四川大学2002年攻读硕士学位研究生入学考试试题 及 答案

1.3.1 第2002年——C语言题目

C语言程序设计（共30分）

一、单项选择题（在每小题的四个备选答案中，选出一个正确答案。每小题1分，共6分）

1、如果I为整型变量，f为float型变量，c表达式'a'+I*f的类型为_____。

A、字符型　　B、整型　　C、单精度型　　D、双精度型

//即使是两个float型的数据相加，都化成double型 float型的数据在运算时一律转化成双精度型，提高其运算精度

2、关于c表达式5||(a=b+c)==3的值，正确的说法为_____。

A、值为1　　B、值为5　　C、值为0

D、仅这一个表达式，不能确定值，必须知道a,b,c的值后才能计算。

//等号==的优先级高于||所以先计算(a=b+c)==3但是不论它为真或假，在与5进行或运算时，按照或运算的法则，只要一方为真，表达式的结果为真。

3、若整型变量a的值为50，则c表达式a＞＞2的值为_____。

A、50　　B、25　　C、12.5　　D、12

//每右移一位除以2，且由于a为整型，结果为整型

4、若c,c1,c2均为字符型变量，下列语句执行后。

```
c='a';
c1=++c;
c2=c++;
```

c1,c2的值分别为_____。

A、c1='b',c2='c'　　B、c1='a', c2='c'　　C、c1='b',c2='b'　　D、c1='a',c2='b'

//前加加，先加1后使用值，后加加是先使用其值在加加，所以c加加后的值赋予c1，c1得值b，c2先使用值，再加加，c2得值b

5、以下c 语句执行后，

```
    int i,    s=0;
for (i=0;i＜10;i++)s+=i;
printf    ('%d\n', i);
```

输出的内容为_____。

A、0　　B、9　　C、10　　D、11

//i 值在循环外定义，所以其值循环结束仍有意义，得值 10

6、以下 c 语句执行后，

char s1[100] =″ I like C 1anguage″ ,s2[100];

strcpy(s2, s1);

strcat (s1, s2);

puts (s2);

输出的内容为_____。

A、I　　B、I like C language　　C、I like C language I like C language

D、I like C language I like C language I like C language

//连接后的串存在 s1 里，对 s2 没有影响

二、阅读程序，写出该程序的执行结果。（4 分）

include 〈stdio.h〉

void main ( )

{ char a [3] [10]= {"Beijing ","Shanghai", "Chengdu"} ;

char p1, (*p2)[10];

p1=a[0];

p2=a;

printf("%c\n",*(a[1]+1));

// a[1]代表是第一行字符串 Shanghai，加 1 后得值第一个字符 h 的地址，*取出其地址里的值 h

printf("%c\n",*(*(a+1)+2));

// *(a+i)=a[i] 代表是第一行字符串 Shanghai，加 2 后得值第 2 个字符 a 的地址，*取出其地址里的值

printf("%c\n",*(p1+1));

//p1 是一字符指针，加一指向第零行第 1 个字符 e 的位置，*取出其地址里的值

printf("%c\n",*(*(p2+1)+2));

//p2 为一指向数组的指针，p2 是与数组名等价的指针，它指向 2 维数组的首行，p2＋1 指针指向了第一行字串 Shanghai，加 2 后得值第 2 个字符 a 的地址，*取出其地址里的值

printf("%s\n",*(p1+1));

//打印的从第一个字符开始的第零行余下的字符串 eijing

printf("%s\n",*(p2[2]);

//p2[2]指的是第二行，打印第二行字符串 Chengdu

printf("%s\n",*(p2＋1));

//p2+1 指向的是第一行字符串 shanghai

三、按要求编写函数。（8 分）

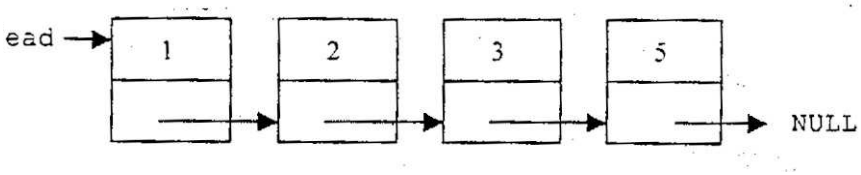
在一程序中定义了如下结构类型用于处理单向链表：

```
struct linkage  {
      int value;
      struct linkage *pNext;
} ;
```

现已经编写出按节点的成员 a 值从小到大建立初始链表的函数 create ( ),如下所示：

```
    struct linkage * create (void )
 {
      struct linkage * pHead;
int a[4]= {2,1,3,5} ;
pHead=NULL;
for （i=0;i＜4;i++）
pHead=addItem(pHead,a[i]);
return pHead;
}
```

在该函数的 for 循环前，pHead 为 NULL；在执行整个 for 循环后，创建的链表如下所示：



即在被调函数 addItem( )中，需要动态分配内存，生成新节点，将传入的第二参数的值赋给该节点的成员 value；并通过第一个参数链表首节点确定适当位置放置该新节点。

请完成函数：

```
struct linkage * addItem (struct linkage *pHead, int value );
```

返回值指向单向链表头节点。

你可能用到下面的动态内存分配函数。

void* malloc (unsigned int size);其作用是在内存的动态存储区中分配一个长

度为 size 的连续空间。返回值为指向分配域起始地址的指针。

1、在空链表中加第一个节点 2 分；
2、加到当前链表头节点前 2 分；
3、加到当前链表头节之后适当位置，可分为两步。2 分；
4、其它部分 2 分，包括整体结构，变量定义，返回值。

```
struct linkage *addItem (struct linkgae *pHead ,int value)
 {      struct linkage *p ,*q,*r,*s;
   p=pHead;
     if     (!p)
 {    r=(struct linkage *) malloc (sizeof (struct linkage));
     r-＞a=value;
     r-＞pNext =NULL;
     pHead=r;
}
else if (value  ＜=p-＞a)
 {    r=(struct linkage *) malloc (sizeof (struct linkage));
     r-＞a=value;
     r-＞pNext =p;
     pHead=r;
else
 {      q=p＞pNext;
while (q)
if (value＜=q-＞a)
break;
else
 {      p=q;
          q=p-＞pNext;
}
r=(struct linkage *)malloc (sizeof (struct linkage));
r-＞a=value;
r-＞pNext=q;
p-＞pNext=r;
}
```

return pHead;
}

四、程序设计。（12 分）

单位 HTML 文件由标签，标题和正文主体等部分组成。如下所示：

〈HTML〉
〈HEAD〉
〈TITLE〉HTML 标题部分</TITLE>
〈/HEAD〉
<BODY>
这里是 HTML 文件的主体部分。
换新行。
</BODY>
〈/HTML〉

符号"<与">"及其之间的内容是标签，如〈HTML〉，</TITLE>等。其它部分是标题和主体。请按以下说明和要求完成程序，将简单的 HTML 文件转换为 TXT 文件：

1. 序将简单 HTML 文件的有标签和换行符（\n）（去掉。标题部分结束（遇到</TITLE>标签）后加上一个换行符（\n）。每一个
标签换成一个换行符（\n）。其余部分原样保留。如上面的简单 HTML 文件处理为（↙表示换行）：

HTML 标题部分↙
这里是 HTML 文件的主体部分。↙
换新行。

2．简单 HTML 文件的标签内没有嵌套情况，即在"<和">"这间不会再出现"<或">"。

3．标签内的内容大小写无关。如
,
,
等价。

4．示签的"<和">"与标签内容间无空格。即不会出现
或< BR>等情况。

5．标题和正文中不会出现"<和">"字符，它们通过转义字符实现，这时尤需考虑。

6．程序需要处理命令行参数。第一个参数指要处理的 HTML 文件名，第二个参数指定处理后要存放的文件名。这两个文件都是文本文件。

7．程序必须进行必要的出错处理，如无法打开文件等。

8．程序不需要包含头文件。

1、main（）函数形参处理：2 分。
2、变量定义 1 分。
3、打开关闭文件 2 分；
4、读源文件 1 分；
5、能从标签中找出内容〈BR〉〈/TITLE〉换为'\n'部分 2 分；
6、区分开标签和其它部分，并按要求写入目标文件 2 分；
7、整体结构正确 2 分；

```
void main (int argc, char *argv[])
 {    char C,tag[7];
int status ,len ;
FILE*fin,*fout;
if (argc!=3)
 {     printf("参数个数不对！ ")
exit (1)
}
if ((fin=fopen (argv [1],"r")))==NULL)
 {
      printf(“打不开源文件”);
      exit(1);
  }
 if ((fout=fopen (argv [2],"w")))==NULL)
 {
    printf("无法写入文件！ ");
      fclose (fin);
      exit (1);
  }

status =0;
while   ((c=fgetc (fin)))!=EOF)
    switch (C)
 {
case '＜':
status=1;
```

```
    len=0;
    break;
    case '>':
    status=0;
    if(len<=6)
    tag[len]=0;
    if(!strcmp(tag,"/title")||!strcmp(tag,"br"))
       fputc('\n',fout);
    break;
    case '\n':break;
    default:
    if (! status)
    fputc (c,fout);
    else
     {    len++;
    if (len＜=6);
    tag[len-1]=c;
    }
    }
    fclose (fin);
    fclose (fout);
}
```