2016级数据结构第六次上机解题报告

A

思路：

队列的简单应用。注意本题有几个坑点。

1. delete操作若q小于0不合法，等于0合法。
2. get操作若m小于等于0不合法。

代码：

#include <iostream>

#include <string>

#include <queue>

using namespace std;

queue <string> q;

queue <string> t;

int main()

{

int n;

string s;

int p;

while (cin >> n)

{

while (n--)

{

cin >> s;

if (s[0] == 'a')

{

cin >> s;

if (q.size()== 25)

cout << "GzhIsSoHandsome" << endl;

else

q.push(s);

}

else if (s[0] == 'd')

{

cin >> p;

if (p < 0)

cout << "GzhIsSoHandsome" << endl;

else if (p > q.size())

cout << "GzhIsSoHandsome" << endl;

else

{

for (int i = 1; i <= p; i++)

q.pop();

}

}

else if (s[0] == 'g')

{

cin >> p;

if (p <= 0)

cout << "GzhIsSoHandsome" << endl;

else if(p>q.size())

cout << "GzhIsSoHandsome" << endl;

else

{

int cnt = 1;

while (!q.empty())

{

s = q.front();

if (cnt == p)

cout << s << endl;

cnt++;

t.push(s);

q.pop();

}

while (!t.empty())

{

s = t.front();

q.push(s);

t.pop();

}

}

}

}

while (!q.empty())

q.pop();

while (!t.empty())

t.pop();

}

return 0;

}

B: ModricWang的序列修改

此题是基础的线性表元素检索和删除操作，由于数据较小，用顺序表或链表都可以完成。

参考代码：

#include <iostream>

#include <list>

#include <ctime>

using namespace std;

int n, k, now;

list<int> l;

list<int>::iterator it;

int main() {

l.clear();

cin >> n >> k;

for (int i = 0; i < n; i++) {

cin >> now;

l.push\_back(now);

}

it = l.begin();

for (int i = 1; i < k; i++) {

it++;

}

l.erase(it);

for (it = l.begin(); it != l.end(); it++) {

cout << \*it << " ";

}

cout << "\n";

}

C:Mdd的数字

这道题求是两个数字之间的汉明距离，其实就是这两个数对应位上不同的个数。比如1（0…01）与-1（1…1）就有31位不同（int共有32位）

参考代码:

#include <iostream>

#include <fstream>

#define in std::cin

#define out std::cout

using namespace std;

int hammingDistance(int x, int y) {

int ans = x ^ y;

int cnt = 0;

while (ans) {

cnt++;

ans &= ans - 1;

}

return cnt;

}

int main() {

int x, y;

while (in >> x >> y) {

out << hammingDistance(x, y) << '\n';

}

return 0;

}

D：

思路：

简单思考可知，只需把原数按字符串读入后正着输出再反着输出就好了。

代码

#include <cstdio>

#include <string>

#include <iostream>

using namespace std;

int main()

{

string s,s2;

while(cin>>s)

{

cout<<s;

s2="";

for(int i=s.length()-1;i>=0;i--)

s2+=s[i];

//reverse(s.begin(),s.end());

cout<<s2<<endl;

}

return 0;

}

E

思路：

字符串的应用和处理。可以把字符对应成数字这样便于处理。

如果仔细观察这一串数字，可以发现他可以还分为很多小串，假设第i小串是123……i，假设第i小串所占的空间是a[i]，则通过对比a[i]与a[i+1]发现，

第i+1串只比第i串多一个数，即i+1，故他们所占的空间差就是第i+1所占的空间。

对任意一个数所占的空间很好求，即 （int）log10（k）+1；

然后就可以求出每一个串的起始位置，通过与n比较就可以确定n出现在那一个串里，最后在求出n在这个串里的相对位置，就可以求出该题的解。

代码：

#include <cstdio>

#include <cmath>

#define max 31270

using namespace std;

unsigned int a[max];

unsigned int s[max];

void start()

{

int i, j;

a[1] = 1;

s[1] = 1;

for (i = 2; i<31270; i++) //打表记录总的位数

{

a[i] = a[i - 1] + (int)log10((double)i) + 1;

s[i] = s[i - 1] + a[i];

}

}

int res(int n)//预处理，不断缩小范围

{

int i = 1, j = 1;

int m = 0;

int len = 0;

for (i = 1; i<31270; i++)

{

if (s[i] >= n)

break;

}

int pos = n - s[i - 1];

for (j = 1; len<pos; j++)

{

len += (int)log10((double)j) + 1;

}

return ((j - 1) / (int)pow(10.0, len - pos)) % 10;

}

char ha(int n)

{

if (n == 0)

return '\*';

else if (n == 1)

return 'a';

else if (n == 2)

return 'b';

else if (n == 3)

return 'c';

else if (n == 4)

return 'd';

else if (n == 5)

return 'e';

else if (n == 6)

return 'f';

else if (n == 7)

return 'g';

else if (n == 8)

return 'h';

else if (n == 9)

return 'i';

}

int main()

{

int t;

int test;

char a;

scanf("%d", &t);

start();

while (t--)

{

scanf("%d", &test);

a = ha(res(test));

printf("%c\n", a);

}

return 0;

}

F

思路：

稳定婚姻问题。

这里用的是Gale-Shapley算法。

设软院班级为a，女院班级为b。

每一位a在所有尚未拒绝她的b中选择一位被他排名最优先的b；

每一位b将正在追求她的a与其当前联谊的a进行比较，选择其中排名优先的a作为其联谊对象。

若某a被其联谊对象抛弃，重新变成无联谊对象的a。

代码：

#include <iostream>

#include <cstdio>

#include <cstring>

#include <string>

#include <algorithm>

#include <queue>

#include <vector>

#include <cmath>

#include <cstdlib>

using namespace std;

const int maxn = 30;

int order[maxn][maxn], pre[maxn][maxn], next1[maxn];

int future\_h[maxn], future\_w[maxn], hash1[100];

queue<int>Q;

int n;

void engage(int man, int woman)

{

int m = future\_h[woman];

if (m)

{

future\_w[m] = 0;

Q.push(m);

}

future\_h[woman] = man;

future\_w[man] = woman;

}

int main()

{

int t, cnt;

char s[50], female[maxn];

scanf("%d", &t);

while (t--)

{

memset(hash1, -1, sizeof(hash1));

scanf("%d", &n);

cnt = 0;

for (int i = 1; i <= n; i++)

{

scanf("%s", s);

hash1[s[0] - 'A'] = ++cnt;

}

cnt = 0;

for (int i = 1; i <= n; i++)

{

scanf("%s", s);

hash1[s[0] - 'A'] = ++cnt;

female[cnt] = s[0];

}

for (int i = 1; i <= n; i++)

{

scanf("%s", s);

int u = hash1[s[0] - 'A'];

cnt = 0;

for (int j = 2; j<n + 2; j++)

pre[u][++cnt] = hash1[s[j] - 'A'];

next1[u] = 1;

future\_w[u] = 0;

Q.push(u);

}

for (int i = 1; i <= n; i++)

{

scanf("%s", s);

int u = hash1[s[0] - 'A'];

cnt = 0;

for (int j = 2; j<n + 2; j++)

{

int v = hash1[s[j] - 'A'];

order[u][v] = ++cnt;

}

future\_h[u] = 0;

}

while (!Q.empty())

{

int man = Q.front();

Q.pop();

int woman = pre[man][next1[man]++];

if (!future\_h[woman]) engage(man, woman);

else if (order[woman][man]<order[woman][future\_h[woman]]) engage(man, woman);

else Q.push(man);

}

while (!Q.empty()) Q.pop();

for (int i = 32; i <= 60; i++)

if (hash1[i] != -1)

printf("%c %c\n", 'A' + i, female[future\_w[hash1[i]]]);

if (t) puts("");

}

return 0;

}

G：

思路：

首先我们可以找到原串的答案是多少，考虑进行操作，最多也就是增加2个（在至少出现2次相邻位相同的情况下达到），如果只出现1次相邻位相同最多也就是增加1个，如果相邻位都是不同的，显然最好情况也就是全反转后保持答案不变。所以我们只需要统计相邻两位有多少是相同的，有多少是不同的就行了。

代码

#include <cstdio>

#include <algorithm>

using namespace std;

char s[100001];

int main()

{

int n,same,dif;

while(scanf("%d",&n)!=EOF)

{

scanf("%s",s);

same=0;

dif=0;

for(int i=0;i<n-1;i++)

if(s[i]==s[i+1])

same++;

else dif++;

printf("%d\n",dif+1+min(same,2));

}

return 0;

}