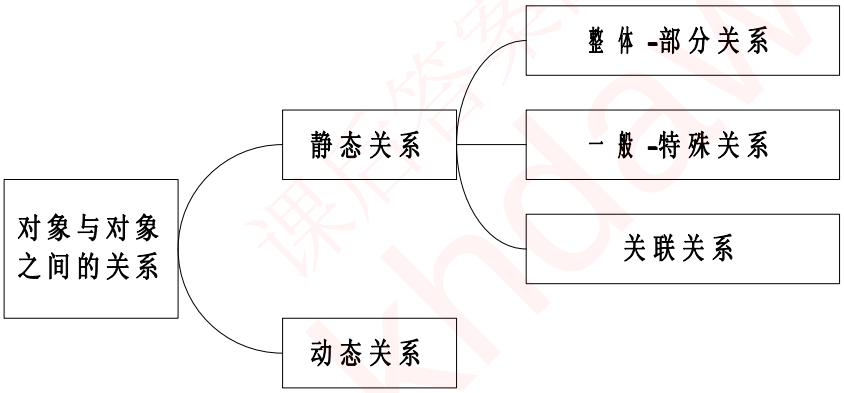
第 1 章习题解答

1．面向过程的程序设计方法的基本观点是什么？
答：面向过程的程序设计方法的基本观点是：以过程或函数作为构建系统的基本单元。

2．面向对象的程序设计方法的基本观点是什么？
答：面向对象的程序设计方法的基本观点是：以对象作为构建系统的基本单元。

3．面向过程的程序设计方法的缺点是什么，面向对象的程序设计方法如何解决这些缺点？
答：面向过程的程序设计方法的缺点是：以过程或函数作为构建系统的基本单元，造成了数据和功能的割裂，而且，由于模块的粒度较小，使得无法适应当今大规模复杂的应用程序开发要求。为了解决 C 语言的局限性，更是为了适应当今大规模程序开发的复杂性要求，开发了面向对象的程序设计方法，即将数据和功能统一考虑，封装在称为“类”的模块中，避免了数据和功能的割裂，也符合人们的思维习惯。

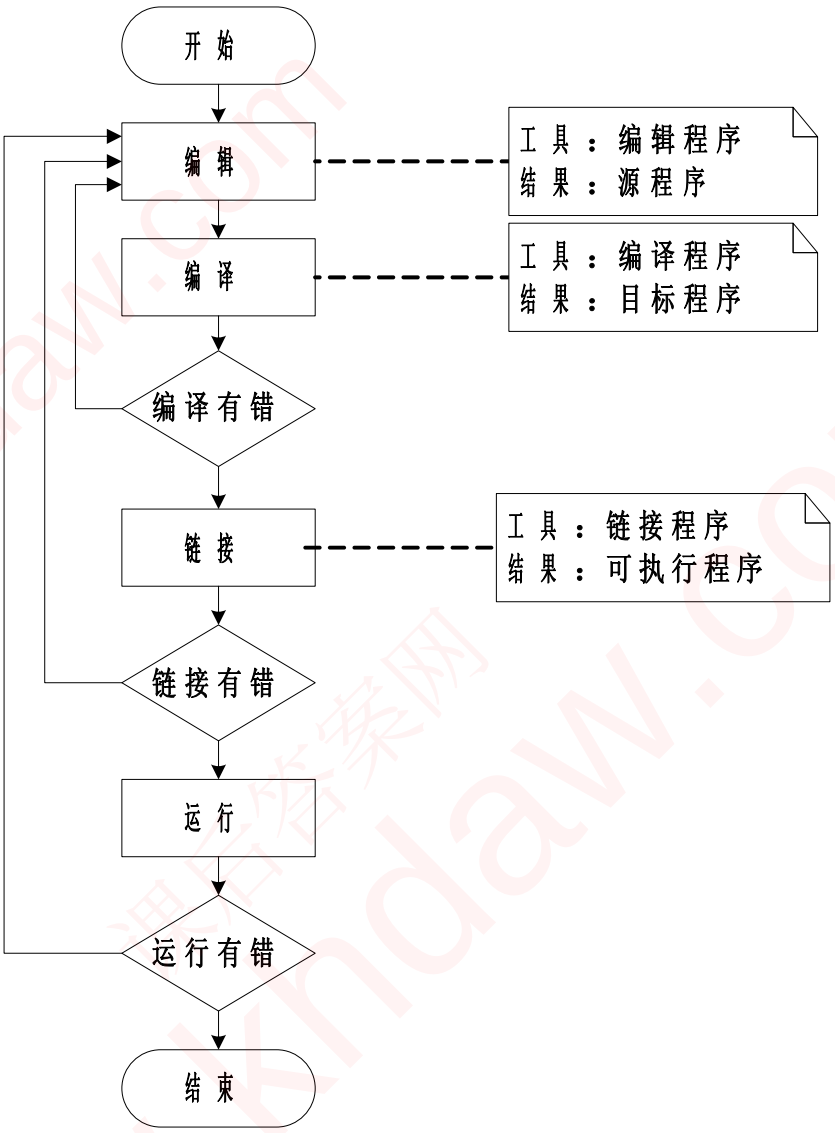
4．对象之间的关系有哪些？
答：对象之间的关系如下图所示。



5．C++语言与 C 语言有何不同？
答：C++语言全面支持并兼容 C 语言，保持了 C 语言简洁高效等很多优点，而且比 C 语言更安全（C++语言引入了强类型检查机制），更为重要的是全面支持面向对象的程序设计方法。C 和 C++语言的关系可以用公式“C++=C+面向对象”来进行简单概括。C++语言对 C 语言的增强表现在两个方面：（1）增加了面向对象的机制；（2）在原来面向过程的基础上，对 C 语言的功能做了很多扩充，如强类型检查机制和引用数据类型等。

6．一个简单的 C++程序应包括哪几个部分？
答：一般来说，完整的 C++程序包括：注释、编译预处理命令、主函数 main 和功能语句等。

7．使用 C++语言开发程序的具体步骤包括哪些？
答：使用 C++语言开发程序的具体步骤，如下图所示。



8．以一个简单的日常生活中的小软件系统为例，试画出基于面向过程程序设计方法的系统结构和面向对象程序设计方法的系统结构。

答：参考课本相关内容。

一、单项选择题

BDAAC　DBACD

二、填空题

1. 下划线
2. 23，24
3. char ch='8';
   或 char ch=56;
   或 char ch=0x38;
   或 char ch=070;
4. double
5. 4
6. 0
7. 3
8. 15
9. 1
10. 2

三、读程序，写结果

1. i=7j=8m=14
2. n=1
3. 7
4. y=6x=6
5. i=6j=8x=6y=7

一、单项选择题

BDDBC　CCBC

例

```
#include <iostream>
using namespace std;
void main(){
    int i=0, n, k,result=0;
    cout<<"请输入一个任意整数：";
    cin>>n;
    k=n;
    while(k)    //判断数据的位数
    {   k/=10;i++;}
    cout<<n<<"是"<<i<<"位数\n";
    cout<<"按位翻转的结果为：";
    while (n!=0){
        i =n % 10;
        result=result*10+i;
            n /= 10;
    }
    cout<<result<<endl;
}
```

例：输入一行字符，分别统计其中的英文字母、数字、空格和其它字符的个数。

```
#include <iostream>
using namespace std;
void main()
{
    int space=0,other=0,letter=0,digit=0;
    char m;
    while((m=getchar())!='\n')
    {   if(m>='0'&&m<='9') digit++;
        else if(m==' ') space++;
        else if(m>='A'&&m<='Z'||m>='a'&&m<='z') letter++;
        else other++;
    }
    cout<<"数字："<<digit<<"个，字母："<<letter
        <<“个，空格：”<<space<<“个，其它：”
      <<other<<"个\n";
}
```

习题 4

一、单项选择题

1. 若有说明 int a[3][4];则 a 数组元素的非法引用是【　　】

A. a[0][2*1]

B. a[1][3]

C. a[4-2][0]

D. a[0][4]

【答案】 D

【解析】数组下标从 0 开始，a[0][4]的列下标越界。

2. 在 C++语言中，引用数组元素时，其数组下标的数据类型允许是【　　】

A. 整型常量

B. 整型表达式

C. 整型常量或整型表达式

D. 任何类型的表达式

【答案】 C

3. 以下不正确的定义语句是【　　】

A. double x[5]={2.0,4.0,6.0,8.0,10.0};

B. int y[5]={0,1,3,5,7,9};

C. char c1[]={'1','2','3','4','5'};

D. char c2[]={'\x10','\xa','\x8'};

【答案】 B

【解析】初始值的个数大于数组的大小，系统会出现编译错误。

4. 对以下说明语句的正确理解是【　　】

int a[10]={6,7,8,9,10};

A. 将 5 个初值依次赋给 a[1]至 a[5]

B. 将 5 个初值依次赋给 a[0]至 a[4]

C. 将 5 个初值依次赋给 a[6]至 a[10]

D. 因为数组长度与初值的个数不相同，所以此语句不正确

【答案】 B

5. 若有说明：int a[ ][4]={0,0};则下面不正确的叙述是【　　】

A. 数组 a 的每个元素都可得到初值 0

B. 二维数组 a 的第一维大小为 1

C. 当初值的个数能被第二维的常量表达式的值除尽时，所得商数就是第一维的大小

D. 只有元素 a[0][0]和 a[0][1]可得到初值，其余元素均得不到确定的初值

【答案】 D

【解析】二维数组初始化时，行大小可以省略，被省略的大小根据初值的个数系统来确定，本题中，有 2 个初值说明是 1 行 4 列，所以第一维为 1。元素 a[0][0]和 a[0][1]赋初值为 0，其余元素初值系统默认为 0。

6. 以下能对二维数组 c 进行正确的初始化的语句是【　　】

A. int c[3][]={{3},{3},{4}};

B. int c[][3]={{3},{3},{4}};

C. int c[3][2]={{3},{3},{4},{5}};

D. int c[][3]={{3},{},{3}};

【答案】 B

【解析】二维数组初始化时，行大小可以省略，列大小不可以省略，所以 A 答案错误。C 答案中初始值行数多于数组大小中的行大小，也是错误的。另外初始化时，初值之间不能有空位置，故 D 错误。

7. 以下不能对二维数组 a 进行正确初始化的语句是【 】

A. int a[2][3]={0};

B. int a[][3]={{1,2},{0}};

C. int a[2][3]={{1,2},{3,4},{5,6}};

D. int a[][3]={1,2,3,4,5,6};

【答案】 C

8. 阅读下面程序，则程序段的功能是【 】

```
#include<iostream>
using namespace std;
int main()
{
    int c[]={23,1,56,234,7,0,34},i,j,t;
    for(i=1;i<7;i++)
    {
        t=c[i];j=i-1;
        while(j>=0 && t>c[j])
        {c[j+1]=c[j];j--;}
        c[j+1]=t;
    }
    for(i=0;i<7;i++)
    cout<<c[i]<<'\t';
    putchar('\n');
    return 0;
}
```

A. 对数组元素的升序排列

B. 对数组元素的降序排列

C. 对数组元素的倒序排列

D. 对数组元素的随机排列

【答案】 B

【解析】每层外层 for 循环结束会增加对一个元素的排序，确定 c[0] ,c[1]的大小顺序，如图 4-1 所示。

c[0]	c[1]	c[2]	c[3]	c[4]	c[5]	c[6]
23	**1**	56	234	7	0	34
1<23						
23	**1**	56	234	7	0	34

图 4-1 第一次排序

第二次将 c[2]分别与前 2 两个元素比较，插入最前面，确定 c[0] ,c[1], c[2] 的大小顺序，如图 4-2 所示。

c[0]	c[1]	c[2]	c[3]	c[4]	c[5]	c[6]
23	**1**	56	234	7	0	34
	1<56					
23	**1**	**1**	234	7	0	34
	23<56					
23	**23**	**1**	234	7	0	34
	t=56					
56	**23**	**1**	234	7	0	34

图 4-2 第二次排序

以此类推，外层 6 次循环后，将无序的数组实现大到小的降序排列。

9. 下列选项中错误的说明语句是【 】

A. char a[]={′t′,′o′,′y′,′o′,′u′,′\0′};

B. char a[]={"toyou\0"};

C. char a[]="toyou\0";

D. char a[]=′toyou\0′;

【答案】 D

10. 下述对 C++语言字符数组的描述中错误的是【 】

A. 字符数组的下标从 0 开始

B. 字符数组中的字符串可以进行整体输入/输出

C. 可以在赋值语句中通过赋值运算符“=”对字符数组整体赋值

D. 字符数组可以存放字符串

【答案】 C

11. 以下二维数组 c 的定义形式正确的是【 】

A. int c[3][]

B. float c[3,4]

C. double c[3][4]

D. float c(3)(4)

【答案】 C

12. 已知：int c[3][4];则对数组元素引用正确的是【 】

A. c[1][4]

B. c[1.5][0]

C. c[1+0][0]

D. 以上表达都错误

【答案】 C

13. 若有以下语句，则正确的描述是【 】

char a[]="toyou";

char b[]={′t′,′o′,′y′,′o′,′u′};

A. a 数组和 b 数组的长度相同

B. a 数组长度小于 b 数组长度

C. a 数组长度大于 b 数组长度

D. a 数组等价于 b 数组

【答案】 C

【解析】 a 数组中存放的是字符串，数组大小为 6 个字节空间，分别存放't','o','y','o','u'和'\0'，b 数组的长度为 5 个字节空间，只存放't','o','y','o','u'，5 个字符。

二、填空题

1. 若有说明：int a[][3]={1,2,3,4,5,6,7};则 a 数组第一维的大小是_____。

【答案】 3

2. 设有数组定义： char array[]="China"; 则数组 array 所占的空间为_____个字节。

【答案】 6

3. 假定 int 类型变量占用两个字节，其有定义：int x[10]={0,2,4};，则数组 x 在内存中所占字节数是_____。

【答案】 20

4. 下面程序的功能是输出数组 s 中最大元素的下标，请填空。

```
#include<iostream>
using namespace std;
int main()
{ int k, p,s[]={1, -9, 7, 2, -10, 3};
for(p =0, k =p; p< 6; p++)
if(s[p]>s[k])  _____
cout<< k<<endl;
return 0;}
```

【答案】 k=p;

5. 下面程序是删除输入的字符串中字符'H'，请填空。

```
#include<iostream>
using namespace std;
int main()
{ char s[80];
int i,j;
gets(s);
for(i=j=0;s[i]!='\0';i++)
if(s[i]!='H')
   {_____}
s[j]='\0';
puts(s);
return 0;}
```

【答案】 s[j++] =s[i];

【解析】 此处相当于补充了 2 条语句，等价于 s[j] =s[i]; j++;。

6. 已知：char a[20]= "abc",b[20]= "defghi";则执行 cout<<strlen(strcpy(a,b));语句后的输出结果为_____。

【答案】 6

7. 有如下定义语句：int aa[][3]={12,23,34,4,5,6,78,89,45};,则 45 在数组 aa 中的行列坐标各为_____。

【答案】 2 和 2

8. 若二维数组 a 有 m 列，则计算任一元素 a[i][j]在数组中相对位置的公式为（假设 a[0][0]

位于数组的第一个位置上） _____。

【答案】 i*m+j+1

9. 定义如下变量和数组：

```
int k;
int a[3][3]={9,8,7,6,5,4,3,2,1};
```

则语句 for(k=0;k<3;k++) cout<<a[k][k];的输出结果是_____。

【答案】 951

【解析】 定义的二维数组可以描述一个方阵：

9 8 7
6 5 4
3 2 1

语句功能是实现该方阵主对角线上元素的输出。

10. 已知：char a[15],b[15]={"I love china"};则在程序中能将字符串 I love china 赋给数组 a 的语句是_____。

【答案】 strcpy(a,b);

三、读程序写结果

1. 程序代码如下

```
#include<iostream>
using namespace std;
int main()
{
    char arr[2][4];
    strcpy(arr[0],"you");
    strcpy(arr[1],"me");
    arr[0][3]='&';
    cout<<arr[0]<<endl;
    return 0;
}
```

【答案】 you&me

2. 程序代码如下：

```
#include<iostream>
using namespace std;
int main()
{
    char a[]={'a', 'b', 'c', 'd', 'e', 'f', 'g','h','\0'};
    int i,j;
    i=sizeof(a);
    j=strlen(a);
    cout<< i <<","<<j<<endl;
    return 0;
}
```

【答案】 9,8

3. 程序代码如下：

```
#include<iostream>
```

```
using namespace std;
int main()
{
    int i;
    int a[3][3]={1,2,3,4,5,6,7,8,9};
    for(i=0;i<3;i++)
        cout<<a[2-i][i];
    return 0;
}
```

【答案】 753

【解析】 程序功能是实现一个方阵次对角线上元素的输出。

4. 程序代码如下：

```
#include<iostream>
using namespace std;
int main()
{
    char a[30]="nice to meet you!";
    strcpy(a+strlen(a)/2,"you");
    cout<<a<<endl;
    return 0;
}
```

【答案】nice to you

5. 程序代码如下：

```
#include<iostream>
using namespace std;
int main()
{
    int k[30]={12,324,45,6,768,98,21,34,453,456};
    int count=0,i=0;
    while(k[i])
    {
        if(k[i]%2==0||k[i]%5==0)
        count++;
        i++;
    }
    cout<< count <<","<<i<<endl;
    return 0;
}
```

【答案】8,10

6. 程序代码如下：

```
#include<iostream>
using namespace std;
int main()
{
```

```
    char a[30],b[30];
    int k;
    gets(a);
    gets(b);
    k=strcmp(a,b);
    if(k>0)   puts(a);
    else if(k<0)   puts(b);
    return 0;
}
```

输入 love↙

China↙

输出结果是？

【答案】love

【解析】 strcmp(a,b)函数功能是比较 a 和 b 字符串的大小，比较是逐个字符的比较，比较方法是 ASCII 码值做减法，k='l'-'C'>，所以结果输出字符串 a。

四、编程题

1．编程实现功能：删去一维数组中所有相同的数，使之只剩一个。数组中的数已按由小到大的顺序排列，函数返回删除后数组中数据的个数。

例如，若一维数组中的数据是：

2 2 2 3 4 4 5 6 6 6 6 7 7 8 9 9 10 10 10

删除后，数组中的内容应该是：

2 3 4 5 6 7 8 9 10。

思路：fun 函数的 2 个形式参数，分别接收数组名，即数组的首地址，和数组中存放的原始数据的个数，fun 函数对原始数组按题目处理后，在主函数中输出的数组 a 就是处理后的数组，这里函数之间参数传递属于地址传递。

程序代码如下：

```
#include<iostream>
using namespace std;
const N=80;
int fun(int a[], int n)
{ int i,j=1;
   for(i=1;i<n;i++)
     if(a[j-1]!=a[i])       a[j++]=a[i];
    return j;
}
int main()
{ int a[N]={ 2,2,2,3,4,4,5,6,6,6,6,7,7,8,9,9,10,10,10}, i, n=19;
    cout<<"原始数组中的数据是："<<endl;
    for(i=0;i<19;i++)
       cout<<a[i]<<" ";
    n=fun(a,n);
    cout<<"\n 删除后，数组中的数据是："<<endl;
    for(i=0;i<n;i++)
        cout<<a[i]<<" ";
```

```
    cout<<endl;
    return 0;
}
```

2．编程实现功能：从键盘上输入若干个学生的成绩，当输入负数时表示输入结束，计算学生的平均成绩，并输出低于平均分的学生成绩。

思路：使用循环输入学生的成绩，注意循环条件有 2 个，分别是输入成绩个数大于数组的大小和输入成绩为负数时结束，同时记录输入成绩的个数 n。结束输入后，计算平均值和筛选小于 60 分的成绩并输出。

程序代码如下：

```
#include<iostream>
using namespace std;
int main( )
{ const N=800;
  float x[N],sum=0,ave,a;
  int n=0,i;
  cout<<"输入学生成绩："<<endl;
  cin>>a;
  while (a>=0 && n<N)
    { sum+=a;
      x[n]=a;
      n++;
      cin>>a;
    }
  cout<<"输入的"<<n<<"名学生成绩为：";
for (i=0; i<n; i++)
    cout<<x[i]<<" ";
cout<<endl;
ave=sum/n;
cout<<"平均分："<<ave<<endl;
cout<<"低于平均分的成绩：";
for (i=0; i<n; i++)
     if (x[i]<ave)   cout<<x[i]<<" ";
 cout<<endl;
 return 0;
}
```

3．编程实现功能：对从键盘上输入的两个字符串进行比较，然后输出两个字符串中第一个不相同字符的 ASCII 码值之差。例如：输入的两个字符串分别为 abcdefg 和 abceef，则输出为-1。

思路：题目要求实现的功能，相当于字符串处理函数 strcmp 的功能，即：

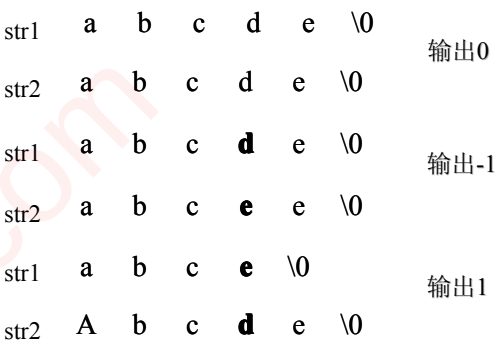


图 4-3 字符串比较示意图

使用循环逐个比较两个字符串的每个字符，当字符出现不相等时，跳出循求不相同的两个字符的 ASCII 码值的差输出。

程序代码如下：

```
#include<iostream>
using namespace std;
#include "string.h"
int main( )
{ char str1[80], str2[80],c;    int    i=0,s;
   gets (str1);
   gets (str2);
   while ((str1[i]==str2[i]) && (str1[i]!= '\0'))
         i++;
   s= str1[i]-str2[i];
   cout<<s<<endl;
   return 0;
}
```

4．编程实现功能：求二维数组周边元素之和。

思路：二维数组中的数据可以看成一个二维矩阵，例如下面的二维数组，输出周边元素之和为：sum=48，用两个并列的 for 循环实现求累加和。

```
1  2  3  4
2  3  4  5
3  4  5  6
4  5  6  7
```

程序代码如下：

```
#include<iostream>
using namespace std;
 #include "string.h"
 const M=4;
 const N=4;
 int main( )
 { int a[M][N],i,j,sum=0;
   for (i=0; i<M; i++)
      for (j=0; j<N; j++)
```

```
        cin>>a[i][j];
    for (i=0; i<N; i++)     /*求第 1 行和第 4 行元素和*/
      { sum+=a[0][i];
        sum+= a[M-1][i];
      }
    for (i=1; i<M-1; i++)     /*求第 1 列和第 4 列元素和，
                                但不包括 a[0][0], a[3][0], a[0][3], a[3][3]四个角上的元素*/
      { sum+=a[i][0];
        sum+= a[i][N-1];
      }
    cout<<"二维数组周边元素之和为："<<sum<<endl;
    return 0;
}
```

5. 编程求出 3 阶方阵的两条对角线上元素之和。

程序代码如下：

```
#include<iostream>
using namespace std;
int main()
{ int arr[3][3]={0,2,2,3,4,4,5,6,6},a=0,b=0,i,j;
   for(i=0;i<3;i++)
      for(j=0;j<3;j++)     /*求主对角线上元素和*/
        if(i==j)
          a=a+arr[i][j];
   for(i=0;i<3;i++)        /*求次对角线上元素和*/
       for(j=2;j>=0;j--)
         if(i+j==2 )
           b=b+ arr[i][j];
       cout<<"主对角线元素和为："<<a<<endl;
       cout<<"次对角线元素和为："<<b<<endl;
return 0;
}
```

6. 编程序求 Fibonacci 数列的前 10 项，并按每行 3 个数的格式输出该数列。Fibonacci 数列的定义为：

$$f_n=\begin{cases}1 & (n=1)\\ 1 & (n=2)\\ f_{n-1}+f_{n-2} & (n>2)\end{cases}$$

程序代码如下：

```
#include<iostream>
#include<iomanip>
using namespace std;
int main( )
```

```
{ int i;
   long f[10]={1,1};
   for (i=2; i<30; i++)
      f[i]=f[i-2]+f[i-1];
   for (i=0; i<10; i++)
      { if(i%3==0)    cout<<"\n";
         cout<<setw(12)<<f[i];
      }
   cout<<endl;
return 0;
}
```

习题 5

一. 单项选择题

1. 以下叙述不正确的是【　】

A. 一个 C++源程序可由一个或多个函数组成

B. 一个 C++源程序必须包含一个 main 函数

C. C++程序的基本组成单位是函数

D. 在 C++程序中，注释说明只能位于一条语句的后面

【答案】 D

【解析】在 C++程序中，注释说明可以位于程序中的位置，需要用“//”或“/*”和“*/”标记，前者注释其后一行，后者注释“/*”和“*/”之间的内容。

2. 在调用函数时，如果实参是简单的变量，它与对应形参之间的数据传递方式是【　】

A. 地址传递

B. 单向值传递

C. 由实参传形参，再由形参传实参

D. 传递方式由用户指定

【答案】 B

3.以下函数值的类型是【　】。

```
fun(float x)
{ float y;
   y=3*x-4 ;
   retun y ;
   }
```

A.不确定　　B.float　　C.void　　D.int

【答案】 D

【解析】C++中规定，函数数据类型省略式，系统默认为 int。

4.若有以下函数调用语句：fun(a,(x,y),fun(n+k,d,(a,b)));，在 fun 函数调用语句中实参的个数是【　】。

A.3　　B.4　　C.5　　D.6

【答案】 A

【解析】fun(a,(x,y),fun(n+k,d,(a,b)))，共有 a、(x,y)和 fun(n+k,d,(a,b))，3 个实际参数；fun(n+k,d,(a,b))也是有 n+k、d 和(a,b)，3 个参数。

5.以下对 C++语言函数的有关描述中，正确的是【　】。

A.在 C++语言中，调用函数时，只能把实参的值传送给形参，形参的值不能传送给实参

B. C++语言中的函数既可以嵌套定义又可以递归调用

C.函数必须有返回值，否则不能使用函数

D.C++程序中有调用关系的所有函数必须放在同一个源程序文件中

【答案】 A

6.以下叙述不正确的是【　】。

A.在不同的函数中可以使用相同名字的变量

B.函数中的形式参数是局部变量

C.在一个函数内定义的变量只在本函数范围内有效

D.在一个函数内的复合语句中定义的变量在本函数范围内有效

【答案】D

7.函数 fun 的功能是计算 x^n。主函数中已经正确定义 m、a、b 变量和赋值，实现计算：m=a^4+b^4-(a+b)^3 正确的调用语句为【　】。

A.m=fun(a^4)+fun(b^4)-fun((a+b)^3);

B.m=fun(a,b,a+b)

C.m=fun(a,4)+fun(b,4)-fun((a+b),3);

D.m=fun((a,4),(b,4),((a+b),3));

【答案】C

8.C++语言的编译系统对宏替换命令是【　】。

A.在程序运行时进行代换的

B.在程序连接时进行代换的

C.和源程序中其他 C++语言同时进行编译的

D.在对源程序中其他成分正式编译之前进行处理

【答案】D

9.以下关于宏的叙述正确的是【　】。

A.宏名必须用大写字母表示

B.宏定义必须位于源程序所有语句之前

C.宏替换没有数据类型限制

D.宏替换比函数调用耗费时间

【答案】C

10.C++语言中形参的默认存储类别是【　】。

A.自动（auto）　　B.静态（static）

C.寄存器（register）　D.外部（extern）

【答案】A

二．填空题

1. 在内存中，存储字符'x'要占用 1 个字节，存储字符串"X"要占用._____ 个字节。

【答案】 2

2. 在 C++语言中，char 型数据在内存中的存储形式是_____

【答案】 ASCII 码

3. 一个 C++源程序中至少应包含一个_____。

【答案】主函数

4. 设有如下函数

```
fun (float x)
{ cout<<"\n"<<x*x;
}
```

则函数的类型是_____

【答案】int 型

5.编写两个函数，分别求出两个整数的最大公约数和最小公倍数，用主函数调用这两个函数，并输出结果，两个整数由键盘输入。请填空完成程序功能。

```
#include <iostream>
using namespace std;
fmax(int m,int n)
```

```
{int r;
 r=m%n;
 while (__[1]____)
    {m=n;n=r;r=m%n;}
 return n;
}
fmin(int m,int n)
{ return    ___[2]___;}
int main()
{   int a,b;
    cin>>a>>b;
    cout<<"fmax is:"<<fmax(__[3]__);
    cout<<"fmin is:"<<fmin(__[3]__);
    return 0;
}
```

【答案】[1] r!=0　　[2]m*n/fmax(m,n)　[3] a,b

三．读程序写结果

1.

```
#include <iostream>
using namespace std;
int main()
{ char c;
  c=('z'-'a')/2+'A';
  putchar(c);
  return 0;}
```

【答案】 M

2.

```
#include <iostream>
using namespace std;
int main()
{ char fun(char,int);
  char a='A';
  int b=13;
  a=fun(a,b);
  putchar(a);
  return 0;
}
char fun(char a,int b)
{ char k;
  k=a+b;
  return k;
}
```

【答案】 N

3.

```
#include <iostream>
   using namespace std;
int aa(int x,int y);
int main()
{ int a=24,b=16,c;
   c=aa(a,b); //调用函数 aa(int x,int y)
   cout<<c<<endl;
   return 0;
}
int aa(int x,int y)     //参数传递，x=24, y=16
{ int w;
   while(y)
   { w=x%y;
      x=y;
      y=w;
    }
   return x;
}
```

【答案】 8

【解析】c=abc(a,b);　调用函数 aa(int x,int y)，参数传递 x=24，y=16，进入 while 循环，y=16，进入第一次循环：z=8，x=16，y=8，y=8 进入第二次循环：z=0，x=8，y=0，因为 y=0 跳出循环，返回 x 的值，x=8，因此主函数中 c=8。

4.

```
#include <iostream>
using namespace std;
void fun()
{ static int a=0;
   a+=2;
   cout<<a<<" ";
 }
 int main()
 { int cc;
    for(cc=1;cc<4;cc++)   fun();
    cout<<"\n";
    return 0;
 }
```

【答案】 2 4 6

5.

```
#include <iostream>
```

```
using namespace std;
unsigned fun6(unsigned num)
{ unsigned k=1;
   do
   {   k *=num; num/=10;
    } while (num);
    return k;
}
int main()
{ unsigned n=26;
   cout<< fun6(n)<<endl;
   return 0;}
```

【答案】 12

6.

```
#include <iostream>
using namespace std;
float fun(int x,int y)
{ return x+y ;}
int main()
{ int a=2,b=5,c=8;
   cout<<fun((int)fun(a+c,b),a-c)<<endl;
   return 0;
}
```

【答案】 9

7.

```
#include <iostream>
using namespace std;
int f()
{ static int i=0;
   int s=1;
   s+=i; i++;
   return s;
}
int main()
{ int i,a=0;
   for(i=0;i<5;i++) a+=f();
   cout<<a<<endl;
   return 0;}
```

【答案】 15

8.

```
#include <iostream>
using namespace std;
#define F(X,Y)    (X)*(Y)
```

```
int main()
{ int a=3, b=4;
cout<<F(a++, b++)<<endl;
return 0;}
```

【答案】12

【解析】 F(a++, b++)等价于(a++)*(b++)得到 3*4=12

四．编程题(以下各题均用函数实现)

1．从键盘输入 8 个浮点数，编程求出其和以及平均值。要求写出求和以及平均值的函数。

程序代码如下：

```
#include<iostream>
#include<cmath>
using namespace std;
double sum(double a[],int n);
double average(double a[],int n);
const N=10;
int main()
{
    int i;double a[N];
    cout<<"输入 10 个浮点数：";
    for(i=0;i<N;i++)
        cin>>a[i];
    cout<<"和："<<sum(a,N)<<","<<"平均值："<<average(a,N)<<endl;;
    return 0;
}
double sum(double a[],int n)        /*求和函数*/
{
    int i;
    double sum=0;
    for(i=0;i<n;i++)
        sum+=a[i];
    return sum;
}
double average(double a[],int n)          /*求平均值函数*/
{
    double average;
    average=sum(a,n)/n;
    return average;
}
```

2．哥德巴赫猜想之一是任何一个大于 5 的偶数都可以表示为两个素数之和，编程验证这一猜想。

程序代码如下：

```
    #include<iostream>
    using namespace std;
    int prime(int n)
```

```
{     int i;
 for(i=2;i<=n-1;i++)
 if(n%i==0)break;
 if(i==n)
        return 1;
 else
        return 0;
}
int main()
{
      int a,x,y;
      cout<<"请输入任意大于 5 的偶数：";
      cin>>a;
      for (x=3;x<=a/2;x++)
 { y=a-x;
        if(prime(x)*prime(y)!=0)
            cout<<a<<"="<<x<<"+"<<y<<endl;
 }
      return 0;
}
```

3．一个素数，当它的数字位置对换以后仍为素数，这样的数称为绝对素数，编写一个程序，求出所有的两位绝对素数。

【思路】编写一个判断一个数是否为素数的函数，若是返回“1”，否则返回“0”。因为要求所有的两位绝对素数，即 10 到 99 之间的绝对素数，首先判断此范围内的某个数是否是素数，如果是，将其十位和个位数字交换构成新数，再判断是不是素数，如果也是，则输出该绝对素数，使用循环完成该范围的遍历即可。

程序代码如下：

```
#include<iostream>
#include<cmath>
using namespace std;
int prime(int x);
int main()
{
    int m,i;
    cout<<"两位绝对素数："<<endl;
    for(i=10;i<100;i++)
        if(prime(i))
        { m=(i%10)*10+i/10;
          if(prime(m)) cout<<i<<"    ";
        }
    cout<<endl;
    return 0;
}
int prime(int x)            /*判断一个数是否为素数*/
```

```
{
    int i;
    for(i=2;i<x;i++)
        if(x%i==0)break;
    if(i>=x)
        return 1;
    else
        return 0;
}
```

4．给定年、月、日，计算该日是该年中的第几天。程序中要求有判断闰年的函数和计算天数的函数。

程序代码如下：

```
#include<iostream>
using namespace std;
int days(int a,int b,int c);
int f(int a);
int main()
{ int year,mon,day;
  cout<<"请输入年月日：";
  cin>>year>>mon>>day;
  cout<<"该日是该年的"<<days(year,mon,day)<<"天。"<<endl;
  return 0;
}
int f(int a)
{ if(a%4==0&&a%100!=0||a%400==0)
      return 1;
  else
  return 0;
}
int days(int a,int b,int c)
{ int d;
  switch(b)
  {  case 1:d=c;return d;
     case 2:d=31+c;return d;
     case 3: if(f(a)) d=31+29+c;
             else d=31+28+c;return d;
     case 4: if(f(a)) d=31*2+29+c;
             else d=31*2+28+c;return d;
     case 5: if(f(a)) d=31*2+30+29+c;
             else d=31*2+30+28+c;return d;
     case 6: if(f(a)) d=31*3+30+29+c;
             else d=31*3+30+28+c;return d;
     case 7: if(f(a)) d=31*3+30*2+29+c;
             else d=31*3+30*2+28+c;return d;
```

```
        case 8: if(f(a)) d=31*4+30*2+29+c;
                else d=31*4+30*2+28+c;return d;
        case 9: if(f(a)) d=31*5+30*2+29+c;
                else d=31*5+30*2+28+c;return d;
        case 10:if(f(a)) d=31*5+30*3+29+c;
                else d=31*5+30*3+28+c;return d;
        case 11:if(f(a)) d=31*6+30*3+29+c;
                else d=31*6+30*3+28+c;return d;
        case 12:if(f(a)) d=31*6+30*4+29+c;
                else d=31*6+30*4+28+c;return d;
    }
}
```

5．定义一个函数，通过 $f(n,x)$ 的形式调用，就可以计算 x^3+x-1，$(5+x)^3+(5+x)-1$，$(\sin x)^3+\sin x-1$ 等样式的表达式的值。

程序代码如下：

```
#include<iostream>
#include<cmath>
#include<iomanip>
using namespace std;
double f(int n,double i)
{       switch(n)
        { case 1:break;
          case 2:i=5+i;break;
          case 3:i=sin(i);break;
        }
        return i*i*i+i-1;
}
int main()
{
        double x;int n;
        cout<<"1:f(x)"<<endl;
        cout<<"2:f(5+x)"<<endl;
        cout<<"3:f(sinx)"<<endl;
        cout<<"请输入："<<endl;
        cout<<"x=";
        cin>>x;
        cout<<"n=";
        cin>>n;
        cout<<f(n,x)<<endl;
        return 0;
}
```

6．编程计算下列函数值：

$f(x,y)=\dfrac{s(x)}{s(y)}$。其中，$s(n)=\sum_{i=1}^{n}p(i)=p(1)+p(2)+\cdots+p(n),p(i)=i!$。要求：

①为计算 $p(i)$、$s(n)$ 和 $f(x,y)$ 各编写一个自定义函数。

②在主函数中键盘输入 x,y。

程序代码如下：

```
#include<iostream>
using namespace std;
int p(int i);
int s(int n);
int main()
{ int x,y;
   float a,b,c;
   cout<<"输入 x y：";
   cin>>x>>y;
   a=s(x);
   b=s(y);
   c=a/b;
  cout<<"f(x,y)="<<c<<endl;
   return 0;
}
int p(int i)
{ int a=1;
   for(;i>0;i--)
      a*=i;
   return a;
}
int s(int n)
{ int x=0;
   for(;n>0;n--)
      x+=p(n);
    return x;
}
```

7．编程计算：$\sum_{i=1}^{n}\dfrac{i+1}{i!}$(精度要求为 $\dfrac{n+1}{n!}<10^{-6}$)。要求使用静态局部变量。

程序代码如下：

```
#include<iostream>
#include<cmath>
using namespace std;
double f(int i);
int main()
{
```

```
    int i=1;double k,sum=0;
    k=(i+1)/f(i);
    while(k>=1e-6)
    {
       cout<<k<<endl;
       sum+=k;
       i++;
       k=(i+1)/f(i);

    }
     cout<<"共"<<i-1<<"项"<<endl;
     cout<<"和为："<<sum<<endl;;
     return 0;
}
double f(int i)
{
    static int j=1;
    j=j*i;
    return j;
}
```

8．编写一函数，由实参传来一个字符串，统计此字符串中字母、数字、空格和其它字符的个数，在主函数中输入字符串，输出上述结果。要求：将存放字母、数字、空格和其它字符个数的变量定义为全局变量。

程序代码如下：

```
#include<iostream>
using namespace std;
void fun(char s[]);
int sp=0,oth=0,lett=0,dig=0;
int main()
{
    char s[80];
    cout<<"输入字符串："<<endl;
    gets(s);
    fun(s);
    cout<<"空格:"<< sp<<" ，英文字符:"<< lett <<"，其它字符:"<<oth<<"，数字:"<<dig<<endl;
    return 0;
}
void fun(char s[])
{   int i=0;
    for(i=0;s[i]!='\0';i++)
        if(s[i]>='0'&&s[i]<='9')
            dig++;
        else if(s[i]==' ')
```

```
            sp++;
        else if(s[i]>= 'A'&&s[i]<= 'Z'||s[i]>= 'a'&&s[i]<= 'z')
            lett++;
        else oth++;
}
```

一、单选题

1. 若定义了 int n=2, *p=&n, *q=p;则下面【D】的赋值是非法的。

   A. p=q　　B. *p=*q　　C. n=*q　　D. p=n

   【分析】p 是整型指针，而 n 是整型，二者的类型不一致。

2. 若定义了 double *p, a;则能通过 cin 操作符给输入项读入数据的程序段是【C】。

   A. p=&a; cin>>p;　　B. *p=&a; cin>>p;

   C. p=&a; cin>>*p;　　D. p=&a; cin>>*a;

   【分析】p 是 double 型指针，通过 p=&a 指向了整型变量 a，因此 a 与*p 等价。

3. 若定义了 int a[10], i=3, *p; p=&a[5];下面不能表示为 a 数组元素的是【D】。

   A. p[-5]　　B. a[i+5]　　C. *p++　　D. a[i-5]

   【分析】i 的初始值为 3，i-5 的值为-2，因此 a[i-5]即为 a[-2],已越过了数组的边界。

4. 若有如下定义：

   ```
   int n[5]={1,2,3,4,5},*p=n;
   ```

   则值为 5 的表达式是【B】。

   A. *+5　　B. *（p+5）　　C. *p+=4　　D. p+4

   【分析】值为 5 的元素在数组 n 中是第 5 个元素，但 C++的数组下标从 0 开始，所以对应的是 n[4]元素。p 初始化指向数组 n，即指向 n[0]，p+4 即指向 n[4]，因此*(p+4)为 n[4]。

5. 设变量 b 的地址已赋给指针变量 ps，下面一定为“真”的表达式是【C】。

   A. b==&ps　　B. b==ps　　C. b==*ps　　D. &b==&ps

   【分析】A 是将指针变量 ps 的地址与 b 变量的值进行比较，当然不可能一定为“真”；B 是将指针变量 ps 的值（即 b 变量的地址）与 b 变量的值进行比较，也不可能一定为“真”；D 是将 b 变量的地址与指针变量 ps 的地址进行比较，当然不相等。

6. 设有以下定义和语句：

   ```
   int a[3][2]={1,2,3,4,5,6},*p[3];
   p[0]=a[1];
   ```

   则*(p[0]+1)所代表的数组元素是【C】。

   A. a[0][1]　　B. a[1][0]　　C. a[1][1]　　D. a[1][2]

   【分析】p 是指针数组，p[0]是一个整型指针，初始化指向二维数组 a 的行 1 的一维数组，即指向 a[1][0],且是一个列指针。所以 p[0]+1 是 a[1][1]的地址，C 为正确答案。

7. 若定义了 char *str=”Hello!”;下面程序段中正确的是【B】。

   A. char c[ ], *p=c; strcpy(p,str);

   B. char c[5], *p; strcpy(p=&c[1],&str[3]);

   C. char c[5]; strcpy(c,str);

   D. char c[5]; strcpy(p=c+2,str+3);

   【分析】在进行字符串复制时，需保证目的字符串具有足够的存储空间。A 中存储源字符串 str 至少需 7 个字节，而目的指针 p 指向的空间 c 未指定空间大小；C 中存储源字符串至少需 7 个字节，而目的字符数组 c 的大小为 5；D 中源字符串至少需要 4 个字节，而目的指针 p 指向的字符数组空间为 3 个字节。

8. 若有下面的程序段，则不正确的 fxy 函数的首部是【A】。

   ```
   #include <iostream.h>
   void main()
   ```

```
{   int a[20], n;
    …
    fxy(n, &a[10]);
    …
}
```

A. void fxy(int i, int j)　　B. void fxy(int x, int *y)

C. void fxy(int m, int n[])　　D. void fxy(int p, int q[10])

【分析】从 fxy 函数的调用形式可知,fxy 函数有两个参数，第一个参数为整型，第二个参数为整型指针，不符合这个约定的只有 A。

9. 不合法的带参数 main 函数的首部形式是【A】。

A. main(int argc, char *argv)　　B. main(int i, char **j)

C. main(int a, char *b[])　　D. main(int argc, char *argv[10])

【分析】main 函数的参数有两个，第一个参数为整型，说明命令行中的参数个数(包含命令)，第二个参数为一个字符二级指针或字符指针数组，记录命令行中的各参数。不符合此约定的只有 A。

10. 设有如下定义 int (*pt)( );则以下叙述中正确的是【D】。

A. pt 是指向一维数组的指针变量

B. pt 是指向整型数据的指针变量

C. pt 是一个函数名，该函数的返回值是指向整型数据的指针

D. pt 是指向函数的指针变量，该函数的返回值是整型数据

【分析】(*pt) 表明 pt 是一个指针变量，() 说明此指针变量是一个指向函数的指针，int 说明此函数的返回值为整型。

二、填空题

1. 请指出在 int *p[3];定义中 p 是____【1】包含三个整型指针的一维数组____。

在 int (*q)();定义中 q 是____【2】函数且其返回值的类型的整型指针____。

2. 若有如下定义，则使指针 p 指向值为 20 的数组元素的表达式是 p+=__4_____。

```
int a[6]={1,5,10,15,20,25},*p=a;
```

【分析】数组 a 中值为 20 的元素是 a[4],因此指针 p 需加 4。

3. 执行以下程序段后，x 的值为__600_____。

```
int a[3][2]={{1,2},{10,20},{15,30}};
int x, *p;
p=&a[0][0];
x=(*p)*(*(p+3))*(*(p+5));
```

【分析】p 指向 a[0][0]，因此 p+3 指向 a[1][1]，p+5 指向 a[2][2].

4. 请填空将函数补充完整，使得 add 函数具有求两个数之和的功能。

```
void add(int a, int b, __【1】int *__c)
{___*【2】c____=a+b;}
```

【分析】函数 add 无返回值，因此两数之和需通过函数的参数返回，因此需定义和值变量为指针变量。

5. 下面程序的功能是输出数组中的最大值，由 s 指针指向该元素，请将该程序补充完整。

```
#include <iostream.h>
void main()
{   int a[8]={6,7,2,9,1,10,5,8},*p,*s;
```

```
    for (p=a,s=a;p-a<8;p++)
        if (__*s>*p_____) s=p;
    cout<<“max:”<<*s<<endl;
}
```

【分析】s 指向打擂台过程中的擂主，即当前的最大值，而 p 依次指向数组中的各元素与 s 指向的擂主进行比大小。

6. 下面程序的功能是通过调用 aver 函数，计算数组中各元素的平均值。请将该程序补充完整。

```
#include <iostream.h>
float aver(int *a, int n)
{   int i;
    float x=0.0;
    for (i=0;i<n;i++)
        x+=____【1】a[i]____;
    x=____【2】x/n____;
    return x;
}
void main()
{   int m[]={2,1,7,4,5,9,6};
    float avg;
    avg=aver(m,7);
    cout<<“average=”<<avg<<endl;
}
```

【分析】第一个空实现数组中各元素求和，第二个空则计算平均值。

7. 下面函数的功能是计算指针 p 所指向的字符串中的字符个数。请将该程序补充完整。

```
unsigned int MStrlen(char *p)
{   unsigned int len;
    len=0;
    for (; *p!=____【1】’\0’____; p++)
    {
        len____【2】++____;
    }
    return____【3】len____;
}
```

【分析】字符串的遍历以字符串结束标记作为循环结束的判定，每遍历一个字符，则字符串长度计数器 len 加 1.

8. 下面函数同样也实现计算字符串 s 中字符个数，但方法与上一题有所不同。请将该程序补充完整。（提示：移动指针 p 使其指向字符串结束标志，此时指针 p 与字符串首地址之间的差值即为字符串中的字符个数）

```
unsigned int MStrlen(char s[])
{   char *p=s;
    while(*p!=____【1】’\0’____)
    {
```

```
        p++;
    }
    return_____【2】p-s-1_____;
}
```

【分析】指针 p 指向字符串末尾，指针 s 指向字符串的开始，注意字符串长度中不计入字符串结束标记字符。

9. 下面函数的功能是对两个字符串进行比较，返回两个字符串中第一个不同字符的 ASCII 值之差。例如，字符串“abcd”和“abm”，输出-10。请将该程序补充完整。

```
int cmp(char *p, char *q)
{   while (*p==*q && *p!=_____【1】'\0'_____)
    { p++; q++;}
    return(_____【2】*p-*q_____);
}
```

10. 下面程序的功能是输出命令行的参数，若程序生成的可执行文件为 file.exe，则执行该程序时键入命令：file NEW BEIJING

程序输出结果为：NEW BEIJING

请将该程序补充完整。

```
#include <iostream.h>
void main(int argc, char **argv)
{   while(--argc_____【1】>0_____)
    {   argv++;
        printf("%s",_____【2】*argv_____);
    }
}
```

【分析】main 函数可有两个参数，一个记录命令行参数的个数，包括命令本身；一个记录命令行中的各个参数。第一个空表示当命令行参数个数大于 0 时，即还有参数应输出参数；第二个空输出命令行中的参数。

三、阅读程序，写结果

1.
```
#include <iostream.h>
using namespace std;
void fact(int m, int n, int *p1, int *p2)
{   *p1=2*m+n;
    *p2=m-n/2;
}
void main()
{   int a,b,c,d;
    a=4; b=7;
    fact(a,b,&c,&d);
    cout<<c<<"  "<<d<<endl;
}
```

15 1

2.
```
#include <iostream.h>
using namespace std;
void main()
```

```
{   char str[]=”abcxyz”,*p;
    for (p=str;*p;p+=2)
        cout<<p;
    cout<<endl;
}
```

abcxyzcxyzyz

3.
```
#include <iostream.h>
using namespace std;
void main()
{   static int x[]={1,2,3};
    int s,i,*p=NULL;
    s=1;
    p=x;
    for(i=0;i<3;i++)
    {
        s*=*(p+i);
    }
    cout<<s<<endl;
}
```

6

4.
```
#include <iostream.h>
using namespace std;
void main()
{   int a[]={1,2,3,4,5};
    int *p= NULL;
    p=a;
    cout<<*p<<*(++p)<<*++p<<*(p--)<<*p++<<*p<<++(*p)<<*p;
}
```

3——3——2——2——2——2——2——1

5.
```
#include <iostream.h>
using namespace std;
char b[]=” program” ;
char *a=” PROGRAM” ;
void main()
{   int i=0;
    cout<<*a<<b+1<<endl;
    while(cout<<*(a+i))
    {
        cout<<*(a+i);
        i++;
    }
    cout<<” i=” <<i<<endl;
    while(--i)
    {
```

```
            cout<<*(b+i);
        }
        cout<<endl<<&b[3]<<endl;
    }
    Program
    PROGRAMi=7
    margor
    gram
```

6.
```
    #include <iostream.h>
    using namespace std;
    void main()
    {
        int a[]={10,20,30,40}, *pa=a;
        int * &pb=pa;
        pb++;
        cout<<*pa<<endl;
    }
    20
```

四、编程题

1. 编写函数，对传送过来的三个数选出最大和最小值，并通过形参传回调用函数。

【分析】打擂台算法，设第一个数为最大和最小值，分别与后面的数“打擂台”，找出最大和最小值。

【参考源代码】

```
void max_min(int a, int b, int c, int *max, int *min)
{
    *max=a; *min=a;
    if (b>*max) *max=b;
    if (c>*max) *max=c;
    if (b<*min) *min=b;
    if (c<*min) *min=c;
    return;
}
```

2. 求一个 3×3 二维数组主对角线元素之和(要求函数参数定义为指针数组方式)。

【参考源代码】

```
int sum_diagnol(int (*a)[3])
{
    return((*a)[0]+(*(a+1))[1]+(*(a+2))[2]);
}
```

3. 有 n 个整数，使前面各数顺序向后移动 m 个位置，最后 m 个数变成最前面 m 个数，

见图 6-18。写一函数实现以上功能，在主函数中输入 n 个整数和输出调整后的 n 个数。

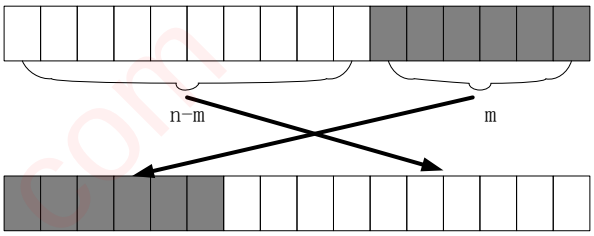


图 6-18 编程题 3 示意

【分析】在移动数组元素时，为了避免原始数据的丢失，用另一个数组保存原数组的数据。函数中采用动态存储分配的形式：new int(n)生成一个 n 个元素的动态数组。

【参考源代码】

```
void num_move(int *a,int n,int m)
{
    int *p;
    p=new int(n);    //动态分配一个具有 n 个元素的整型数组 p
    int i;
    for (i=0;i<n;i++)    //将数组 a 复制到数组 p 中
        p[i]=a[i];
    for (i=0;i<n;i++)    //将数组 p 中的元素移动 m 个位置后复制到数组 a 中
        a[i]=p[(i+n-m)%n];
    return;
}
```

4. 编写一个字符串连接的函数 radd(char *s,char *t,int f)，其中 f 为标志变量，当 f=0 时，将 s 指向的字符串连接到 t 指向的字符串的后面；当 f=1 时，将 t 指向的字符串连接到 s 指向的字符串的后面。并写出调用该函数的完整程序。

【分析】程序主体仅实现了字符串 s 连接到字符串 t 后的功能，因此需依据标志变量 f 的值决定是否交换字符指针 s 和 t 的指向。

【参考源代码】

```
void radd(char *s,char *t,int f)
{
    if (f==1)     //交换字符指针 s 和 t 的指向
    {   char *cp;
        cp=s;
        s=t;
        t=cp;
    }
    //将字符串 s 连接到字符串 t 的后面。
    while (*t) t++;      //将 t 指针指到第一个字符串的末尾
    while (*s) *t++=*s++;    //依次将第二个字符串中的字符复制到第一个字符串的末尾。
    *t=0;              // 为新的字符串添加字符串结束标志
    return;
}
```

5. 编写程序，利用指向函数的指针实现求 1 到 n 的和与阶乘。

【分析】分别定义两个函数实现 1 到 n 的和与阶乘。在主函数中定义一个函数指针，依据用户的输入决定函数指针的指向。

【参考源代码】

```
#include <iostream>
using namespace std;

long sum(int n)      //求 1 到 n 的和
{
  int i;
  long s=0;
  for (i=0;i<=n;i++)
    s=s+i;
  return s;
}
long factorial(int n)      //求 1 到 n 的阶乘
{
  int i;
  long p=1;
  for (i=1;i<=n;i++)
     p=p*i;
  return p;
}

void main()
{
  long (*fm)(int);     //定义一个函数指针 fm
  int f,n;
  cin>>f>>n;
  // 用户输入 f，f 的值为 0，fm 指向求和的函数；值为 1，fm 指向求阶乘的函数。
  if (f==0)
    fm=sum;
  else if (f==1)
    fm=factorial;
  cout<<(*fm)(n)<<endl;

}
```

习题 7

一、单项选择题

1. 设有如下定义，则在 Microsoft VisualC++6.0 环境下，表达式 sizeof(y)的值是【　　】。

```
struct data
{int a;
 char b[30];
 char c;
 data *last,*next;
 }y;
```

A) 43　　B) 46　　C) 48　　D) 44

【答案】 D

【解析】结构变量所占内存空间等于各成员所占内存空间之和。但是，对于 Visual C++ 6.0 平台来说，默认对齐方式为字长对齐，默认字长为 4 个字节，因此，结构变量所占存储空间应为字长的整数倍。sizeof(y)的结果应该为 44，即结构变量 y 所占存储空间为 44 个字节。各成员占存储空间图示如下：

占内存空间	1-4	5-34	35　36	37-40	41-44
成员	a	b[30]	c	last	next

如果将 char c 替换为 bool c 或替换为 short c，结果都是一样的。

2. 设有如下定义，则在 Microsoft VisualC++6.0 环境下，表达式 sizeof(y)的值是【　　】。

```
struct data
{int a;
 char b[30];
 data *last,*next;
 char c;
 }y;
```

A) 43　　B) 46　　C) 48　　D) 44

【答案】 C

【解析】同上题。各成员占存储空间图示如下：

占内存空间	1-4	5-36	37-40	41-44	45-48
成员	a	b[30]	last	next	c

3. 若有以下定义，则结构体成员引用形式不正确的是【　　】。

```
struct {int day,mouth,year;}a,*p=&a;
```

A) p->day　　B) a->day　　C) (*p).day　　D) a.day

【答案】B

4. 设有以下程序段，则表达式的值不为 100 的是【　　】。

```
struct st
{int a,*b;}
int main()
{int m1[]={10,100},m2[]={100,200};
 st *p,x[]={99,m1,100,m2};
```

```
p=x;
……
}
```

A) *(++p->b)　　B) (++p)->a　　C) ++p->a　　D) (++p)->b

【答案】D

【解析】(++p)->b 为指针变量，存放内存地址，其值等于 m2。

5. 设有如下定义，则在 Microsoft VisualC++6.0 环境下，表达式 sizeof(data)的值是【　　】。

```
struct data
{   int a;
    float b;
    union
    {char u1;
    double u2;
    };
};
```

A) 12　　B) 14　　C) 16　　D) 10

【答案】C

【解析】共用体变量所占的内存长度等于最长的成员的长度，无论共用体类型有多少成员，它们共用内存单元。

6. 设有如下定义，则下列叙述中正确的是【　　】。

```
struct student
{   int a;
    float b;
    char c[30];
}stu;
```

A) stu 是结构体变量名　　B) student 是结构体变量名

C) stu 是结构体类型名　　D) struct 是结构体类型名

【答案】A

7. 设有如下定义，则引用共用体中 h 成员的正确形式为【　　】。

```
union un
{int h;char c[10];};
struct st
{int a[2];
 un h;
}s={{1,2},3},*p=&s;
```

A) p.un.h　　B) (*p).h.h　　C) p->st.un.h　　D) s.un.h

【答案】B

8. 当定义一个结构体变量时系统分配给它的内存是【　　】。

A) 各成员所需内存量的总和

B) 结构体中第一个成员所需内存量

C) 成员中占内存量最大者所需的容量

D) 结构体中最后一个成员所需内存量

【答案】A

二、填空题

1. 设有如下定义，则在 Microsoft VisualC++6.0 环境下，变量 s 在内存中占的字节数是_____。

```
struct st
{char num[5];
 int age;
 float score;
 }s;
```

【答案】16

2. 若定义了 struct{int d,m,y;}a,*p=&a; 可用 a.d 引用结构体成员，请写出引用结构体成员 a.d 的其它两种形式_____①_____、_____②_____。

【答案】p->d、(*p).d

3. 设有以下结构体类型的定义，请把结构体数组 stu 的定义补充完整。

```
struct Student
 {char num[10];
  int age;
  float score;
  };
  _____ stu[15];
```

【答案】**Student 或 struct Student**

4. 以下程序用来输出结构体变量 x 所占内存单元的字节数，请填上适当内容。

```
struct st
{double d;
 char arr[20];
};
int main()
{ st x;
 cout<<"x size："<<_____<<endl;
 return 0;
}
```

【答案】**sizeof(x)或 sizeof(st)**

5. 下面程序的输出结果是_____。

```
#include<iostream>
using namespace std;
struct Student
{int num;
 char name[20];
 float score;
};
void print(Student *p)
{cout<<p->name<<endl;}
int main()
{ Student s[3]={1,”zhao”,89.0,2,”wang”,90.5,3,”Li”,95.5}
 print(s+2);
 return 0;
```

```
}
```

【答案】**Li**

三、阅读程序，写出程序运行结果

1. 程序代码如下

```
#include<iostream>
using namespace std;
struct sp
{ int a;
   int *b;
}*p;
int d[3]={10,20,30};
sp t[3]={70,&d[0],80,&d[1],90,&d[2]};
int main()
{p=t;
 cout<<++(p->a)<<*++p->b<<endl;
 return 0;
}
```

【答案】7120

2. 程序代码如下。

```
#include<iostream>
using namespace std;
union data
{ int a;
   float b;
   long c;
   char d[4];
}*p;
int main()
{ data u;
u.a=0x41424344;
cout<<"\na="<<u.a<<"\nb="<<u.b<<"\nc="<<u.c;
cout<<"\nd[0]="<<u.d[0]<<",d[1]="<<u.d[1]<<",d[2]="<<u.d[2]<<",d[3]="<<u.d[3]<<endl;
return 0;
}
```

【答案】a=1094861636
b=12.1414
c=1094861636
d[0]=D,d[1]=C,d[2]=B,d[3]=A

四、编程题

1．定义一个表示时间的结构体，可以精确表示年、月、日、时、分、秒；提示用户输入年、月、日、时、分、秒的值，然后完整地显示出来。

程序代码如下：

```
#include <iostream>
using namespace std;
```

```
struct Tdate
{
    int year;
    int month;
    int day;
    int hour;
    int minute;
    int second;
};
int main()
{
    Tdate myDate;
    cout << "Enter the value of year: ";
    cin >> myDate.year;
    cout << "Enter the value of month: ";
    cin >> myDate.month;
    cout << "Enter the value of day: ";
    cin >> myDate.day;
    cout << "Enter the value of hour: ";
    cin >> myDate.hour;
    cout << "Enter the value of minute: ";
    cin >> myDate.minute;
    cout << "Enter the value of second: ";
    cin >> myDate.second;
    cout << "Current time is: "
         << myDate.year << "-"
         << myDate.month << "-"
         << myDate.day << " "
         << myDate.hour << ":"
         << myDate.minute << ":"
         << myDate.second << endl;
    return 0;
}
```

2. 编写一个记录 30 个学生的姓名、学号、年龄和性别的程序，要求使用结构体类型。用 for 循环获得键盘输入数据，数据输入完毕后用屏幕输出。

程序代码如下：

```
#include <iostream>
using namespace std;

struct StudentStruct
{   char name [10];
    char sex;
    int age;
```

```
    long number;
} student[30];
void main()
{
for (int i=0;i<30;i++)
{
cin>> student[i].name;
cin>> student[i].sex;
cin>> student[i].age;
cin>> student[i].number;
}
for (int i=0;i<30;i++)
{
cout<< student[i].name;
cout<<student[i].sex;
cout<<student[i].age;
cout<<student[i].number<<endl;
}
}
```

3. 定义一个结构体变量（包括年、月、日），编写程序，输入年、月、日，程序能计算并输出该日在本年中是第几天。注意闰年问题。

程序代码如下：

```
#include<iostream>
using namespace std;
struct Date
{int year,month,day;}date;   //结构变量 date 用来存放输入的年、月、日

int main()
{int i,n=0;     //n 为天数
cout<<"input year,month,day:";
cin>>date.year>>date.month>>date.day;
//累计计算该日是第几天
for(i=1;i<date.month;i++)
switch(i)
  {case 1:
   case 3:
   case 5:
   case 7:
   case 8:
   case 10:
   case 12:n+=31;break;
   case 4:
   case 6:
```

```
    case 9:
    case 11:n+=30;break;
    case 2:if(date.year%4==0&&date.year%100!=0||date.year%400==0)
                n+=29;   //是闰年
           else
                n+=28;   //不是闰年
  }
n+=date.day;
cout<<date.month<<"/"<<date.day<<" is the "<<n<<"th day in "<<date.year<<"."<<endl;
return 0;
}
```

程序代码还可改写如下：

```
#include<iostream>
using namespace std;

struct Date
{int year,month,day;}date;
int main()
{int i,n=0;     //n 为天数
int day_tab[13]={0,31,28,31,30,31,30,31,31,30,31,30,31};
cout<<"input year,month,day:";
cin>>date.year>>date.month>>date.day;
//累计计算该日是第几天
for(i=1;i<date.month;i++)
    n+=day_tab[i];

n+=date.day;
if(date.year%4==0&&date.year%100!=0||date.year%400==0)
    n+=1;   //闰年加 1 天
cout<<date.month<<"/"<<date.day<<" is the "<<n<<"th day in "<<date.year<<"."<<endl;
return 0;
}
```

4. 写一个函数 days，实现上面的计算。由主函数将结构变量的值传递给 days 函数，计算后将日子数传回主函数输出。

程序代码如下：

```
#include<iostream>
using namespace std;
struct Date
{int year,month,day;};

int days(Date d1)
{int i,n=0;     //n 为天数
```

```
int day_tab[13]={0,31,28,31,30,31,30,31,31,30,31,30,31};
//累计计算该日是第几天
for(i=1;i<d1.month;i++)
    n+=day_tab[i];

n+=d1.day;
if(d1.year%4==0&&d1.year%100!=0||d1.year%400==0)
    n+=1;   //闰年加 1 天
return n;
}

int main()
{
Date someday;
cout<<"input year,month,day:";
cin>>someday.year>>someday.month>>someday.day;
cout<<someday.month<<"/"<<someday.day<<"/"<<someday.year
     <<" is the "<<days(someday)<<"th day in the year."<<endl;
return 0;
}
```

5. 有 5 个学生，每个学生的数据包括 num（学号）、name（姓名）、score[3]（3 门课的成绩）、total（总分）及 average（平均分）。要求编写 3 个函数，它们的功能分别为：（1）输入函数，用于从键盘读入学号、姓名和 3 门课的成绩；（2）计算总分和平均分的函数，用于计算每个学生的总分及平均分；（3）输出函数，显示每位学生的学号、姓名、总分及平均分。这 3 个函数的形式参数均为结构体指针和整型变量，函数的类型均为 void。

程序代码如下：

```
#include<iostream>
using namespace std;
const n=5;
struct Student
{ int num;
  char name[20];
  float score[3];
  float total;
  float average;
};

void input(Student *p,int n)
{
    for(int i=0;i<n;i++,p++)
    {
        cout<<"input student"<<i+1<<endl;
        cout<<"NO.:";
```

```
            cin>>p->num;
            cout<<"name:";
            cin>>p->name;
            for(int j=0;j<3;j++)
            {
                cout<<"score"<<j+1<<":";
                cin>>p->score[j];
            }
        }
    }

    void fun(Student *p,int n)
    {
        for(int i=0;i<n;i++,p++)
        {
         p->total=0;
         for(int j=0;j<3;j++)
             p->total+=p->score[j];
         p->average=p->total/3;
        }
    }

    void display(Student *p,int n)
    {
       cout<<"num"<<"    "<<"name"<<"    "<<"total"<<"    "<<"average"<<endl;
       for(int i=0;i<n;i++,p++)
          cout<<p->num<<"    "<<p->name<<"        "<<p->total<<"        "<<p->average<<endl;
    }

    int main()
    {
        Student stu[n];
        input(stu,n);
        fun(stu,n);
        display(stu,n);
        return 0;
    }
```

6．阅读下面的源程序，说明它实现什么功能。

提示：函数 rand()可以生成 0~RAND_MAX(0x7fff)之间的一个随机数，srand 函数为它设置种子。函数 time()可以取得系统当前的时间，是一个无符号长整数。

源程序：

```
#include<iostream>
#include<ctime>
```

```
using namespace std;
enum colorball{redball,yellowball,blueball,whilteball,blachball};

int main()
{ srand((unsigned)time(NULL));
   int count=0;
   for(int i=0;i<100;i++)
      if(rand()*5/RAND_MAX==redball)
           count++;
   cout<<count<<"%"<<endl;
   return 0;
   }
```

答案：

有 5 种颜色的球，每次随机取一个，共取 100 次，看看取到红球的概率。

第 8 章习题解答

一. 选择题

1. 关于类和对象，不正确的说法是（）
A. 类是一种数据类型，它封装了数据和函数
B. 类是对某一类对象的抽象
C. 可以基于类这种数据类型定义类的引用
D. 一个类的对象只有一个
【答案】D
【解析】类是一种数据类型，可以基于“类”这种数据类型定义多个称为“对象”的变量。

2. 类定义的外部，可以被访问的类的成员有（）
A. public 的类成员
B. public 或 private 的类成员
C. private 或 protected 的类成员
D. public 或 private 的类成员
【答案】A
【解析】类的成员数据或成员函数的访问属性分为三种情况：private、public 和 protected，即私有访问属性、公有访问属性和保护访问属性，类定义的外部只能访问公有访问属性的成员。

3 关于 this 指针，说法错误的是（）
A. this 指针必须显式说明
B. 当创建一个对象后，this 指针就指向该对象
C. 成员函数拥有 this 指针
D. 静态成员函数拥有 this 指针
【答案】D
【解析】this 指针是一种特殊的指针，它指向成员函数当前操作的数据所属的对象。不同的对象调用相同的成员函数时，this 指针将指向不同的对象，也就可以访问不同对象的成员数据。而静态成员函数是一个类的所有对象共享的成员，而不仅仅是某一对象的成员。因此，可以在没有任何对象存在的情况下，可以使用静态成员函数，而使用 this 指针必须有明确的对象所指。

4. 调用形式参数为普通对象的函数时，系统会自动调用相应类的（）
A. 名字不同于类名的一般成员函数
B. 构造函数
C. 析构函数
D. 拷贝构造函数
【答案】D
【解析】若函数的形参为类的对象，调用函数时，实参赋值给形参，系统自动调用拷贝构造函数实现拷贝赋值。

5. 定义某类的对象后，再删除该对象，系统会自动调用（）
A. 名字不同于类名的一般成员函数

B. 拷贝构造函数
C. 构造函数
D. 析构函数
【答案】D
【解析】当对象生存期结束时，需要调用析构函数，释放对象所占的内存空间。

6. 对于析构函数，不正确的描述是（）
A. 系统可以提供默认的析构函数
B. 析构函数不能进行重载
C. 析构函数没有参数
D. 析构函数可以设置默认参数
【答案】D
【解析】析构函数是无参的，析构函数的名字又是唯一的。

7. 关于静态成员，不正确的描述为（）
A. 静态成员函数可以访问一般成员数据
B. 静态成员函数可以访问静态成员数据
C. 静态成员函数不可访问一般成员函数
D. 静态成员函数可以访问静态成员函数
【答案】A
【解析】静态成员函数是一个类的所有对象共享的成员，而不仅仅是某一对象的成员。因此，可以在没有任何对象存在的情况下使用静态成员函数。如果在静态成员函数中访问一般成员数据，会造成这些一般成员数据没有所属对象的错误。

8. 关于友元，错误的描述是（）
A. 关键字 friend 用于声明友元
B. 一个类中的成员函数可以是另一个类的友元
C. 类与类之间的友元关系不具有传递性
D. 类与类之间的友元关系具有对称性
【答案】D
【解析】友元关系不具有对称性和传递性。

9. 下列有关类模板、类和对象的说法中，正确的是（）
A. 对象可以由类模板直接定义或生成
B. 类可以定义或生成类模板
C. 对象是类的实例，为对象分配存储空间而不是为类分配存储空间
D. 类是对象的实例，为类分配存储空间而不是为对象分配存储空间
【答案】C
【解析】由类模板可以生成类，由类可以生成对象。定义类时无需分配存储空间，而定义对象时需要分配存储空间。

10. 下列关于类的可访问性，错误的描述是（）
A. 类外的一般函数可以访问该类的公有成员数据
B. 类外的一般函数可以访问该类的公有成员函数

C. 同一个类的成员函数可以访问该类的公有成员数据
D. 同一个类的成员函数不可访问该类的私有成员数据
【答案】D
【解析】同一个类的成员函数可以访问该类的任何访问属性的成员数据。

二. 简答题

1．简要说明类与对象之间的关系。
答：类表示现实生活中的一类实体，类是具有相同属性的对象的抽象，类实际是一种用户自定义数据类型，基于该数据类型可以定义对象。

2．简要描述构造函数的作用，类的成员函数是构造函数需要满足什么条件。
答：构造函数完成对象的初始化工作，构造函数必须满足以下几个条件：构造函数的名字与类的名字相同，构造函数没有返回值。

3．简要描述析构函数的作用，类的成员函数是析构函数需要满足什么条件。
答：析构函数用于当对象生存期结束时释放对象所占的内存空间，析构函数必须满足以下几个条件：析构函数的名字是：”~”+类的名字，析构函数没有返回值，析构函数没有参数。

4．何种情况之下一定要手动编写析构函数代码。
答：构造函数中使用 new 运算符向堆空间申请了用于存储成员数据的存储空间，必须具体定义析构函数用来删除该空间资源。

5．什么是浅拷贝构造函数，什么是深拷贝构造函数。
答：当定义拷贝构造函数时，只是完成一一对应的成员值的简单复制，称其为浅拷贝构造函数。深拷贝构造函数是对浅拷贝构造函数的完善，对指针成员必须重新申请空间并返回空间地址给相应指针成员。

6．何种情况之下一定要手动编写深拷贝构造函数代码。
答：如果类的数据成员包含指针变量，类的构造函数必须使用 new 运算符为这个指针动态申请堆空间。如果此时还只是简单的使用浅拷贝的方式进行对象复制。最后，在退出运行时，由于两个对象的指针指向同一个堆空间中的资源，当调用析构函数先后删除这两个指针指向的堆空间中的资源时，程序会报错。为了解决此问题，必须定义深拷贝构造函数。

7．简要描述调用拷贝构造函数的几种情况。
答：拷贝构造函数常应用于以下三种情况：（1）当用一个类的对象去初始化该类的另一个对象时，系统自动调用拷贝构造函数实现拷贝赋值。（2）若函数的形参为类的对象，调用函数时，实参赋值给形参，系统自动调用拷贝构造函数实现拷贝赋值。（3）当函数的返回值是类的对象时，系统调用拷贝构造函数实现拷贝赋值。

8．简要描述静态成员和一般成员的区别。
答：静态成员是一个类的所有对象共享的成员，而不仅仅是某一对象的成员。一般成员是隶属于某个具体的对象的。

9．从类的内部和类的外部两个方面描述访问属性 private 和 public 的访问规则。

答：类的内部和类的外部两个方面描述访问属性 private 和 public 的访问规则，如下表所示。

	公有成员数据	私有成员数据	公有成员函数	私有成员函数
内的内部的可访问性：同一个类的成员函数	可以访问	可以访问	可以访问	可以访问
内的外部的可访问性：另一个类的成员函数或一般函数	可以访问	不可访问	可以访问	不可访问

10．什么是组合关系，组合关系在 C++中是如何实现的。

答：现实世界中对象与对象之间的整体-部分关系又称为组合关系。组合关系是通过成员对象的方式实现的。

三、阅读程序，并给出结果

1．

```
#include <iostream>
using namespace std;
class Test
{
private:
    int a;
    int b;
public:
    Test(int x, int y)
    {
        a=x;
        b=y;
        cout<<"Constructor 1"<<endl;
        cout<<a<<","<<b<<endl;
    }
    Test(Test &t)
    {
        cout<<"Constructor 2"<<endl;
        cout<<t.a<<","<<t.b<<endl;
    }
};
int main()
{
    Test t1(9,7);
    Test t2(t1);
    return 0;
}
```

1．运行结果是：

Constructor 1
9,7
Constructor 2
9,7
【解析】main 函数中首先调用普通构造函数 Test(int x, int y)生成一个对象 t1，然后调用拷贝构造函数 Test(Test &t)生成另一个对象 t2，这两个对象 t1 和 t2 具有相同的成员数据值。

2．

```
#include<iostream>
using namespace std;
class Test
{
private:
    int x,y;
public:
    Test()
    {
        x=y=0;
    }
    Test(int a,int b)
    {
        x=a;
        y=b;
    }
    void disp()
    {
        cout<<"x="<<x<<",y="<<y<<endl;
    }
};
int main()
{
    Test s1,s2(1,2),s3(10,20);
    Test *pa[3]={&s1,&s2,&s3};
    for(int i=0;i<3;i++)
        pa[i]->disp();
    return 0;
}
```

2．运行结果是：
x=0,y=0
x=1,y=2
x=10,y=20
【解析】Test s1 调用的是默认构造函数 Test()，Test s2(1,2),s3(10,20)调用的是构造函数 Test(int a, int b)，然后定义一个指针数组，并且每个数组元素指向一个对象，使用 for 循环语句进行遍历访问。

3.

```
#include<iostream>
using namespace std;
class Test
{
    char c1,c2;
public:
    Test(char a)
    {
        c2=(c1=a)-32;
    }
    void disp()
    {
            cout<<c1<<"转换为"<<c2<<endl;
    }
};
void main()
{
    Test a('a'),b('b');
    a.disp();
    b.disp();
}
```

3．运行结果是：

a 转换为 A

b 转换为 B

【解析】构造函数 Test(char a)的作用是将参数 a 赋值给成员数据 c1，并且让成员数据 c2 和成员数据 c1 相差 32，即同一个字母的大小写形式的 ASCII 码差值。

4.

```
#include<iostream>
using namespace std;
class Test
{
    int n;
    static int sum;
public:
    Test(int x)
    {
        n=x;
    }
    void add()
    {
        sum+=n;
```

```
    }
    void disp()
    {
        cout<<"n="<<n<<",sum="<<sum<<endl;
    }
};
int Test::sum=0;
void main()
{
    Test a(2),b(3),c(5);
    a.add();
    a.disp();
    b.add();
    b.disp();
    c.add();
    c.disp();
}
```

4．运行结果是：

n=2,sum=2

n=3,sum=5

n=5,sum=10

【解析】sum 是类 Test 的静态成员数据，为所有对象共享。

四、编程题

1．定义一个长方体类 Box，该类中定义的成员数据包括：长度、宽度和高度，定义的成员函数包括：构造函数、析构函数、设置长方体长度、宽度和高度的成员函数以及计算长方体体积的成员函数，并在 main 函数中创建该类对象，调用计算长方体体积的成员函数并输出结果。

思路：将长度、宽度和高度实现为长方体类 Box 的私有成员数据，将设置长方体长度、宽度和高度的成员函数以及计算长方体体积的成员函数定义为公有访问属性。

程序代码如下：

```
#include <iostream>
using namespace std;
class Box
{
private:
    float length;
    float width;
    float height;
public:
    Box();
    ~Box();
    void setLength(float pLength);
    void setWidth(float pWidth);
```

```
    void setHeight(float pHeight);
    float volumn();
};
// 构造函数
Box::Box()
{
}
// 析构函数
Box::~Box()
{
}
// 设置长度函数
void Box::setLength(float pLength)
{
    length=pLength;
}
// 设置宽度函数
void Box::setWidth(float pWidth)
{
    width=pWidth;
}
// 设置高度函数
void Box::setHeight(float pHeight)
{
    height=pHeight;
}
float Box::volumn()
{
    return length*width*height;
}
int main()
{
    Box b;
    b.setLength(1.20f);
    b.setWidth(0.86f);
    b.setHeight(2.30f);
    cout<<"the volumn is: "<<b.volumn()<<endl;
    return 0;
}
```

2．定义一个学生类 Student，该类中包括：一个一般成员数据 score（分数）和两个静态成员数据 total（总分）和 count（学生总的人数），一般成员函数 scoreTotalCount(float s)用于设置分数、计算总分和累计学生人数，静态成员函数 sum 用于返回总分，静态成员函数 average 用于求平均值。

思路：将一般成员数据 score（分数）和两个静态成员数据 total（总分）和 count（学生总的人数）都设置为公有访问属性，将一般成员函数 scoreTotalCount(float s)，静态成员函数 sum，静态成员函数 average 也设置为公有访问属性。

程序代码如下：

```
#include <iostream>
using namespace std;
class Student
{
public:
    float score;
    static float total;
    static int count;
public:
    void scoreTotalCount(float s);
    static float sum();
    static float average();
};
float Student::total = 0.0f;
int Student::count = 0;
void Student::scoreTotalCount(float s)
{
    this->score = s;
    total = total+s;
    count++;
}
float Student::sum()
{
    return total;
}
float Student::average()
{
    return total/count;
}
int main()
{
    Student s1,s2,s3;
    s1.scoreTotalCount(96.0f);
    s2.scoreTotalCount(86.0f);
    s3.scoreTotalCount(90.0f);
    cout<<"the sum is: "<<Student::sum()<<endl;
    cout<<"the average is: "<<Student::average()<<endl;
    return 0;
}
```

3．定义一个员工类 Employee，该类中包括的成员数据有：员工编号、员工姓名、员工部门、员工职位、员工年龄和员工学历，其中，员工姓名、员工部门、员工职位和员工学历以字符指针方式定义，该类中包括的成员函数有：构造函数、拷贝构造函数、析构函数和打印员工信息的成员函数。在 main 函数中创建三个该类的对象，一种创建方式是调用一般构造函数，另一种创建方式是调用拷贝构造函数，再一种创建方式是使用 new 向堆空间申请，创建后分别调用打印员工信息的成员函数进行输出。

思路：将员工编号、员工姓名、员工部门、员工职位、员工年龄和员工学历等成员数据设置为私有访问属性，将构造函数、拷贝构造函数、析构函数和打印员工信息的成员函数设置为公有访问属性。对于拷贝构造函数，其参数必须为自身对象的引用，而且由于成员数据中包含指针类型的变量，需要定义深拷贝构造函数。

程序代码如下：

```
#include <iostream>
using namespace std;
class Employee
{
private:
    int id;
    char* name;
    char* department;
    char* position;
    int age;
    char* degree;
public:
    Employee(int pId, char* pName, char* pDepartment, char* pPosition, int pAge, char* pDegree);
    Employee(Employee& emp);
    ~Employee();
    void printEmp();
};
// 拷贝构造函数
Employee::Employee(int pId, char* pName, char* pDepartment, char* pPosition, int pAge, char* pDegree)
{
    cout<<"constructing..."<<endl;
    id=pId;
    name = new char[strlen(pName)+1];
    if(name != 0)
       strcpy(name,pName);
    department = new char[strlen(pDepartment)+1];
    if(department != 0)
        strcpy(department,pDepartment);
    position = new char[strlen(pPosition)+1];
    if(position != 0)
        strcpy(position,pPosition);
```

```
    age=pAge;
    degree = new char[strlen(pDegree)+1];
    if(degree != 0)
        strcpy(degree,pDegree);
}
// 深拷贝构造函数
Employee::Employee(Employee& emp)
{
    cout<<"deep copy constructing..."<<endl;
    id=emp.id;

    name=new char[strlen(emp.name)+1];
    if(name != 0)
        strcpy(name,emp.name);

    department=new char[strlen(emp.department)+1];
    if(department != 0)
        strcpy(department,emp.department);

    position=new char[strlen(emp.position)+1];
    if(position != 0)
        strcpy(position,emp.position);

    age=emp.age;

    degree=new char[strlen(emp.degree)+1];
    if(degree != 0)
        strcpy(degree,emp.degree);
}
// 析构函数
Employee::~Employee()
{
    cout<<"deconstructing..."<<this->id<<endl;
    delete[] name;
    delete[] department;
    delete[] position;
    delete[] degree;
}
// 打印员工信息
void Employee::printEmp()
{
    cout<<this->id<<", "
        <<this->name<<", "
        <<this->department<<", "
```

```
            <<this->position<<", "
            <<this->age<<", "
            <<this->degree<<endl;
}
int main()
{
    Employee emp1(101,"Zhang San","软件开发部","项目经理",30,"硕士");
    Employee emp2(emp1);
    Employee* p = new Employee(102,"Li Si","软件开发部","软件工程师",25,"本科");
    emp1.printEmp();
    emp2.printEmp();
    p->printEmp();
    delete p;
    return 0;
}
```

4．设计一个 Bank 类，实现银行某账号的资金往来账目管理，包括建账号、存入、取出等。

思路：Bank 类包括私有数据成员 top(当前账指针)，date(日期)，money(金额)，rest(余额)和 sum(累计余额)。另有一个构造函数和三个成员函数 bankin()(处理存入账)，bankout()处理取出账)和 disp() (输出明细账)。

程序代码如下：

```
#include <iostream>
#include <string>
using namespace std;
#define Max 100
class Bank
{
    int top;
    char date[Max][10];        // 日期
    int money[Max];            // 金额
    int rest[Max];             // 余额
    static int sum;            // 累计余额
public:
    Bank()
    {
        top=0;
    }
    void bankin(char d[],int m)
    {
        strcpy(date[top],d);
        money[top]=m;
        sum=sum+m;
        rest[top]=sum;
        top++;
```

```
    }
    void bankout(char d[],int m)
    {
        strcpy(date[top],d);
        money[top]=-m;
        sum=sum-m;
        rest[top]=sum;
        top++;
    }
    void disp();
};
int Bank::sum=0;
void Bank::disp()
{
    int i;
    cout<<"日期          存入     取出     余额"<<endl;
    for(i=0;i<top;i++)
    {
        cout<<date[i]<<"    ";
        if(money[i]<0)
        {
            cout<<"                "<<-money[i];
            cout<<"     "<<rest[i]<<endl;
        }
        else
        {
            cout<<money[i];
            cout<<"                    "<<rest[i]<<endl;
        }
    }
}
void main()
{
    Bank obj;
    obj.bankin("2001.2.5",1000);
    obj.bankin("2001.3.2",2000);
    obj.bankout("2001.4.1",500);
    obj.bankout("2001.4.5",800);
    obj.disp();
}
```