2013 年下半年程序员考试下午真题

（参考答案）

● 阅读以下说明和流程图，填补流程医中的空缺（1）～（5）将解答填入答题纸的对应栏内。

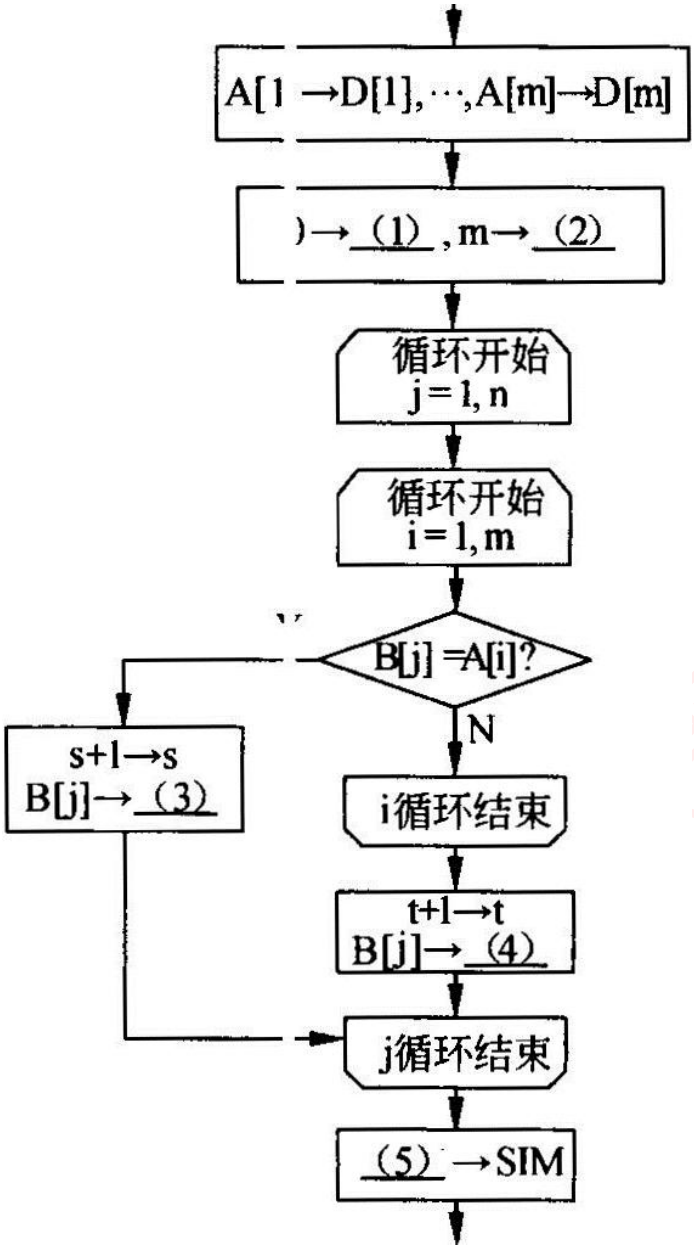
【说明】

两个包含有限个元素的非空集合 A、B 的相似度定义为 |A∩B|/|A∪B|，即它们的交 集大小(元素个数）与并集大小之比。

以下的流程图计算两个非空整数集合(以数组表示）的交集和井集，并计算其相似度。已知整数组 A[1:m]和 B[l:n]分别存储了集合 A 和 B 的元素(每个集合中包含的元素各不相同)，其交集存放于数组 C[1:s]，并集存放于数组 D[1:t]，集合 A 和 B 的相似度存放于 SIM。

例如，假设 A={ 1， 2，3，4}，B={ 1， 4，5，的，则 C={1， 4}，D={1， 2，3，4，5，6}，A 与 B 的相似度 SIM=1/3。

【流程图】



●

阅读以下说明和 C 函数，填充函数中的主缺，将解答填入答题纸的对应栏内。

【说明】

下面的函数 sort(int n，int a[])对保存在数在 a 中的整数序列进行非递减排序。由于该 序列中的元素在一定范围内重复取值，因此排序方法是先计算出每个元素出现的次数并记录在数组 b 中，再从小到大顺序地排列各元素即可得到一个非递减有序序列。例如， 对于序列 6，5，6，9，6，4，8，6，5. 其元素在整数区间 [4，9]内取值，因此使数组元素 b[0] ～b[5]的下标。0～5 分别对应数值 4～9. 顺序地扫描序列的每一个元素并累计其出现的次数，即将 4 的个数记入 b[0]， 5 的个数记 b[1]，依此类推，9 的个数记入 b[5]最后依次判断数组 b 的每个元素值，并将相应个数的数值顺序地写入结果序列即可。

对于上例，所得数组 b 的各个元素值如下:

b[0]	b[1]	b[2]	b[3]	b[4]	b[5]
1	2	3	4	5	6

那么在输出序列中写入 1 个 4、2 个 5、4 个 6、1 个 8、1 个 9，即得 4，5，5，6，6，6，6，8，9，
从而完成排序处理。
【C 函数】

```
void sort(int n ， int a[ ])
{   int  *b;
int i ,k ,number;
int minimum  = a[0] , maximum = a[0];
/* minimum 和 maximum 分别表示数组 a 的最小、最大元素值*/
for(i=1;i<n;i++){
    If( （1） ) minimum=a[j];
    Else
      If( （2） ) maximum=a[i];
   }
number = maximum - minimum + 1;
if  (number<=1)   return;
b =  (int *)calloc (number ， sizeof(int)) ;

if (!b) return;

for(i=0;i<n;i++){  /*     计算数组 a 元素值出现的次数并计入数组 b */


    K=a[i]-minimum; ++b[k];
}
/*按次序在数组 a 中写入排好的序列*/
i=（3） ;
for( k=0;  k<number;  k++  )
for(;  （4）  ; --b[k]   )
a [i++]  = minimum  +（5）;

}
```

- 阅读以下说明和 C 代码，填充代码中的空缺，将解答填入答题纸的对应栏内。

【说明 1】
下面的函数 countChar(char *text)统计字行串 text 中不同的英文字母数和每个英文字母出现的次数(英文字母不区分大小写)。

【c 代码 1】

```
int countChar( char *text )
{
    int i，sum = 0;   /* sum 保存不同的英文字母数*/
    char *ptr;
```

```
    int  c[26]  =  {0}; /*数组 c 保存每个英文字母出现的次数*/
    /*c[0]记录字母 A 或 a 的次数，c[1] 记录字母 B 或 b 的次数，依此类推
*/

      ptr =  (1)  ;/*ptr 初始时指向字符串的首字符*/
    while  (*ptr)  {
        if  (  isupper(*ptr)  )
          c[*ptr  -  'A']++;
        e1se
          if  (  islower(*ptr)  )
            c[*ptr  - 'a'] ++;
          (2)  ;  /*指向下一个字符*/
    }

    for(i=0;i < 26;i++)
        if (3) sum++;
      return  sum;
    }
```

【说明 2】

将下面 C 代码 2 中的空缺补全后运行，使其产生以下输出。

```
f2:f2:f2:2
f3:f3: 1
```

【C 代码 2】

```
*include  <stdio.h>
int f1 (int  (*f) (int)) ;
int f2 (int) ;
int f3 (int) ;

int main __(3)__
{
    printf("%d\n" , f1( (4) ));
    printf("%d\n" , f1( (5) ));
```

```
    return  0;
}
int  f1 ( int  (*f) (int)  )
{
    int  n  = 0;
    /*通过函数指针实现函数调用，以返回值作为循环条件*/

while  (__(6)__)  n++;
return   n;
}
int f2(int n)
{
    printf("f2: ");
    return  n*n-4;
}

int f3 (int n)
{
    printf("f3: ");
    return  n-l;
}
```

● 阅读以下说明和 C 程序，填充程序中的空缺，将解答填入答题纸的对应栏内。

【说明】

正整数 n 若是其平方数的尾部，则称 n 为同构数。例如，6 是其平方数 36 的尾部，76 是其平方数 5776 的尾部，6 与 76 都是同构数。下面的程序求解不超过 10000 的所有同构数。

已知一位的同构数有三个: 1，5，6，到此二位同构数的个位数字只可能是 1，5，6 这三个数字。依此类推，更高位数同构数的个位数字也只可能是 1，5，6 这三个数字。

下面程序的处理思路是:对不超过 10000 的每一个整数 a，判断其个位数字，若为 1、5 或 6，则将 a 转换为字符串 as，然后对 a 进行平方运算，并截取其尾部与 as 长度相等的若干字符形成字符串后与出比较，根据古们相等与否来断定 a 是否为同构数。

【C 程序】

```
#include<stdio.h>
```

```
#include<stdlib.h>
#include<string.h>
int  myitoa(int ，char  *);  /*将整数转换为字符串*/
/* right 取得指定字符串尾部长度为 lergth 的子串，返回所得子串的首字符指针*/
char  *right(char* ，  int  length);

int main __(4)__
{
    int a ，t;  int  len;
    char as[10] ，rs[20];

    printf("[1 ，l0000]内的同构数: \r");
    for(a=1;a<=10000;a++)  {
       t =  ___(1)___;  /*取整数 a 的个位数字*/
       if (t!=1&& t!=5 && t!=6)  continue;
       len  =  myitoa(a ，as);  /*数 a 转换为字符串，存入 as */
       myitoa(a*a ， rs);  /*数 a 的平方转换为字符串，存入 rs */
/*比较字符串 as 与 rs 末尾长度为 len 的子串是否相等*/
    if (  strcmp (as ，__(2)__)==0  ) /*若相同则是同构数并输出*/
       printf("%s 的平方为%s\n" ，as，rs);
  }
  return 0;
}
int  myitoa(int num ，char *s)  /*将整数 num 转换为字符串存入 s */
{
int i ，n  =  0;
  char ch;

/*从个位数开始，取 num 的每一位数字转换沟字符后放入 s[] */
  while  (num)  {
      s[n++]  =  ( 3 )  + '0' ;
      num = num/10;
  }
  s[n]='\0';
      for(i=0;  i<n/2;  i++)  {  /*将 s 中的字符串逆置*/
      __(4)__; s[i] = s[n-i-1]; 5 [n-i-1] = ch;
```

```
}
return  n;  /*返回输入参数 num 的位数*/
}
char *right(char *.ms  , int  length)
/*取字符串 ms 尾部长度为 length 的子串，返回所得子串的首字符指针*/
{
    int i;

    for( ; *ms; ms++);              /*使 ms 到达原字符串的尾部事*/
    for(  i=0;  i<length;  ____(5)____);/*使 ms 指向所得子串的首部字符*/
    return  ms;

}
```

●

阅读以下说明和 C++代码，填充程序中的空缺，将解答填入答题纸的对应栏内。

【说明】

某应急交通控制系统(TraficControlSystem)在红灯时控制各类车辆 (Vehicle)的通行，其类图如图 5-1 所示，在紧急状态下应急车辆在红灯时可通行，其余车辆按正常规则通行。

下面的 C++代码实现以上设计，请完善其中的空缺。

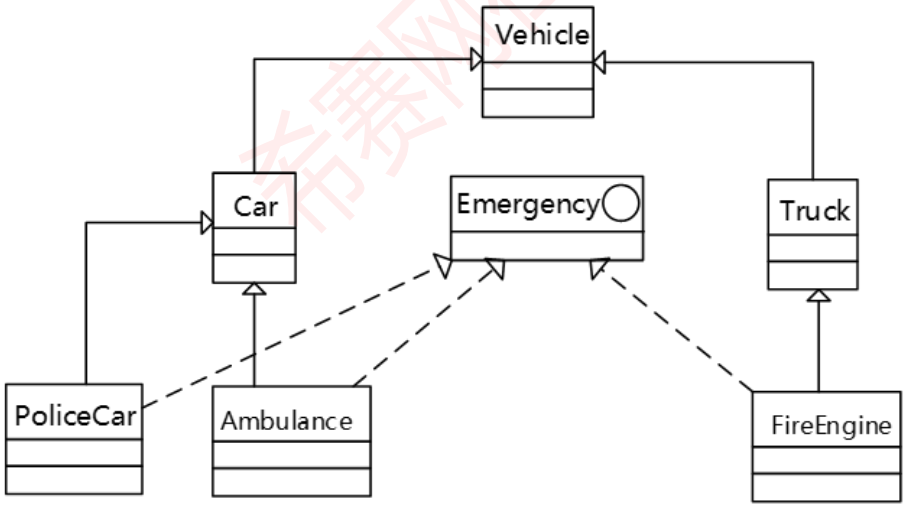


图5-1 类图

【C++代码】

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

class Vehicle { /*抽象基类，车辆*/ public: virtual void run( ) = 0; }; class Emergency { /*抽象基类，可在红灯时通行的接口，函数均为纯虚函数 */ public: (1) = 0; // isEmergent ( )函数接口 (2) = 0; // runRedLight( ) 函数接口 }; class Car: public Vehicle { public: ～Car ( ) { } void run ( ) { /* 代码略*/} }; class Truck: public Vehicle { public: ～Truck( ) { } void run ( ) { /*代码略*/ } }; class PoliceCar: (3) { private: bool isEmergency; public: PoliceCar( ) : Car( ) , Emergency( ){ this->isEmergency = false; } PoliceCar(bool b) : Car( ) , Emergency( ) { this->isEmergency = b;} ～policeCar ( ) { } bool isEmergent ( ) { return （4）; } void runRedLight( ) { /*代 码略*/ } }; /*类 Ambulance、 FireEngine 实现代码略 */ class TraficControlSystern { /*交通控制类*/ private: Vehicle* v[24}; int nurnVeh:cles; /*在构造函数中设置初始值为 0*/ public: void control__(5)__ { //控制在紧急情况下应急车辆红灯通行，其他情况按常规通行 for (int i = 0; i < numVehicles ; i++) { Emergency * ev = dynamic_cast<Emergency*>(v[i]); if (ev != 0) (5) ->runRedLight __(6)__ ; else (6) ->run __(7)__ ; } } void add(Vehicle * vehicle) { v[numVehicles++] = vehicle; / *添加车辆./ void shutDown__(8)__ { for (int i = 0; i < numVehicles; i++) { delete v[i]; } } } ; int main( ) { TraficControlSystern. tcs = new TraficControlSystern; tcs->add(new Car( )); tcs->add(new PoliceCar( )); tcs->add(new Ambulance( ));tcs->add(new Ambulance(true)); tcs->add(new FireEngine(true)); tcs->add(new FireEngine( )); tcs->add(new Truck( )); tcs->control( ); tcs->shutDown( ); delete tcs; }

● 阅读以下说明和 Java 代码，填充程序中的空缺，将解答填入答题纸的对应栏内。

【说明】

某应急交通控制系统(TraficControlSystem)在红灯时控制各类车辆 (Vehicle)的通行，其类图如图 6-1 所示，在紧急状态下应急车辆在红灯时可通行，其余车辆按正常规则通行。

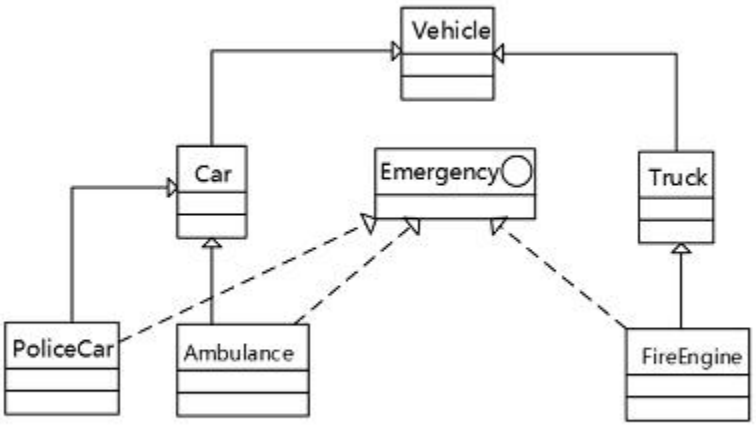


图6-1 类图

下面的 Java 代码实现以上设计，请完善其中的空缺。

【Java 代码 】

```
abstract class Vehicle {
```

```
    public Vehicle( ) { )
    abstract  void  run( );
};
interface  Emergency  {
        (1)   ;
        (2)   ;
};
class  Car  extends  Vehicle  {
      public  Car( )  {      )
      void  run ( ) {  /*代码略*/
};
class  Truck  extends  Vehicle  {
       public  Truck( )  { )
       void  run   ( ) {      /*代码略*/
};

class PoliceCar    (3)    {
      boolean  isEmergency  =  false;
      public  PoliceCar__(6)__    {    )
      public   PoliceCar(boolean   b) {   this.isEmergency=b;   }
      public   boolean   isEmergent( ) { return     ( 4 )   ;   }
      public  void  runRedLight( )  {    /*代码略*/  }
};
/*类 h 由 ulance 、 FireEngine  实现代码略叫*/
public   class   TraficControlSystem     { /*交通控制类*/
      private  Vehicle[ ]  v  =  new  Vehicle[24];
      int  numVehicles;
      public  void control( )  {
          for  (int  i = 0;  i <  numVehicles;  i++)  {
                if  (v[i]  instanceof  EmErgency  &&   ((Emergency)v[i]).
                isEmergent__(7)__)  {
                  (     5    ) .runRedLigh   (    );
                 }    else
                  ( 6 )  .run ( ) ;
          }
      }
          void  add(Vehicle  vehicle)   {  v[numVehicles++]  = vehiclei} /*添加车辆*/
          void   shutDown__(8)__     {   /*代码略* /}
          public  static void  main (Str :.ng [ ]  args)   {
                TraficControlSystem    tcs  =  new   TraficControlSystem__(9)__;
                tcs.add(new     Car__(10)__};
                tcs.add(new  PoliceCar__(11)__;
                tcs.add(new  Ambulance__(12)__;
                tcs.add(new   Ambulance(t:ue));
                tcs.add(new FireEngine( :rue));
                tcs.add(new Truck__(13)__);
                tcs.add(new FireEngine( );
                tcs.control__(14)__;
                tcs.shutDown__(15)__;
      }
```

```
}
```