**复试上机指导概述**

1. 本文档仅供研友之间学习讨论使用，请勿用于商业用途。
2. 本文档所提供的源代码均整理自网络，不代表录入者本人的任何观点或立场。
3. 真题只是提供辅助作用，关键还是研友平时关于动手能力的练习和对于算法、数据结构的理解。参加过ACM的同学有一定优势。
4. 上机时，可以快速阅读所有的题目，按照从易到难的次序做题，保证会的一定得分。
5. 熟悉编程环境，熟悉c/cpp的常用函数；
6. 为了快速测试代码的正确性，尤其是矩阵输入情况，可以利用标准输入重定向，freopen(“c:\\input.txt”, “r”, stdin);加快测试过程；
7. 注意程序边界条件的测试。

By 冷阳

**2017年上级试题及参考答案**

1. 求中位数的位置

先输入一个整型数字N，接着输入n个无序的数字。要求输入升序排列后的中位数，以及该中位数输入的次序。如果N为偶数，则输出有两个中位数；如果N为奇数，输出最中间的数字即可。

Example1:

Input: 5

9 2 7 1 6

Output:

6 5

Example2:

Input：6

9 6 7 1 2 3

Output:

3 6

6 2

示例程序(CodeBlocks 13.12编译调试成功)：

#include <iostream>

#include <cstring>

#include <algorithm>

#include <vector>

using namespace std;

typedef struct number

{

int num;

int pos;

}Number;

bool cmp(Number a, Number b)

{

return a.num < b.num;

}

int main()

{

int n;

cin >> n;

vector<Number> array(n);

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

{

cin >> array[i].num;

array[i].pos = i + 1;

}

sort(array.begin(), array.end(), cmp);

if(n & 2)

{

cout << array[n / 2 - 1].num << " " << array[n / 2 - 1].pos << endl;

}

cout << array[n / 2].num << " " << array[n / 2].pos << endl;

return 0;

}

1. 查找未定义变量

输入二个C语言语句u，第一句为正常的C语言变量定义语句，符合C语言语法要求，变量间可以有多个空格，包括数组、指针定义等。第二句为变量运算语句，要求输出第二个C语言语句中未定义的变量。

Example：

int x12,y=1,num\_stu=89,a[30],\*p;

Sum=num+x12\*y;

Output:

Sum num

示例程序（CodeBlocks 13.12编译调试成功）：

#include <iostream>

#include <cstring>

#include <cstdlib>

using namespace std;

int main()

{

char s[1000];

char vars[100][10];

int var\_index = 0;

char current\_var[20] = "";

bool recording\_variable = true;

bool recorded = false;

for(int r = 0; r < 2; ++r)

{

cin.getline(s, 1000);

if(r == 1)

{

recording\_variable = true;

recorded = false;

memset(current\_var, 0, sizeof(current\_var));

}

for(int i = 0; i < strlen(s); ++i)

{

if(r == 0 && s[i] == '\*')

{

continue;

}else if(recording\_variable && ('A' <= s[i] && s[i] <= 'Z' || 'a' <= s[i] && s[i] <= 'z' || '0' <= s[i] && s[i] <= '9' || s[i] == '\_'))

{

current\_var[strlen(current\_var)] = s[i];

recorded = true;

}else if(s[i] == '=' || s[i] == ';' || s[i] == '[')

{

if(recording\_variable)

{

if(r == 0)

{

strcpy(vars[var\_index++], current\_var);

}else

{

bool flag = false;

for(int j = 0; j < var\_index; ++j)

{

if(strcmp(vars[j], current\_var) == 0)

{

flag = true;

break;

}

}

if(!flag)

{

cout << current\_var << ' ';

}

}

memset(current\_var, 0, sizeof(current\_var));

recorded = false;

}

if(r == 0)

{

recording\_variable = false;

}

}else if(s[i] == ' ')

{

if(recording\_variable && recorded)

{

if(r == 0)

{

strcpy(vars[var\_index++], current\_var);

}else

{

bool flag = false;

for(int j = 0; j < var\_index; ++j)

{

if(strcmp(vars[j], current\_var) == 0)

{

flag = true;

break;

}

}

if(!flag)

{

cout << current\_var << ' ';

}

}

memset(current\_var, 0, sizeof(current\_var));

recorded = false;

}

}else if(s[i] == ',' || s[i] == '+' || s[i] == '-' || s[i] == '\*' || s[i] == '/')

{

if(recording\_variable)

{

if(r == 0)

{

strcpy(vars[var\_index++], current\_var);

}else

{

bool flag = false;

for(int j = 0; j < var\_index; ++j)

{

if(strcmp(vars[j], current\_var) == 0)

{

flag = true;

break;

}

}

if(!flag)

{

cout << current\_var << ' ';

}

}

memset(current\_var, 0, sizeof(current\_var));

recorded = false;

}

recording\_variable = true;

}

}

if(r == 0)

{

for(int i = 0; i < var\_index; ++i)

{

char s\_int[] = "int";

if(strcmp(vars[var\_index], s\_int) == 0)

{

memset(vars[var\_index], 0, sizeof(vars[var\_index]));

}

cout << vars[i] << endl;

}

}

}

return 0;

}

1. 找家谱成员

输入若干行，每一行的第一个输入为家谱中的某成员，该行接着输入的信息为每个孩子姓名。

最后一行的输入为要求查找的二个家谱成员的关系。

要求，根据输入的家谱成员信息，建立二叉树家谱关系图，并输入二位待查成员在家谱中的关系，包括输出他们最近邻的共同祖先以及在家谱中相差的层次数。

Input:

Ye Shu Ba

Shu Ge Mei1

Ba Self Mei2

Ge Son1 Son2

Son2 Mei1

Output:

Shu 1

示例程序（CodeBlocks 13.12编译调试成功）：

#include <iostream>

#include <cstdlib>

#include <cstring>

#include <cmath>

#include <algorithm>

#include <vector>

using namespace std;

typedef struct node

{

char name[20];

int depth;

struct node\* father;

}Node;

int main()

{

char s[1000];

int lineno = 0;

Node nodes[100];

memset(nodes, 0, sizeof(nodes));

int node\_index = 0;

while(cin.getline(s, 100))

{

char names[3][20];

int name\_index = 0, pos\_index = 0;

memset(names, 0, sizeof(names));

bool last\_is\_space = false;

for(int i = 0;i < strlen(s); ++i)

{

if(s[i] == ' ')

{

name\_index += 1;

pos\_index = 0;

last\_is\_space = true;

}else

{

names[name\_index][pos\_index++] = s[i];

last\_is\_space = false;

}

}

name\_index += 1;

if(name\_index == 3)

{

if(lineno == 0)

{

strcpy(nodes[node\_index].name, names[0]);

nodes[node\_index].father = NULL;

nodes[node\_index].depth = 0;

node\_index++;

}

for(int i = 0; i < node\_index; ++i)

{

if(strcmp(names[0], nodes[i].name) == 0)

{

for(int j = 1; j < name\_index; ++j)

{

strcpy(nodes[node\_index].name, names[j]);

nodes[node\_index].father = &nodes[i];

nodes[node\_index].depth = nodes[i].depth + 1;

node\_index++;

}

}

}

}else

{

Node \*p0, \*p1;

for(int i = 0; i < node\_index; ++i)

{

if(strcmp(names[0], nodes[i].name) == 0)

{

p0 = &nodes[i];

}

if(strcmp(names[1], nodes[i].name) == 0)

{

p1 = &nodes[i];

}

}

int d\_depth = p0->depth - p1->depth;

while(p0->depth > p1->depth)

{

p0 = p0->father;

}

while(p1->depth > p0->depth)

{

p1 = p1->father;

}

while(p0 != p1)

{

p0 = p0->father;

p1 = p1->father;

}

cout << p0->name << ' ' << abs(d\_depth) << endl;

}

lineno++;

}

return 0;

}

**2016年上机试题及答案**

1. 给定一个数n，将这个数的各位顺序颠倒，称为逆序数m。

例如1234的逆序数是4321

如果m是n的整数倍，那么输出n\*k=m

例如输入1089，输出1089\*9=9801

如果m不是n的整数倍，那么输出n和n的逆序数

例如输入1234，输出1234 4321

再例输入23200，输出23200 00232

已知输入开头不包括多余的0.

示例程序（CodeBlocks 13.12未成功运行，但代码本身并无逻辑错误。结果并非理想结果原因是math.h对于pow函数定义差异。）：

#include <stdio.h>

#include <math.h>

int getNUMDes(int num)

{

int i = 0;

while((num)/(int)pow(10,i))

{

i++;

}

return i;

}

int getReserveNum(int num, int decs)

{

int reverseNum = 0;

int i = 0;

while(decs)

{

reverseNum = reverseNum + (num / (int)pow(10, decs - 1)) \* (int)pow(10, i);

//printf("\n%d %d %d %d %d\n", reverseNum, i, decs, num, (int)pow(10, decs - 1));

num = num % (int)pow(10, decs - 1);

decs--;

i++;

}

return reverseNum;

}

int main()

{

int num;

scanf("%d", &num);

int decs = getNUMDes(num);

int reverseNum = getReserveNum(num, decs);

if(0 == (reverseNum % num))

{

printf("%d\*%d=%d", num, reverseNum / num, reverseNum);

}else

{

printf("%d %d", num, reverseNum);

}

getchar();

getchar();

return 0;

}

二，给一个C语言的enum定义语句，输出enum中规定的各项值。

Example:

Input:

Enum BOOL{true, false};

Output:

true0

false1

示例程序（CodeBlocks 13.12编译调试成功）：

#include <stdio.h>

#include <string.h>

#include <malloc.h>

#include <stdlib.h>

#define INVALID 9999

const int LEN = 200;

char name[LEN][LEN];

int num[LEN];

int cur\_Index = 0;

void parse(char enumStr[], int len);

char \*TypeContent = (char \*)malloc(LEN);

int TypeContentLen = 0;

int lastPos = 0;

char eval(char c);

char \*getTypeContent(char enumStr[], int len);

void scan(char enumStr[], int &pos);

int count\_p(int num, int &preCount);

void print();

void parse(char enumStr[], int len)

{

char \*content = getTypeContent(enumStr, len);

int BEG = 0;

while(lastPos < TypeContentLen)

{

scan(content, BEG);

}

}

char \*getTypeContent(char enumStr[], int len)

{

int i = 0;

int pos = 0;

while(true)

{

if(enumStr[i] == '{')

{

while(i++)

{

if(enumStr[i] == '}')

{

TypeContent[pos++] = '\0';

TypeContentLen = pos;

return TypeContent;

}else

{

TypeContent[pos++] = enumStr[i];

}

}

}

i++;

}

return NULL;

}

char eval(char c)

{

return c - '0';

}

void scan(char enumStr[], int &pos)

{

int end\_p = pos;

int i = 0;

while(true)

{

if(enumStr[end\_p] == ',')

{

name[cur\_Index][i] = 0;

num[cur\_Index] = INVALID;

pos = end\_p + 1;

cur\_Index++;

return;

}

if(enumStr[end\_p] == '=')

{

name[cur\_Index][i] = 0;

while(true)

{

end\_p++;

if(enumStr[end\_p] != ' ')

{

num[cur\_Index] = atoi(&enumStr[end\_p]);

break;

}

}

while(true)

{

if(end\_p >= TypeContentLen)

{

lastPos = TypeContentLen;

return;

}

end\_p++;

if(enumStr[end\_p] == ' '){break;}

}

pos = end\_p++;

cur\_Index++;

return;

}

name[cur\_Index][i++] = enumStr[end\_p++];

}

}

int count\_p(int num, int &preCount)

{

if(num != INVALID)

{

preCount = num;

return preCount;

}else

{

preCount = preCount + 1;

return preCount;

}

}

void print()

{

int i = 0;

int preCount = 0;

for(;i <= cur\_Index; i++)

{

printf("%s:%d\n", name[i],count\_p(num[i], preCount));

}

}

int main()

{

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

{

num[i] = INVALID;

}

char \*enumStr = "enumdate{JAN=1, FEB, MAR, APR, MAY, JUN, JULY, AUG, SEP, OCT, NOV, DEC, MON=1, TUE, WED, THU, FRI, SAT, SUN, found=1949};";

parse(enumStr, strlen(enumStr));

print();

getchar();

getchar();

return 0;

}