华为笔试

1、局部变量能否和全局变量重名？

答：能，局部会屏蔽全局。要用全局变量，需要使用"::"

局部变量可以与全局变量同名，在函数内引用这个变量时，会用到同名的局部变量，而不会用到全局变量。对于有些编译器而言，在同一个函数内可以定义多个同名的局部变量，比如在两个循环体内都定义一个同名的局部变量，而那个局部变量的作用域就在那个循环体内。

2、如何引用一个已经定义过的全局变量？

答：extern

可以用引用头文件的方式，也可以用 extern 关键字，如果用引用头文件方式来引用某个在头文件中声明的全局变理，假定你将那个变写错了，那么在编译期间会报错，如果你用 extern 方式引用时，假定你犯了同样的错误，那么在编译期间不会报错，而在连接期间报错。

3、全局变量可不可以定义在可被多个.C 文件包含的头文件中？为什么？

答：可以，在不同的 C 文件中以 static 形式来声明同名全局变量。

可以在不同的 C 文件中声明同名的全局变量，前提是其中只能有一个 C 文件中对此变量赋初值，此时连接不会出错。

4、语句 for(; 1 ;)有什么问题？它是什么意思？

答：无限循环，和 while(1)相同。

5、do……while 和 while……do 有什么区别？

答：前一个循环一遍再判断，后一个判断以后再循环。

6、请写出下列代码的输出内容

```
#i nclude<stdio.h>
main()
{
int a,b,c,d;
a=10;
b=a++;
c=++a;
d=10*a++;
printf("b, c, d: %d, %d, %d", b, c, d) ;
return 0;
}
```

答：10，12，120

一、判断题（对的写 T，错的写 F 并说明原因，每小题 4 分，共 20 分）

1、有数组定义 int a[2][2]={{1},{2,3}};则 a[0][1]的值为 0。（ ）

2、int (*ptr) (),则 ptr 是一维数组的名字。（ ）

3、指针在任何情况下都可进行>, <, >=, <=, = =运算。（ ）

4、switch(c) 语句中 c 可以是 int, long, char, float, unsigned int 类型。（ ）

5、#define print(x) printf(" the no, " #x ",is ")

二、填空题（共 30 分）

1、在 windows 下，写出运行结果，每空 2 分，共 10 分。

```
char str[ ]= " Hello " ;

char *p=str;

int n=10;

sizeof(str)=(          )

sizeof(p)=(            )

sizeof(n)=(            )

void func(char str[100])
{      }
```

```
sizeof(str)=(          )
```

2、void setmemory(char **p, int num)

```
{ *p=(char *) malloc(num);}

void test(void)

{   char *str=NULL;

     getmemory(&str,100);

     strcpy(str, " hello " );

     printf(str);

}
```

运行 test 函数有什么结果？
（ ）10 分

3、设 int arr[]={6,7,8,9,10};

```
          int *ptr=arr;

     *(ptr++)+=123;
```

```
printf(" %d,%d ",*ptr,*(++ptr));
```

(
) 10 分

二、编程题（第一小题 20，第二小题 30 分）

1、 不使用库函数，编写函数 int strcmp(char *source, char *dest)

相等返回 0，不等返回-1；

2、 写一函数 int fun(char *p)判断一字符串是否为回文，是返回 1，不是返回 0，出错返回-1

```
nt arr[]={6,7,8,9,10};
int *ptr=arr;
*(ptr++)+=123;
printf(" %d,%d ",*ptr,*(++ptr));
```

请问输出是什么？

解答：

这道题的答案取自于编译器,因为不同的编译器有不同的压栈顺序，一般情况下是从右往左压，即答案为 8，8 。但我们也不否认出现 7，8 或 8，9 的情况。

1. 请你分别画出 OSI 的七层网络结构图和 TCP/IP 的五层结构图。
2. 请你详细地解释一下 IP 协议的定义，在哪个层上面？主要有什么作用？TCP 与 UDP 呢？
3. 请问交换机和路由器各自的实现原理是什么？分别在哪个层次上面实现的？
4. 请问 C++的类和 C 里面的 struct 有什么区别？
5. 请讲一讲析构函数和虚函数的用法和作用。
6. 全局变量和局部变量有什么区别？是怎么实现的？操作系统和编译器是怎么知道的？
7. 8086 是多少位的系统？在数据总线上是怎么实现的？

找错

```
Void test1()
{
    char string[10];                [11]
```

```
    char* str1="0123456789";
    strcpy(string, str1);
}


Void test2()
{
    char string[10], str1[10];
    for(I=0; I<10;I++)                    I 未定义
    {
        str1[i] ='a';                         I
    }
    strcpy(string, str1);                  运行有问题?
}


Void test3(char* str1)
{
       char string[10];
       if(strlen(str1)<=10)
       {
       strcpy(string, str1);
       }
}
```

2.　　找错

```
#define MAX_SRM 256

DSN get_SRM_no()
{
               static int SRM_no;
               int I;
               for(I=0;I{
SRM_no %= MAX_SRM;
if(MY_SRM.state==IDLE)
{
       break;
}
}
if(I>=MAX_SRM)
return (NULL_SRM);
else
```

```
return SRM_no;
}
```

3.　　写出程序运行结果

```
int sum(int a)
{
                auto int c=0;
                static int b=3;
c+=1;
b+=2;
return(a+b+C);
}

void main()
{
                int I;
int a=2;
for(I=0;I<5;I++)
{
        printf("%d,", sum(a));
}
}
```

4.

```
int func(int a)
{
        int b;
        switch(a)
        {
                case 1: 30;
                case 2: 20;
                case 3: 16;
                default: 0
}
return b;
}
```

则 func(1)=?

5:

```
int a[3];
a[0]=0; a[1]=1; a[2]=2;
int *p, *q;
p=a;
```

q=&a[2];
则 a[q-p]=? 2

6.
定义 int **a[3][4], 则变量占有的内存空间为：_____4

7.
编写一个函数，要求输入年月日时分秒，输出该年月日时分秒的下一秒。如输入 2004 年 12 月 31 日 23 时 59 分 59 秒，则输出 2005 年 1 月 1 日 0 时 0 分 0 秒。

1、一个学生的信息是：姓名，学号，性别，年龄等信息，用一个链表，把这些学生信息连在一起， 给出一个 age, 在些链表中删除学生年龄等于 age 的学生信息。

```
#include "stdio.h"
#include "conio.h"

struct stu{
	char name[20];
	char sex;
	int no;
	int age;
	struct stu * next;
}*linklist;
struct stu *creatlist(int n)
{
	int i;
	//h 为头结点，p 为前一结点，s 为当前结点
	struct stu *h,*p,*s;
	h = (struct stu *)malloc(sizeof(struct stu));
	h->next = NULL;
	p=h;
	for(i=0;i<n;i++)
	{
		s = (struct stu *)malloc(sizeof(struct stu));
		p->next = s;
		printf("Please input the information of the student: n
ame sex no age \n");
		scanf("%s %c %d %d",s->name,&s->sex,&s->no,&s->age);
		s->next = NULL;
		p = s;
	}
	printf("Create successful!");
	return(h);
}
void deletelist(struct stu *s,int a)
```

```
{
  struct stu *p;
  while(s->age!=a)
  {
    p = s;
    s = s->next;
  }
  if(s==NULL)
    printf("The record is not exist.");
  else
  {
    p->next = s->next;
    printf("Delete successful!");
  }
}
void display(struct stu *s)
{
  s = s->next;
       while(s!=NULL)
       {
                printf("%s %c %d %d\n",s->name,s->sex,s->no,s->age);
                s = s->next;
       }
}
int main()
{
       struct stu *s;
  int n,age;
  printf("Please input the length of seqlist:\n");
  scanf("%d",&n);
       s = creatlist(n);
       display(s);
  printf("Please input the age:\n");
  scanf("%d",&age);
  deletelist(s,age);
  display(s);
       return 0;
}
```

2、实现一个函数，把一个字符串中的字符从小写转为大写。

```
#include "stdio.h"
#include "conio.h"
```

```
void uppers(char *s,char *us)
{
	for(;*s!='''\0''';s++,us++)
	{
		if(*s>='''a''''&&*s<='''z''')
			*us = *s-32;
		else
			*us = *s;
	}
	*us = ''''\0''''';
}
int main()
{
	char *s,*us;
	char ss[20];
	printf("Please input a string:\n");
	scanf("%s",ss);
	s = ss;
	uppers(s,us);
	printf("The result is:\n%s\n",us);
	getch();
}
```

1、局部变量能否和全局变量重名？

答：能，局部会屏蔽全局。要用全局变量，需要使用"::"

局部变量可以与全局变量同名，在函数内引用这个变量时，会用到同名的局部变量，而不会用到全局变量。对于有些编译器而言，在同一个函数内可以定义多个同名的局部变量，比如在两个循环体内都定义一个同名的局部变量，而那个局部变量的作用域就在那个循环体内。

2、如何引用一个已经定义过的全局变量？

答：extern

可以用引用头文件的方式，也可以用 extern 关键字，如果用引用头文件方式来引用某个在头文件中声明的全局变理，假定你将那个变写错了，那么在编译期间会报错，如果你用 extern 方式引用时，假定你犯了同样的错误，那么在编译期间不会报错，而在连接期间报错。

3、全局变量可不可以定义在可被多个.C 文件包含的头文件中？为什么？

答：可以，在不同的 C 文件中以 static 形式来声明同名全局变量。

可以在不同的 C 文件中声明同名的全局变量，前提是其中只能有一个 C 文件中对此变量赋初值，此时连接不会出错。

4、语句 for(; 1 ;)有什么问题？它是什么意思？

答：无限循环，和 while(1)相同。

5、do……while 和 while……do 有什么区别？

答：前一个循环一遍再判断，后一个判断以后再循环。

6、请写出下列代码的输出内容

```
#i nclude<stdio.h>
main()
{
int a,b,c,d;
a=10;
b=a++;
c=++a;
d=10*a++;
printf("b，c，d：%d，%d，%d"，b，c，d）；
return 0;
}
```

答：10，12，120

1.static 有什么用途？（请至少说明两种）

1)在函数体，一个被声明为静态的变量在这一函数被调用过程中维持其值不变。

2）在模块内（但在函数体外），一个被声明为静态的变量可以被模块内所用函数访问，但不能被模块外其它函数访问。它是一个本地的全局变量。

3）在模块内，一个被声明为静态的函数只可被这一模块内的其它函数调用。那就是，这个函数被限制在声明它的模块的本地范围内使用

2.引用与指针有什么区别？

1）引用必须被初始化，指针不必。

2）引用初始化以后不能被改变，指针可以改变所指的对象。

3）不存在指向空值的引用，但是存在指向空值的指针。

3.描述实时系统的基本特性

在特定时间内完成特定的任务，实时性与可靠性。

4.全局变量和局部变量在内存中是否有区别？如果有，是什么区别？

全局变量储存在静态数据库，局部变量在堆栈。

5.什么是平衡二叉树？

左右子树都是平衡二叉树 且左右子树的深度差值的绝对值不大于1。

6.堆栈溢出一般是由什么原因导致的？

没有回收垃圾资源。

7.什么函数不能声明为虚函数？

constructor 函数不能声明为虚函数。

8.冒泡排序算法的时间复杂度是什么？

时间复杂度是 O(n^2)。

9.写出 float x 与“零值”比较的 if 语句。

if(x>0.000001&&x<-0.000001)

10.Internet 采用哪种网络协议？该协议的主要层次结构？

Tcp/Ip 协议

主要层次结构为： 应用层/传输层/网络层/数据链路层/物理层。

11.Internet 物理地址和 IP 地址转换采用什么协议？

ARP (Address Resolution Protocol)（地址解析協議）

12.IP 地址的编码分为哪俩部分？

IP 地址由两部分组成，网络号和主机号。不过是要和“子网掩码”按位与上之后才能区分哪些是网络位哪些是主机位。

13.用户输入 M,N 值，从 1 至 N 开始顺序循环数数，每数到 M 输出该数值，直至全部输出。写出 C 程序。

循环链表，用取余操作做

14.不能做 switch()的参数类型是：

switch 的参数不能为实型。

1.写出判断 ABCD 四个表达式的是否正确，若正确，写出经过表达式中 a 的值(3 分)

```
int a = 4;
(A)a += (a++); (B) a += (++a) ;(C) (a++) += a;(D) (++a) += (a++);
a = ?
```

答：C 错误，左侧不是一个有效变量，不能赋值，可改为(++a) += a;

改后答案依次为 9,10,10,11

2.某 32 位系统下，C++程序，请计算 sizeof 的值(5 分).

```
char str[] = “http://www.ibegroup.com/”
char *p = str ;
int n = 10;
请计算
sizeof (str ) = ? (1)
sizeof ( p ) = ? (2)
sizeof ( n ) = ? (3)
void Foo ( char str[100]){
请计算
sizeof( str ) = ? (4)
}
void *p = malloc( 100 );
请计算
sizeof ( p ) = ? (5)
```

答：(1) 17 (2) 4 (3) 4 (4) 4 (5) 4

3. 回答下面的问题. (4 分)
(1).头文件中的 ifndef/define/endif 干什么用? 预处理
答: 防止头文件被重复引用
(2). #i nclude 和 #i nclude “filename.h” 有什么区别?
答: 前者用来包含开发环境提供的库头文件, 后者用来包含自己编写的头文件。

(3).在 C++ 程序中调用被 C 编译器编译后的函数, 为什么要加 extern “C” 声明?
答: 函数和变量被 C++编译后在符号库中的名字与 C 语言的不同, 被 extern "C"修饰的变
量和函数是按照 C 语言方式编译和连接的。由于编译后的名字不同, C++程序不能直接调
用 C 函数。C++提供了一个 C 连接交换指定符号 extern“C”来解决这个问题。

(4). switch()中不允许的数据类型是?
答: 实型

4. 回答下面的问题(6 分)
(1).Void GetMemory(char **p, int num){
*p = (char *)malloc(num);
}
void Test(void){
char *str = NULL;
GetMemory(&str, 100);
strcpy(str, "hello");
printf(str);
}
请问运行 Test 函数会有什么样的结果?
答: 输出“hello”
(2). void Test(void){
char *str = (char *) malloc(100);
strcpy(str, “hello”);
free(str);
if(str != NULL){
strcpy(str, “world”);
printf(str);
}
}
请问运行 Test 函数会有什么样的结果?
答: 输出“world”
(3). char *GetMemory(void){
char p[] = "hello world";
return p;
}
void Test(void){

```
char *str = NULL;
str = GetMemory();
printf(str);
}
```

请问运行 Test 函数会有什么样的结果?
答：无效的指针，输出不确定

5. 编写 strcat 函数(6 分)
已知 strcat 函数的原型是 char *strcat (char *strDest, const char *strSrc);
其中 strDest 是目的字符串，strSrc 是源字符串。
（1）不调用 C++/C 的字符串库函数，请编写函数 strcat
答：
VC 源码：

```
char * __cdecl strcat (char * dst, const char * src)
{
char * cp = dst;
while( *cp )
cp++; /* find end of dst */
while( *cp++ = *src++ ) ; /* Copy src to end of dst */
return( dst ); /* return dst */
}
```

（2）strcat 能把 strSrc 的内容连接到 strDest，为什么还要 char * 类型的返回值?
答：方便赋值给其他变量

6.MFC 中 CString 是类型安全类么?
答：不是，其它数据类型转换到 CString 可以使用 CString 的成员函数 Format 来转换

7.C++中为什么用模板类。
答：(1)可用来创建动态增长和减小的数据结构
（2）它是类型无关的，因此具有很高的可复用性。
（3）它在编译时而不是运行时检查数据类型，保证了类型安全
（4）它是平台无关的，可移植性
（5）可用于基本数据类型

8.CSingleLock 是干什么的。
答：同步多个线程对一个数据类的同时访问

9.NEWTEXTMETRIC 是什么。
答：物理字体结构，用来设置字体的高宽大小

10.程序什么时候应该使用线程，什么时候单线程效率高。
答：1. 耗时的操作使用线程，提高应用程序响应
2. 并行操作时使用线程，如 C/S 架构的服务器端并发线程响应用户的请求。
3. 多 CPU 系统中，使用线程提高 CPU 利用率

4. 改善程序结构。一个既长又复杂的进程可以考虑分为多个线程，成为几个独立或半独
立的运行部分，这样的程序会利于理解和修改。
其他情况都使用单线程。

11.Windows 是内核级线程么。
答：见下一题

12.Linux 有内核级线程么。
答：线程通常被定义为一个进程中代码的不同执行路线。从实现方式上划分，线程有两
种类型：“用户级线程”和“内核级线程”。 用户线程指不需要内核支持而在用户程序
中实现的线程，其不依赖于操作系统核心，应用进程利用线程库提供创建、同步、调度
和管理线程的函数来控制用户线程。这种线程甚至在象 DOS 这样的操作系统中也可实现
，但线程的调度需要用户程序完成，这有些类似 Windows 3.x 的协作式多任务。另外一
种则需要内核的参与，由内核完成线程的调度。其依赖于操作系统核心，由内核的内部
需求进行创建和撤销，这两种模型各有其好处和缺点。用户线程不需要额外的内核开支
，并且用户态线程的实现方式可以被定制或修改以适应特殊应用的要求，但是当一个线
程因 I/O 而处于等待状态时，整个进程就会被调度程序切换为等待状态，其他线程得不
到运行的机会；而内核线程则没有各个限制，有利于发挥多处理器的并发优势，但却占
用了更多的系统开支。
Windows NT 和 OS/2 支持内核线程。Linux 支持内核级的多线程

13.C++中什么数据分配在栈或堆中，New 分配数据是在近堆还是远堆中？
答：栈：存放局部变量，函数调用参数,函数返回值，函数返回地址。由系统管理
堆：程序运行时动态申请，new 和 malloc 申请的内存就在堆上

14.使用线程是如何防止出现大的波峰。
答：意思是如何防止同时产生大量的线程，方法是使用线程池，线程池具有可以同时提
高调度效率和限制资源使用的好处，线程池中的线程达到最大数时，其他线程就会排队
等候。

15 函数模板与类模板有什么区别？
答：函数模板的实例化是由编译程序在处理函数调用时自动完成的，而类模板的

实例化
必须由程序员在程序中显式地指定。

16 一般数据库若出现日志满了，会出现什么情况，是否还能使用？
答：只能执行查询等读操作，不能执行更改，备份等写操作，原因是任何写操作都要记
录日志。也就是说基本上处于不能使用的状态。

17 SQL Server 是否支持行级锁，有什么好处？
答：支持，设立封锁机制主要是为了对并发操作进行控制，对干扰进行封锁，保证数据
的一致性和准确性，行级封锁确保在用户取得被更新的行到该行进行更新这段时间内不
被其它用户所修改。因而行级锁即可保证数据的一致性又能提高数据操作的迸发性。

18 如果数据库满了会出现什么情况，是否还能使用？
答：见 16

19 关于内存对齐的问题以及 sizof()的输出
答：编译器自动对齐的原因：为了提高程序的性能，数据结构（尤其是栈）应该尽可能
地在自然边界上对齐。原因在于，为了访问未对齐的内存，处理器需要作两次内存访问
；然而，对齐的内存访问仅需要一次访问。

20 int i=10, j=10, k=3; k*=i+j; k 最后的值是？
答：60，此题考察优先级，实际写成： k*=(i+j);，赋值运算符优先级最低

21.对数据库的一张表进行操作,同时要对另一张表进行操作,如何实现?
答：将操作多个表的操作放入到事务中进行处理

22.TCP/IP 建立连接的过程?(3-way shake)
答：在 TCP/IP 协议中，TCP 协议提供可靠的连接服务，采用三次握手建立一个连接。
第一次握手：建立连接时，客户端发送 syn 包(syn=j)到服务器，并进入 SYN_SEND 状
态，等待服务器确认；
第二次握手：服务器收到 syn 包，必须确认客户的 SYN（ack=j+1），同时自己也发送一个
SYN 包（syn=k），即 SYN+ACK 包，此时服务器进入 SYN_RECV 状态；
第三次握手：客户端收到服务器的 SYN＋ACK 包，向服务器发送确认包 ACK(ack=k+1)
，此包发送完毕，客户端和服务器进入 ESTABLISHED 状态，完成三次握手。

23.ICMP 是什么协议,处于哪一层?
答：Internet 控制报文协议，处于网络层（IP 层）

24.触发器怎么工作的?
答：触发器主要是通过事件进行触发而被执行的，当对某一表进行诸如 UPDATE、 INSERT
、 DELETE 这些操作时，数据库就会自动执行触发器所定义的 SQL 语句，从而确保对数
据的处理必须符合由这些 SQL 语句所定义的规则。

25.winsock 建立连接的主要实现步骤?
答：服务器端：socker()建立套接字，绑定（bind）并监听（listen），用 accept（）
等待客户端连接。
客户端：socker()建立套接字，连接（connect）服务器，连接上后使用 send()和 recv（
），在套接字上写读数据，直至数据交换完毕，closesocket()关闭套接字。
服务器端：accept（）发现有客户端连接，建立一个新的套接字，自身重新开始等待连
接。该新产生的套接字使用 send()和 recv（）写读数据，直至数据交换完毕，closesock
et()关闭套接字。

26.动态连接库的两种方式?
答：调用一个 DLL 中的函数有两种方法：
1. 载入时动态链接（load-time dynamic linking），模块非常明确调用某个导出函数
，使得他们就像本地函数一样。这需要链接时链接那些函数所在 DLL 的导入库，导入库向
系统提供了载入 DLL 时所需的信息及 DLL 函数定位。
2. 运行时动态链接（run-time dynamic linking），运行时可以通过 LoadLibrary 或 Loa
dLibraryEx 函数载入 DLL。DLL 载入后，模块可以通过调用 GetProcAddress 获取 DLL 函数的
出口地址，然后就可以通过返回的函数指针调用 DLL 函数了。如此即可避免导入库文件了
。

27.IP 组播有那些好处?
答：Internet 上产生的许多新的应用，特别是高带宽的多媒体应用，带来了带宽的急剧
消耗和网络拥挤问题。组播是一种允许一个或多个发送者（组播源）发送单一的数据包
到多个接收者（一次的，同时的）的网络技术。组播可以大大的节省网络带宽，因为无
论有多少个目标地址，在整个网络的任何一条链路上只传送单一的数据包。所以说组播
技术的核心就是针对如何节约网络资源的前提下保证服务质量。

找错

```
Void test1()
{
    char string[10];
    char* str1="0123456789";
strcpy(string, str1);
}


Void test2()
{
    char string[10], str1[10];
for(I=0; I<10;I++)
{
        str1[i] ='a';
}
strcpy(string, str1);
}

Void test3(char* str1)
{
    char string[10];
    if(strlen(str1)<=10)
{
    strcpy(string, str1);
}
}
```

2. 找错

```
#define MAX_SRM 256

DSN get_SRM_no()
{
        static int SRM_no;
        int I;
        for(I=0;I{
SRM_no %= MAX_SRM;
if(MY_SRM.state==IDLE)
{
    break;
}
}
if(I>=MAX_SRM)
return (NULL_SRM);
else
```

```
return SRM_no;
}
```

3.　写出程序运行结果

```
int sum(int a)
{
        auto int c=0;
        static int b=3;
c+=1;
b+=2;
return(a+b+C);
}

void main()
{
        int I;
int a=2;
for(I=0;I<5;I++)
{
    printf("%d,", sum(a));
}
}
```

4.

```
int func(int a)
{
    int b;
    switch(a)
    {
        case 1: 30;
        case 2: 20;
        case 3: 16;
        default: 0
}
return b;
}
```

则 func(1)=?

5:

```
int a[3];
a[0]=0; a[1]=1; a[2]=2;
int *p, *q;
p=a;
```

q=&a[2];
则 a[q-p]=?

6.
定义 int **a[3][4], 则变量占有的内存空间为：_____