#include<iostream>

using namespace std;

class cNode

{

public:

int info;

char carr;

char letter[28];

char dierction[28];

int ct=0;

cNode\* pNext = NULL, \* pLeft = NULL, \* pRight = NULL;

};

void coby(cNode\*ptrav,cNode\* pf)

{

for (int i = 0; pf->dierction[i] != NULL; i++)

{

ptrav->dierction[i] = pf->dierction[i];

}

}

class cTree

{

public:

cNode\* pRoot;

cTree()

{

pRoot = NULL;

}

void remove(cNode\* ptrav)

{

if (ptrav == NULL)

{

return;

}

remove(ptrav->pLeft);

remove(ptrav->pRight);

delete ptrav;

}

void insert(cNode\* pnn)

{

if (pRoot == NULL)

{

pRoot = pnn;

}

else

{

cNode\* ptrav = pRoot;

cNode\* pB = NULL;

while (ptrav != NULL)

{

pB = ptrav;

if (pnn->info > ptrav->info)

{

ptrav = ptrav->pRight;

}

else

{

ptrav = ptrav->pLeft;

}

}

if (pnn->info > pB->info)

{

pB->pRight = pnn;

}

else

{

pB->pLeft = pnn;

}

}

}

void Disp(cNode\* pTrav, cNode\* pf)

{

int i = 0;

if (pTrav == NULL)

{

return;

}

//if (pTrav->pLeft != NULL)

//{

//if (pf != NULL && pTrav->ct != 1)

//{

// coby(pTrav, pf);

// for (i = 0; pTrav->dierction[i] != NULL; i++)

// {

// }

//

// if (pf->ct == 0)

// {

// pTrav->dierction[i] = '0';

// //pTrav->ct = 1;

// }

//}

//if (pTrav->pLeft != NULL)

//{

Disp(pTrav->pLeft, pTrav);

//}

////cout << pTrav->letter << ",," << pTrav->info << endl;

//for (i = 0; pTrav->dierction[i] != NULL; i++)

//{

//}

//if (pTrav->ct != i - 1)

//{

// pTrav->dierction[i] = '1';

//}

//pTrav->ct = 1;

//if (pTrav->pRight != NULL)

//{

Disp(pTrav->pRight, pTrav);

//}

//return;

//if (pf != NULL && pTrav->ct != 0)

//{

// coby(pTrav, pf);

// for (i = 0; pTrav->dierction[i] != NULL; i++)

// {

// }

//

// if (pf->ct == 0)

// {

// pTrav->dierction[i] = '1';

// //pTrav->ct = 1;

// }

//}

}

void DispAll()

{

Disp(pRoot,NULL);

}

};

class List

{

public:

cNode\* pHead;

cNode\* pTail;

List()

{

pHead = NULL;

pTail = NULL;

}

~List()

{

cNode\* pTemp = pHead;

while (pTemp != NULL)

{

//pHead = pHead->pNext;

//delete pTemp;

//pTemp = pHead;

}

}

void Attach(cNode\* pnn)

{

if (pHead == NULL)

{

pHead = pnn;

pTail = pnn;

}

else

{

pTail->pNext = pnn;

pTail = pnn;

}

}

void Display()

{

int j = 0;

cNode\* pTrav = pHead;

while (pTrav != NULL)

{

//cout << pTrav->letter << " " << pTrav->info << endl;

pTrav = pTrav->pNext;

}

//cout << endl;

}

};

void sorting(int ct[],char x[])

{

int temp;

char tem;

for (int i = 0; x[i]!=NULL; i++)

{

for (int j = i + 1; x[j]!=NULL; j++)

{

if (ct[i] > ct[j])

{

temp = ct[i];

ct[i] = ct[j];

ct[j] = temp;

tem = x[i];

x[i] = x[j];

x[j] = tem;

}

}

}

}

void merge(cNode\* pnn, cNode\* pn, cNode\* pCurr, cTree\* tree)

{

int tot = 0;

int driver = 0;

tot += pnn->info;

tot += pn->info;

//cout << tot;

pCurr->info = tot;

for (int i = 0; pnn->letter[i] != NULL; i++)

{

pCurr->letter[driver] = pnn->letter[i];

driver++;

}

for (int i = 0; pn->letter[i] != NULL; i++)

{

pCurr->letter[driver] = pn->letter[i];

driver++;

}

pCurr->pLeft = pnn;

pCurr->pRight = pn;//from

tree->pRoot = pCurr;

}

int sort(List l, cNode\* pCurr)

{

cNode\* pnn = l.pHead;

cNode\* pb = NULL;

int f = 0;

if (pnn != NULL)

{

while (pnn->pNext != NULL)

{

if (pCurr->info < pnn->info)

{

if (pnn->pNext == NULL)

{

pnn->pNext = pCurr;

l.pTail = pCurr;

f = 1;

break;

}

if (pb == NULL)

{

l.pHead = pCurr;

pCurr->pNext = pnn;

f = 1;

break;

}

else

{

pb->pNext = pCurr;

pCurr->pNext = pnn;

f = 1;

break;

}

}

pb = pnn;

pnn = pnn->pNext;

}

if (f == 0)

{

l.pTail->pNext = pCurr;

l.pTail = pCurr;

}

}

return 0;

}

void binaryput(cTree\* tree, char g,List &ph2)

{

cNode\* ph22 = new cNode;

cNode\* pnn = tree->pRoot;

int f = 0;

char arr[260] = { NULL };

int s = 0;

for (int i = 0; ; i++)

{

for (int m = 0; pnn->pRight->letter[m] != NULL; m++)

{

if (pnn->pRight->letter[m] == g)

{

// ////cout << right;

arr[s] = '1';

s++;

pnn = pnn->pRight;

break;

}

}

if (pnn->letter[1] == NULL && pnn->letter[0] == g)

{

//cout << arr;

for (int i = 0; arr[i] != NULL;i++)

{

ph22->dierction[i] = arr[i];

}

ph22->carr= g;

ph2.Attach(ph22);

break;

}

for (int m = 0; pnn->pLeft->letter[m] != NULL; m++)

{

if (pnn->pLeft->letter[m] == g)

{

// //cout << left;

arr[s] = '0';

s++;

pnn = pnn->pLeft;

break;

}

}

if (pnn->letter[1] == NULL && pnn->letter[0] == g)

{

//cout << arr;

for (int i = 0; arr[i] != NULL;i++)

{

ph22->dierction[i] = arr[i];

}

ph22->carr = g;

ph2.Attach(ph22);

break;

}

}

}

void check(char ar[], cNode\* ph22,char end[],int &enddriver, int &ardriver)

{

int i;

/////////check;

int ct = 0;

if (ar != NULL)

{

while (ph22 != NULL)

{

for (i = 0; ph22->dierction[i] != NULL; i++)

{

}

for (int s = 0; ar[s] != NULL, ph22->dierction[s] != NULL; s++)

{

if (ph22->dierction[s] == ar[s])

{

ct++;

}

}

if (ct == i)

{

end[enddriver] = ph22->carr;

enddriver++;

ardriver = 0;

for (int mm = 0; mm < 500; mm++)

{

ar[mm] = NULL;

}

return;

}

ct = 0;

ph22 = ph22->pNext;

}

}

}

void main()

{

List l;

char x[1000] = { NULL };//= { 'a','a','a','a','b','b','b','d','d','e','e','e','e','e','g','g','g','g','g','g','f','f','f','f','f','f','m' };

x[0] = 'a';

x[5] = 'c';

int ct[26] = {0}, ctB = 0, ctD = 0;

char ar[26] = { 'a','b','c','d','e','f', 'g', 'h', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p', 'q', 'r', 's', 't', 'u', 'v', 'w', 'x', 'y', 'z'};

for (int i = 0; x[i]!=NULL; i++)

{

if (x[i] == 'a')

{

ct[0]++;

}

if (x[i] == 'b')

{

ct[1]++;

}

if (x[i] == 'c')

{

ct[2]++;

}

if (x[i] == 'd')

{

ct[3]++;

}

if (x[i] == 'e')

{

ct[4]++;

}

if (x[i] == 'f')

{

ct[5]++;

}

if (x[i] == 'g')

{

ct[6]++;

}

if (x[i] == 'h')

{

ct[7]++;

}

if (x[i] == 'i')

{

ct[8]++;

}

if (x[i] == 'j')

{

ct[9]++;

}

if (x[i] == 'k')

{

ct[10]++;

}

if (x[i] == 'l')

{

ct[11]++;

}

if (x[i] == 'm')

{

ct[12]++;

}

if (x[i] == 'n')

{

ct[13]++;

}

if (x[i] == 'o')

{

ct[14]++;

}

if (x[i] == 'p')

{

ct[15]++;

}

if (x[i] == 'q')

{

ct[16]++;

}

if (x[i] == 'r')

{

ct[17]++;

}

if (x[i] == 's')

{

ct[18]++;

}

if (x[i] == 't')

{

ct[19]++;

}

if (x[i] =='u')

{

ct[20]++;

}

if (x[i] == 'v')

{

ct[21]++;

}

if (x[i] == 'w')

{

ct[22]++;

}

if (x[i] == 'x')

{

ct[23]++;

}

if (x[i] == 'y')

{

ct[24]++;

}

if (x[i] == 'z')

{

ct[25]++;

}

}

sorting(ct,x);

cNode\* pnn;

cNode\* coby;

for (int i = 0; i < 26; i++)

{

pnn = new cNode;

pnn->letter[0] = ar[i];

pnn->info = ct[i];

l.Attach(pnn);

}

l.Display();

pnn = l.pHead;

cTree\* tree = new cTree;

//cout << "---------------------------------" << endl;

pnn = l.pHead;

coby = l.pHead;

l.pTail;

cNode\* pCurr = new cNode;

cNode\* pn = new cNode;

while (pnn != NULL)

{

pnn = l.pHead;

pCurr = new cNode;

pn = pnn->pNext;

if (pn == NULL)

{

break;

}

merge(pnn, pn, pCurr, tree);

//cout << pCurr->letter << endl;

l.pHead = pnn->pNext->pNext;

// int r=sort(l, pCurr);

/////////////////////////////////////////////////////////////////////////

cNode\* pnn = l.pHead;

cNode\* pb = NULL;

int f = 0;

if (pnn != NULL)

{

while (pnn->pNext != NULL)

{

if (pCurr->info < pnn->info)

{

if (pnn->pNext == NULL)

{

pnn->pNext = pCurr;

l.pTail = pCurr;

f = 1;

break;

}

if (pb == NULL)

{

l.pHead = pCurr;

pCurr->pNext = pnn;

f = 1;

break;

}

else

{

pb->pNext = pCurr;

pCurr->pNext = pnn;

f = 1;

break;

}

}

pb = pnn;

pnn = pnn->pNext;

}

if (f == 0)

{

l.pTail->pNext = pCurr;

l.pTail = pCurr;

}

}

////////////////////////////////////////////////////////////////////////

pnn = l.pHead;

if (pnn == NULL)

{

break;

}

}

//

//cout << "wait 2" << endl;

tree->DispAll();

int c = 0;

//

//tree->disple(tree->pRoot,'e',c);

////cout<<;

//cout << "wait" << endl;

List phase2;

for (int i = 0; i < 26 /\*num of a->z\*/; i++)

{

////cout << ar[i] << endl;

binaryput(tree, ar[i],phase2);

//

}

////cout << tree->pRoot->pLeft->info;

//

gets\_s(x);

//char orig[1000] = { NULL };// original[27] = { 'a','a','a','a','b','b','b','d','d','e','e','e','e','e','g','g','g','g','g','g','f','f','f','f','f','f','m'};

//gets\_s(orig);

cNode\* ph22 = phase2.pHead;

while (ph22 != NULL)

{

cout << ph22->carr<<",,"<<ph22->dierction<<endl;

ph22=ph22->pNext;

}

//////tmam hna

pnn = new cNode;

int ibit = 7;

int bigct = 0;

char array[70000] = { NULL };

char ptemp = 0;

int driver = 0;

int fletter = 0;

for (int i = 0; x[i]!=NULL/\*number of letters\*/; i++)

{

char z = x[i];

ph22 = phase2.pHead;

fletter = 0;

while (ph22->carr!= z)

{

if (z>='a'&&z<='z')

{

}

else

{

fletter = 1;

break;

}

ph22 = ph22->pNext;

}

int ct = 0;

if (fletter != 1)

{

for (; ibit >= 0; ibit--)

{

if (ph22->dierction[ct] == '1')

{

ptemp = ptemp | (1 << ibit);

}

if (ptemp == -82)

{

cout << "hna";

}

ct++;

if (ph22->dierction[ct] == NULL)

{

if (ibit == 0)

{

ibit = 8;

array[driver] = ptemp;

ptemp = 0;

driver++;

}

ibit--;

break;

}

if (ibit == 0)

{

ibit = 8;

array[driver] = ptemp;

ptemp = 0;

driver++;

}

}

}

}

array[driver] = ptemp;

// array[0] = array[0] | (1 << 7);

for (int i = 0; array[i]!=NULL;i++)

{

cout << array[i]<<endl;

}

int t = 0;

int ctt = 0;

//////////////////فك ضغط

char done[10000] = { NULL };

int donedriver = 0;

char arrsearch[1000] = {NULL};

/// array[4] = -1;

/// array[5] = -11;

/// array[6] = 85;

/// array[7] = 0;

/// array[8] = 10;

for(int i=0;array[i]!=NULL/\*number of used bits\*/;i++)

{

for (ibit = 7; ibit > -1; ibit--)

{

char reg = array[i] & (1 << ibit);

//cout << (1 << ibit);

if (reg !=0)

{

// cout <<'1'<<endl;

done[donedriver] = '1';

donedriver++;

}

else

{

// cout << "zero" << endl;

done[donedriver] = '0';

donedriver++;

}

}

}

char arrr[500];

char end[10000] = { NULL };

int enddriver = 0;

int ardriver = 0;

//for()

for (int i = 0; i < donedriver; i++)

{

arrr[ardriver] = done[i];

ardriver++;

check(arrr, phase2.pHead, end, enddriver, ardriver);

}

for (int i = 0; end[i]!=NULL; i++)

{

cout << end[i];

}

cout << endl << endl;

system("pause");

}