每日一题day21_3月27日

一. 单选

1. 对于char*pa[7]的描述中，正确的是（ ）

A pa是一个指向数组的指针，所指向的数组是7个char型元素

B pa是一个指向某数组中第7个元素的指针，该元素是char型变量

C pa[7]表示数组的第7个元素的值，是char型的值

D pa是一个具有7个元素的指针数组，每个元素是一个char型指针

正确答案：D

2. 不能把字符串"HELLO!"赋给数组b的语句是（ ）

A char b[10]={'H'，'E'，'L'，'L'，'O'，'!'，'\0'};

B char b[10];b="HELLO!";

C char b[10];strcpy(b，"HELLO!");

D char b[10]="HELLO!";

正确答案：B

3. 下面程序段的运行结果是（ ）

```
int main（int argc, char *argv[]）
{
    char *s = "abcdefg";
    s += 2;
    fprintf(stderr, "%d\n", s);
    return 0;
}
```

A cde

B 字符"c"

C 字符"c"的地址

D 不确定

正确答案：C

4.
在32位小端的机器上，如下代码输出是什么：

```
char array[12] = {0x01 , 0x02 , 0x03 , 0x04 , 0x05 , 0x06 , 0x07 , 0x08};
 short *pshort = (short *)array;
 int *pint = (int *)array;
 int64 *pint64 = (int64 *)array;
 printf("0x%x , 0x%x , 0x%llx , 0x%llx", *pshort , *(pshort+2) , *pint64 , *(pint+2));
```

A 0x201 , 0x403 , 0x807060504030201 , 0x0

B 0x201 , 0x605 , 0x807060504030201 , 0x0

C 0x201 , 0x605 , 0x4030201 , 0x8070605

D 0x102 , 0x506 , 0x102030405060708 , 0x0

正确答案：B

5.
下面程序应该输出多少？

```
char *c[] = { "ENTER", "NEW", "POINT", "FIRST" };
char **cp[] = { c+3, c+2, c+1, c };
char ***cpp = cp;

int main(void)
{
    printf("%s", **++cpp);
    printf("%s", *--*++cpp+3);
    printf("%s", *cpp[-2]+3);
    printf("%s\n", cpp[-1][-1]+1);
    return 0;
}
```

A POINTERSTEW

B FERSTEPOINW

C NEWPOINTW

D POINTFIREST

正确答案：A

6. 当一个类A 中没有声明任何成员变量与成员函数,这时sizeof(A)的值是多少？

A 1

B 0

C 4

D 运行时错误

正确答案：A

7. 下面有关malloc和new，说法错误的是？

A new 建立的是一个对象， malloc分配的是一块内存.

B new 初始化对象，调用对象的构造函数，对应的delete调用相应的析构函数，malloc仅仅分配内存，free仅仅回收内存

C new和malloc都是保留字，不需要头文件支持

D new和malloc都可用于申请动态内存，new是一个操作符，malloc是是一个函数

正确答案：C

8. 若char是一字节，int是4字节，指针类型是4字节，代码如下：

```
class CTest
{
   public:
      CTest():m_chData('\0'),m_nData(0)
      {
      }
      virtual void mem_fun(){}
   private:
      char m_chData;
      int m_nData;
      static char s_chData;
};
char CTest::s_chData='\0';
```

问：
（1）若按4字节对齐sizeof(CTest)的值是多少？
（2）若按1字节对齐sizeof(CTest)的值是多少？
请选择正确的答案。

A 16 4

B 16 10

C 12 9

D 10 10

正确答案：C

9.
下列代码的输出为：

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

```
int main(void)
{
    vector<int>array;
    array.push_back(100);
    array.push_back(300);
    array.push_back(300);
    array.push_back(500);
    vector<int>::iterator itor;
    for (itor = array.begin(); itor != array.end(); itor++)
    {
        if (*itor == 300)
        {
            itor = array.erase(itor);
        }
    }
    for (itor = array.begin(); itor != array.end(); itor++)
    {
        cout << *itor << " ";
    }
    return 0;
}
```

A 100 300 300 500

B 100 300 500

C 100 500

D 程序错误

正确答案：B

10. 观察下面一段代码:

```
class ClassA
{
public:
    virtual ~ ClassA(){};
    virtual void FunctionA(){};
};
class ClassB
{
public:
   virtual void FunctionB(){};
};
class ClassC : public ClassA,public ClassB
{
    public:
};
```

```
ClassC aObject;
ClassA* pA=&aObject;
ClassB* pB=&aObject;
ClassC* pC=&aObject;
```

关于pA,pB,pC的取值,下面的描述中正确的是:

A pA,pB,pC的取值相同.

B pC=pA+pB

C pA和pB不相同

D pC不等于pA也不等于pB

正确答案：C

二. 编程

1. 洗牌在生活中十分常见，现在需要写一个程序模拟洗牌的过程。 现在需要洗2n张牌，从上到下依次是第1张，第2张，第3张一直到第2n张。首先，我们把这2n张牌分成两堆，左手拿着第1张到第n张（上半堆），右手拿着第n+1张到第2n张（下半堆）。接着就开始洗牌的过程，先放下右手的最后一张牌，再放下左手的最后一张牌，接着放下右手的倒数第二张牌，再放下左手的倒数第二张牌，直到最后放下左手的第一张牌。接着把牌合并起来就可以了。 例如有6张牌，最开始牌的序列是1,2,3,4,5,6。首先分成两组，左手拿着1,2,3；右手拿着4,5,6。在洗牌过程中按顺序放下了6,3,5,2,4,1。把这六张牌再次合成一组牌之后，我们按照从上往下的顺序看这组牌，就变成了序列1,4,2,5,3,6。 现在给出一个原始牌组，请输出这副牌洗牌k次之后从上往下的序列。
输入描述：
第一行一个数T(T ≤ 100)，表示数据组数。对于每组数据，第一行两个数n,k(1 ≤ n,k ≤ 100)，接下来一行有2n个数a1,a2,...,a2n(1 ≤ ai ≤ 1000000000)。表示原始牌组从上到下的序列。
输出描述：
对于每组数据，输出一行，最终的序列。数字之间用空格隔开，不要在行末输出多余的空格。
示例1:
输入
3 3 1 1 2 3 4 5 6 3 2 1 2 3 4 5 6 2 2 1 1 1 1
输出
1 4 2 5 3 6 1 5 4 3 2 6 1 1 1 1

正确答案：

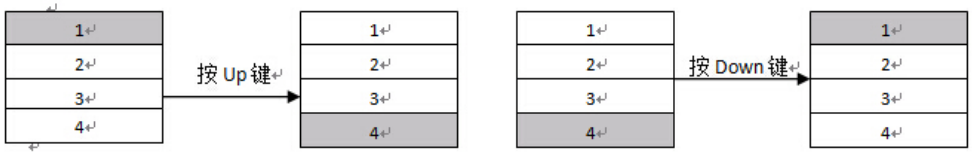
2.

MP3 Player因为屏幕较小，显示歌曲列表的时候每屏只能显示几首歌曲，用户要通过上下键才能浏览所有的歌曲。为了简化处理，假设每屏只能显示4首歌曲，光标初始的位置为第1首歌。

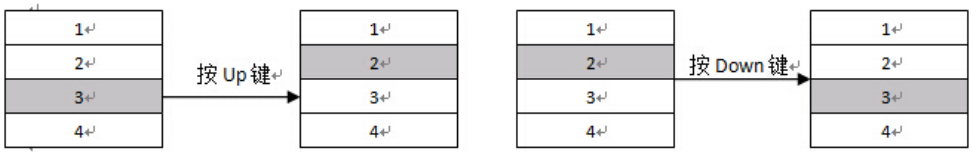
现在要实现通过上下键控制光标移动来浏览歌曲列表，控制逻辑如下：

歌曲总数<=4的时候，不需要翻页，只是挪动光标位置。

光标在第一首歌曲上时，按Up键光标挪到最后一首歌曲；光标在最后一首歌曲时，按Down键光标挪到第一首歌曲。

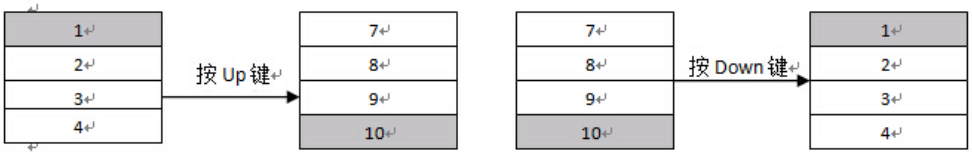


其他情况下用户按Up键，光标挪到上一首歌曲；用户按Down键，光标挪到下一首歌曲。

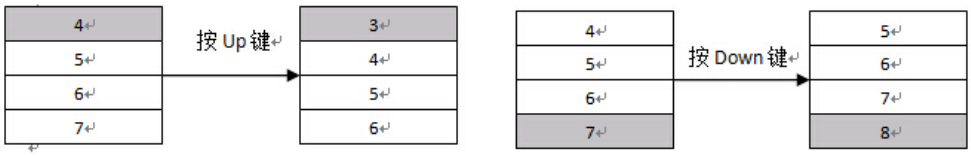


2. 歌曲总数大于4的时候（以一共有10首歌为例）：

特殊翻页：屏幕显示的是第一页（即显示第1 style="color: black;">– 4首）时，光标在第一首歌曲上，用户按Up键后，屏幕要显示最后一页（即显示第7-10首歌），同时光标放到最后一首歌上。同样的，屏幕显示最后一页时，光标在最后一首歌曲上，用户按Down键，屏幕要显示第一页，光标挪到第一首歌上。



一般翻页：屏幕显示的不是第一页时，光标在当前屏幕显示的第一首歌曲时，用户按Up键后，屏幕从当前歌曲的上一首开始显示，光标也挪到上一首歌曲。光标当前屏幕的最后一首歌时的Down键处理也类似。



其他情况，不用翻页，只是挪动光标就行。

输入描述：
输入说明：
1 输入歌曲数量
2 输入命令 U或者D

输出描述：
输出说明
1 输出当前列表
2 输出当前选中歌曲

示例1:
输入
10
UUUU

输出
7 8 9 10
7

正确答案：