
void input(Cd &cd, int n)
{
    cd = new Code[n + 1];
    int i = 1;
    while (i <= n)
    {
        cin >> cd[i].c_ASCII;
        cd[i].str = new char[10];
        cin >> cd[i].str;
        i++;
    }
}
void Decoding(Cd &cd, int n, char s[])
{
    char *p = s;
    char s1[10];
    int i = 0, k;
    while (*p != '\0')
    {
        s1[i++] = *p;
        s1[i] = '\0';
        for (int k = 1; k <= n; k++)
        {
            if (strcmp(s1, cd[k].str) == 0)
            {
                cout << char(cd[k].c_ASCII);
                memset(s1, '\0', sizeof(s1));
                i = 0;
                break;
            }
        }
        p++;
    }
}
int main()
{
    int n;
    cin >> n;
    Cd cd;
    input(cd, n);
    char str1[MAX];
    cin >> str1;
```

```
    Decoding(cd, n, str1);
    return 0;
}
```