

## 专题7

### 目录

Problem ID	Title
1001	Immediate Decodability
1002	统计难题
1003	Phone List
1004	What Are You Talking About
1005	Ancient Printer
1006	单词数
1007	Hat's Words
1008	Repository
1009	T9

## 1001

### AC代码

```
1  #include<bits/stdc++.h>
2  using namespace std;
3
4  int flag;
5  struct tr {
6      int num;
7      tr* ch[2];
8      tr()
9      {
10         for (int i = 0; i < 2; i++) {
11             ch[i] = 0;
12         }
13         num = 0;
14     }
15 }*root;
16
17 void insert(string s)
18 {
19     if (flag)return;
20     tr* p = root;
21     for (int i = 0; s[i] != '\0'; i++) {
22         int temp = s[i] - '0';
23         if (!p->ch[temp]) {
```

```

24         p->ch[temp] = new tr;
25     }
26     p = p->ch[temp];
27     if (p->num == 2) {
28         flag = 1;
29         return;
30     }
31     p->num = 1;
32 }
33 p->num = 2;
34 }
35 void Free(tr* p)
36 {
37     for (int i = 0; i < 2; i++)
38         if (p->ch[i])
39             Free(p->ch[i]);
40     delete p;
41 }
42 void solve()
43 {
44     string s;
45     int cnt = 1;
46     root = new tr;
47     while (cin >> s) {
48         if (s == "9") {
49             if (flag) {
50                 cout << "Set " << cnt++;
51                 cout << " is not immediately decodable\n";
52             }
53             else {
54                 cout << "Set " << cnt++;
55                 cout << " is immediately decodable\n";
56             }
57             Free(root);
58             root = new tr;
59             flag = 0;
60             continue;
61         }
62         insert(s);
63     }
64 }
65 int main()
66 {
67     ios::sync_with_stdio(false);
68     cin.tie(0); cout.tie(0);
69     solve();
70     return 0;
71 }

```

```
1  #include<bits/stdc++.h>
2  using namespace std;
3
4  struct tr {
5      int cnt;
6      tr* ch[27];
7      tr()
8      {
9          for (int i = 0; i < 26; i++)
10             ch[i] = 0;
11         cnt = 0;
12     }
13 }* root;
14 void init()
15 {
16     root = new tr;
17 }
18 void insert(string s)
19 {
20     tr* p = root;
21     for (int i = 0; s[i] != '\0'; i++) {
22         int temp = s[i] - 'a';
23         if (p->ch[temp] == 0)
24             p->ch[temp] = new tr;
25         p = p->ch[temp];
26         p->cnt++;
27     }
28 }
29 void Free(tr* p)
30 {
31     for (int i = 0; i < 26; i++)
32         if (p->ch[i])
33             Free(p->ch[i]);
34     delete p;
35 }
36 int count(string s)
37 {
38     tr* p = root;
39     for (int i = 0; s[i] != '\0'; i++) {
40         int temp = s[i] - 'a';
41         if (p->ch[temp] == 0)
42             return 0;
43         else
44             p = p->ch[temp];
45     }
46     return p->cnt;
47 }
48 void solve()
49 {
50     init();
51     string s;
52     while (getline(cin, s), !s.empty())
53         insert(s);
```

```

54     while (cin >> s)
55         cout << count(s) << endl;
56     Free(root);
57 }
58 int main()
59 {
60     ios::sync_with_stdio(false);
61     cin.tie(0); cout.tie(0);
62     solve();
63     return 0;
64 }

```

---

## 1003

### AC代码

```

1  #include<bits/stdc++.h>
2  using namespace std;
3
4  int flag;
5  struct tr {
6      int num;
7      tr* ch[11];
8      tr()
9      {
10         for (int i = 0; i < 10; i++) {
11             ch[i] = 0;
12         }
13         num = 0;
14     }
15 }*root;
16
17 void insert(string s)
18 {
19     if (flag)return;
20     tr* p = root;
21     int len = s.size();
22     for (int i = 0; i<len; i++) {
23         int temp = s[i] - '0';
24         if (!p->ch[temp]) {
25             p->ch[temp] = new tr;
26         }
27         p = p->ch[temp];
28         if ((p->num == 2)|| (p->num==1&&i==len-1)) {
29             flag = 1;
30             return;
31         }
32         p->num = 1;
33     }
34     p->num = 2;
35 }
36 void Free(tr* p)
37 {
38     for (int i = 0; i < 10; i++)

```

```

39         if (p->ch[i])
40             Free(p->ch[i]);
41         delete p;
42     }
43     void init()
44     {
45         root = new tr;
46     }
47     void solve()
48     {
49         string s;
50         int n;
51         init();
52         cin >> n;
53         while (n--) {
54             cin >> s;
55             insert(s);
56         }
57         if (flag)cout << "NO\n";
58         else cout << "YES\n";
59         Free(root);
60         flag = 0;
61     }
62     int main()
63     {
64         ios::sync_with_stdio(false);
65         cin.tie(0); cout.tie(0);
66         int tt;
67         cin >> tt;
68         while (tt--)solve();
69         return 0;
70     }

```

---

## 1004

### AC代码

```

1  #include<bits/stdc++.h>
2  using namespace std;
3  int main()
4  {
5      char s[3005], a[100];
6      string s1, s2, s3;
7      int k=0;
8      map<string, string>v;
9      cin >> s;
10     while (cin >> s1) {
11         if (s1 == "END")break;
12         cin >> s2;
13         v[s2] = s1;
14     }
15     getchar();
16     while (gets(s))
17     {

```

```

18         if (strcmp(s, "START") == 0)continue;
19         if (strcmp(s, "END") == 0)break;
20         for (int i = 0; s[i] != '\0'; i++) {
21             if (islower(s[i]))
22                 a[k++] = s[i];
23             else {
24                 a[k] = '\0';
25                 if (v.count(a))
26                     cout << v[a];
27                 else cout << a;
28                 cout << s[i];
29                 for (int j = 0; j < k; j++)
30                     a[j] = '\0';
31                 k = 0;
32             }
33         }
34         cout << endl;
35     }
36     return 0;
37 }

```

---

**1005**

**AC代码**

```

1  #include<bits/stdc++.h>
2  using namespace std;
3  string s;
4  struct tr {
5      char c;
6      vector<tr*>ch;
7      tr() {
8          c = 0;
9          for (int i = 0; i < ch.size(); i++)
10             ch[i] = 0;
11     }
12 }*root,*add;
13 void insert(tr* p, int cnt)
14 {
15     if (cnt == s.size())
16         return;
17     for (int i = 0; i < p->ch.size(); i++) {
18         if (p->ch[i]->c != s[cnt])
19             continue;
20         p = p->ch[i];
21         insert(p, cnt + 1);
22         return;
23     }
24     add = new tr;
25     add->c = s[cnt];
26     p->ch.push_back(add);
27     p = add;
28     insert(p, cnt + 1);
29 }

```

```

30 int dfs(tr* p)
31 {
32     int sum = 0;
33     int len = p->ch.size();
34     for (int i = 0; i < len; i++)
35         sum += dfs(p->ch[i]) + 1;
36     return sum;
37 }
38 void Free(tr* p)
39 {
40     int len = p->ch.size();
41     for (int i = 0; i < len; i++)
42         Free(p->ch[i]);
43     delete p;
44 }
45 void solve()
46 {
47     int tt;
48     while (cin >> tt) {
49         root = new tr;
50         int mx = 0;
51         for (int i = 0; i < tt; i++) {
52             cin >> s;
53             insert(root, 0);
54             mx = max(mx, int(s.size()));
55         }
56         cout << dfs(root)*2-mx+tt << endl;
57         Free(root);
58     }
59 }
60 int main()
61 {
62     ios::sync_with_stdio(false);
63     cin.tie(0); cout.tie(0);
64     solve();
65     return 0;
66 }

```

---

**1006**

**AC代码**

```

1  #include<bits/stdc++.h>
2  using namespace std;
3
4  struct tr {
5      int num;
6      tr* ch[27];
7      tr(){
8          for (int i = 0; i < 26; i++)
9              ch[i] = 0;
10         num = 0;
11     }
12 }*root;

```

```
13 int sum;
14 void insert(string s)
15 {
16     tr* p = root;
17     for (int i = 0; s[i] != '\0'; i++) {
18         if (s[i] == ' ') {
19             if (i == 0 || s[i - 1] == ' ')
20                 continue;
21             if (p->num == 0)
22                 p->num = 1, sum++;
23             p = root;
24             continue;
25         }
26         int temp = s[i] - 'a';
27         if (!p->ch[temp])
28             p->ch[temp] = new tr;
29         p = p->ch[temp];
30     }
31     if (p->num == 0 && s[s.size() - 1] != ' ' && s[s.size() - 1] != '\0')
32         p->num = 1, sum++;
33 }
34
35 void Free(tr* p)
36 {
37     for (int i = 0; i < 26; i++)
38         if (p->ch[i])
39             Free(p->ch[i]);
40     delete p;
41 }
42 void solve()
43 {
44     string s;
45     root = new tr;
46     while (getline(cin, s), s != "#") {
47         sum = 0;
48         insert(s);
49         cout << sum << endl;
50         Free(root);
51         root = new tr;
52     }
53 }
54 int main()
55 {
56     ios::sync_with_stdio(false);
57     cin.tie(0); cout.tie(0);
58     solve();
59     return 0;
60 }
```



```
1  #include<bits/stdc++.h>
2  using namespace std;
3  char str[50000+5][20];
4  struct tries{
5      tries *next[26];
6      bool sign;
7      tries(){
8          memset(next,NULL,sizeof(next));
9          sign=false;
10     }
11 }* root;
12
13 tries * create(){
14     tries *tmp=new tries();
15     return tmp;
16 }
17
18 void Insert(char *p,tries * root){
19
20     while(*p){
21         int id=*p-'a';
22         if(root->next[id]==NULL){
23             root->next[id]=create();
24         }
25         root=root->next[id];
26         p++;
27     }
28     root->sign=true;
29 }
30
31 bool searchTwo(char *p){
32     tries *tmp=root;
33     while(*p){
34         int id=*p-'a';
35         if(tmp->next[id]!=NULL){
36             p++;
37             tmp=tmp->next[id];
38         }
39         else return false;
40     }
41     if(tmp->sign) return true;
42     return false;
43 }
44
45 bool searchOne(char *p){
46     tries *tmp=root;
47     while(*p){
48         int id=*p-'a';
49         if(tmp->next[id]!=NULL){
50             if(tmp->sign){
51                 if(searchTwo(p))return true;
52             }
53             p++;tmp=tmp->next[id];
```

```

54     }
55     else return false;
56 }
57 return false;
58 }
59
60 int main()
61 {
62     int cnt;
63     root=new tries();
64     while(scanf("%s",str[cnt])!=EOF){
65         Insert(str[cnt],root);
66         cnt++;
67     }
68     for(int i=0;i<cnt;i++){
69         if(searchone(str[i]))
70             puts(str[i]);
71     }
72     return 0;
73 }

```

## 1008

### AC代码

```

1  #include<bits/stdc++.h>
2  const int N=500005;
3  int n,m,cnt=1,cur;
4  char str[30];
5  struct node
6  {
7      int p[26];
8      int flag;
9      int last;
10 } tree[N];
11 void build(char *s,int cur)
12 {
13     int p=0;
14     while(*s)
15     {
16         if(tree[p].p[*s-'a']==-1)
17             tree[p].p[*s-'a']=cnt++;
18         p=tree[p].p[*s-'a'];
19         if(tree[p].last!=cur)
20         {
21             tree[p].last=cur;
22             tree[p].flag++;
23         }
24         s++;
25     }
26 }
27
28 int find(char *s)
29 {

```

```

30     int p=0;
31     while(*s)
32     {
33         if(tree[p].p[*s-'a']==-1)
34             return 0;
35         p=tree[p].p[*s-'a'];
36         s++;
37     }
38     return tree[p].flag;
39 }
40 int main()
41 {
42     int n,i,m,num;
43     for(i=0;i<N;i++)
44     {
45         tree[i].flag=0;
46         tree[i].last=-1;
47         memset(tree[i].p,-1,sizeof(tree[i].p));
48     }
49     scanf("%d",&n);
50     for(cur=0;cur<n;cur++)
51     {
52         scanf("%s",str);
53         for(i=0;i<=strlen(str);i++)
54             build(str+i,cur);
55     }
56     scanf("%d",&m);
57     while(m--)
58     {
59         scanf("%s",str);
60         printf("%d\n",find(str));
61     }
62     return 0;
63 }

```

---

1009

AC代码

```

1  #include<bits/stdc++.h>
2  using namespace std;
3  char word[110],res[110],tmp[110];
4  char phone[10][4]={{'a','b','c'},{'d','e','f'},{'g','h','i'},{'j','k','l'},
5  {'m','n','o'},{'p','q','r'},{'s'},{'t','u','v'},{'w','x','y','z'}};
6  int num[10]={3,3,3,3,3,4,3,4},flag,max_p;
7
8  struct tree{
9      int cnt;
10     tree *next[26];
11 }*root;
12
13 tree *Create(){
14     tree *p;
15     p=(tree *)malloc(sizeof(tree));

```

```

15     p->cnt=0;
16     for(int i=0;i<26;++i) p->next[i]=NULL;
17     return p;
18 }
19
20 void Insert(char *word,int k){
21     tree *p=root;
22     int i=0,x,l=strlen(word);
23     while(i<l){
24         x=word[i++]-'a';
25         if((p->next[x])==NULL) p->next[x]=Create();
26         p=p->next[x];
27         p->cnt+=k;
28     }
29 }
30
31 void Search(int l,int len,tree *p){
32     if(l==len){
33         flag=1;
34         if(p->cnt>max_p){
35             max_p=p->cnt;
36             for(int i=0;i<len;++i) res[i]=tmp[i];
37             res[len]='\0';
38         }
39         return;
40     }
41     int pos=word[l]-'2',x;
42     char t;
43     for(int i=0;i<num[pos];++i){
44         t=phone[pos][i];
45         x=t-'a';
46         if((p->next[x])==NULL) continue;
47         tmp[l]=t;
48         Search(l+1,len,p->next[x]);
49     }
50 }
51
52 int main(){
53     int t,c=0,n,p,m,mark;
54     scanf("%d",&t);
55     while(t--){
56         printf("Scenario #%d:\n",++c);
57         scanf("%d",&n);
58         root=Create();
59         for(int i=0;i<n;++i){
60             scanf("%s %d",word,&p);
61             Insert(word,p);
62         }
63         scanf("%d",&m);
64         for(int i=0;i<m;++i){
65             mark=0;
66             scanf("%s",word);
67             int l=strlen(word);
68             for(int j=0;j<l-1;++j){
69                 flag=0,max_p=0;
70                 if(!mark) Search(0,j+1,root);
71                 if(flag) printf("%s\n",res);
72                 else printf("MANUALLY\n"),mark=1;

```

```
73     }
74     printf("\n");
75 }
76 printf("\n");
77 }
78 }
```

---