期末模拟卷

单选题

2-1以下程序的输出结果是（ ）

```
int main( )

{ int i,  x[3][3]={9,  8,  7,  6,  5,  4,  3,  2,  1},  *p=&x[1][1];
  for(i=0;  i<4;  i+=2)
      cout<<p[i];
}
```

A.52

B.51

C.53

D.97

2-2以下程序不正确的是()

A.`int main(){ class A {int v;}; A a; a.v=3; return 0;}`

B.`int main() {class A {public:int v;A *p;}; A a; a.p=&a; return 0;}`

C. `int main(){class A {public:int v;} A *P=new A; p->v=4;delete p; return 0;}`

D. `int main(){class A{public:int v;A*p;}; A a; a.p=new A; delete a.p; return 0;}`

2-3设有程序段 `int m=20; while (m=0) m=m++;` 则下面描述中正确的是()

A.`while` 循环执行 10 次

B.循环是无限循环

C.循环体语句一次也不执行

D.循环体语句执行一次

2-4若有以下调用语句，则不正确的 `fun( )` 函数的首部是()

```
int main( )
{
   …
 int a[50], n;
   …
 fun(n, &a[9]);
   …
}
```

A.`void fun(int m, int x[ ])`

B.`void fun(int s, int h[41])`

C.`void fun(int p, int *s)`

D.`void fun(int n, int a)`

2-5下列叙述中，正确的是()

A.类的构造函数可以重载

B.类的析构函数可以重载

C.一个类可以没有构造函数

D.一个类可以没有析构函数

2-6下列程序段的输出结果是()

```
void fun(int &x,int &y)
{ cout<<x<<y;
  x=3;
  y=4;}
 int main( )
 { int x=1,y=2;
 fun(y,x);
  cout<<x<<y;
   }
```

A.2 1 4 3

B.1 2 1 2

C.1 2 3 4

D.2 1 1 2

2-7cout 是由 I/O 流库预定义的()

A.类

B.对象

C.包含文件

D.常量

2-8在下面几项中，运算符在 C++ 中不能被重载的是()

A.`&&`

B.`++`

C.`~`

D.`: :`

2-9已知：`print( )`函数是一个类的常成员函数，它无返回值，下列表示中，()是正确的

A.`const void print( );`

B.`void const print( );`

C.`void print(const);`

D.`void print( ) const;`

2-10 关于虚函数的描述中，（ ）是正确的

A.虚函数是一个 static 类型的成员函数

B.虚函数是一个非成员函数

C.基类中说明了虚函数后，派生类中与其对应的函数可不必说明为虚函数

D.派生类的虚函数与基类的虚函数具有不同的参数个数和类型

2-11关于纯虚函数和抽象类的描述中，（ ）是错误的

A.纯虚函数是一种特殊的虚函数，它没有具体的实现

B.一个基类中说明有纯虚函数，该基类的派生类一定不再是抽象类

C.抽象类是指具有纯虚函数的类

D.抽象类只能作为基类来使用，其纯虚函数的实现由派生类给出

2-12以下说法正确的是（ ）

A.派生类可以和基类有同名成员函数，但是不能有同名成员变量

B.派生类的成员函数中，可以调用基类的同名同参数表的成员函数

C.派生类和基类的同名成员函数必须参数表不同，否则就是重复定义

D.派生类和基类的同名成员变量存放在相同的存储空间

2-13下面描述中，表达错误的是（ ）

A.公用继承时基类中的public成员在派生类中仍是public的

B.公用继承时基类中的private成员在派生类中仍是private的

C.公用继承时基类中的protected成员在派生类中仍是protected的

D.私有继承时基类中的public成员在派生类中是private的

2-14下面对模板的声明，正确的是（ ）

A. `template<T>`

B. `template<class T1, T2>`

C. `template<class T1, class T2>`

D. `template<class T1; class T2>`

2-15如果某函数的返回值是一个对象，则该函数被调用时，返回的对象（）

A.是通过复制构造函数初始化的

B.是通过无参数的构造函数初始化的

C.用哪个构造函数初始化取决于函数中return语句是怎么写的

D.不需要初始化

2-20以下函数返回指针 a 所指数组中最小值所在的下标值：

```
fun(int *a,  int n)
{
    int i,  j=0,  p;
    p=j;
    for(i=j;  i<n;  i++)
    if(a[i]<a[p])  -------;
    return(p);
}
```

在下划线处应填入的是（）

A. `i=p`

B. `a[p]=a[i]`

C. `p=j`

D. `p=i`

2-16在下面类声明中，关于生成对象不正确的是（）

```
class point
{ public:
     int x;
     int y;
  point(int a,int b)
    {x=a; y=b;}
 };
```

A. `point p(10,2);`

B. `point *p=new point(1,2);`

C. `point *p=new point[2];`

D. `point *p[2]={new point(1,2), new point(3,4)};`

2-17如果默认参数的函数声明为 `void fun(int a, int b=1, char c='a', double d=3.2);`，则下面调用写法正确的是（）

A. `fun();`

B. `fun(2,3);`

C. `fun(2, ,'c', 3.14)`

D. `fun(int a=1);`

2-18下面关于静态成员的描述中，正确的是（　　）

A.静态数据成员是类的所有对象共享的数据

B.类的每个对象都有自己的静态数据成员

C.类的不同对象有不同的静态数据成员值

D.静态数据成员不能通过类的对象访问

2-19下面关于友元的描述中，错误的是（）

A.友元函数可以访问该类的私有数据成员

B.一个类的友元类中的成员函数都是这个类的友元函数

C.友元可以提高程序的运行效率

D.类与类之间的友元关系可以继承

函数题

6-1 2018final时间倒转

本题已给出时钟类及其成员函数实现，要求补充完整运算符--重载函数（前置和后置），使之能够实现时钟对象自减1秒。时间范围是从0:0:0——23:59:59。

时钟类定义如下：

```
class Clock {
    public:
        Clock(int NewH=0, int NewM=0, int NewS=0);
        void ShowTime();
        friend Clock operator--(Clock& op);
        friend Clock operator--(Clock& op,int);
    private:
        int Hour, Minute, Second;
};
```

裁判测试程序样例：

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

class Clock {
    public:
        Clock(int NewH=0, int NewM=0, int NewS=0);
        void ShowTime();
        friend Clock operator--(Clock& op);
        friend Clock operator--(Clock& op,int);
    private:
        int Hour, Minute, Second;
};

Clock::Clock(int NewH, int NewM, int NewS) {
    Hour=NewH;
    Minute=NewM;
    Second=NewS;
}
void Clock::ShowTime() {
    cout<<Hour<<":"<<Minute<<":"<<Second<<endl;
}

/*---------请在这里填写答案----------*/

int main() {
    int h, m, s;
    cin>>h>>m>>s;
    Clock a(h,m,s);
```

```
    (--a).ShowTime();
    (a--).ShowTime();
    a.ShowTime();

    return 0;
}
```

输入样例：

在这里给出一组输入。例如：

```
10 10 12
```

输出样例：

在这里给出相应的输出。例如：

```
10:10:11
10:10:11
10:10:10
```

6-2 2018Final链表(龙舟队)

凤湖中学有一支优秀的龙舟队。在任意时刻，龙舟队里至少有3人，其中至少有1名鼓手。龙舟队的成员一直在调整中。鼓手的离队有可能导致队中暂时没有鼓手，这时，就要自动选择最有资历的队员作为鼓手。

现在用一个单链表来存储龙舟队成员的信息，每个成员信息包括是否为鼓手， 编号(所有成员的编号不相同)和年级:

链表中按先鼓手，后一般队员的顺序存储队员信息。如果有多个鼓手，则按他们的编号排序，编号较小者在前；一般队员也按编号从小到大的规则排序。

在自动选择鼓手时，优先选择年级最高的，如果有多位队员同时具有最高的年级号，则选择其中编号最小的那位作鼓手。

输入时，首先输入若干名队员信息，每条信息包括3项:是否鼓手（0否，1是），编号，年级。当输入-1时结束输入初始信息。接着，进行队员增减操作，每次增减操作最多可能包括4项：第一项为操作类型(0为减少，1为增加，-1为终止操作），如果第1项为-1，则终止输入；如果第1项为0，则输入离开的成员的编号（假设所输入的编号肯定在当前龙舟队中）；如果第1项为1时，则输入新队员的3项信息：是否鼓手（0否，1是），编号（假设新加入成员的编号不会与现有成员相同），年级。

输出：在每次增减操作后，均输出当前龙舟队的前3位成员的信息（假设在建立链表后，任意时刻该链表中的结点数量均不会小于3）。

函数接口定义：

```
Player *create();//建立链表
Player* addPlayer(Player* head, Player* q);//加链表里加入q指向的结点
Player* removePlayer(Player *head, int num);//从链表里删除编号为num的结点，如果删除导致缺少鼓手则自动选择一位新鼓手
```

裁判测试程序样例：

```
#include <iostream>
using namespace std;
struct Player{
   bool drummer;
   int num;
   int grade;
   Player* next;
};
void print(Player *head){
    Player *p=head;
    cout<<p->num<<' ';
    p=p->next;
    cout<<p->num<<' ';
    p=p->next;
    cout<<p->num<<endl;
}

/* 请在这里填写答案 */

int main(){
    Player *head=NULL;
    int task, d, num, grade;
    head=create();
    while(1){
        cin>>task;
        if(task<0) break;
        if(task==1){
            Player*q = new Player;
            cin>>d>>q->num>>q->grade;
            if(d==1) q->drummer=true; else q->drummer=false;
            q->next = NULL;
            head=addPlayer(head, q);
        }
        if(task==0){
            cin>>num;
            head=removePlayer(head, num);
        }
        print(head);
    }
    return 0;
}
```

输入样例：

```
1 0 7
0 5 9
0 3 8
0 16 8
1 20 9
-1
0 0
0 20
-1
```

输出样例：

```
20 3 5
5 3 16
```

6-3 2018Final静态成员（黑名单）

怡山小学生物组是公认的熊孩子天堂，他们每天都在做各种尝试，如: 强迫蚕宝宝吃各种不同的菜叶，把小狗和小白鼠关进一个笼子里，重复输入流浪狗记录等等。忍无可忍的黄老师决定往成员信息里添加一项黑名单，限制黑名单中同学的单独活动，以保证生物组的日常管理秩序。

黑名单的增加来自两个时刻，一个是在增加新成员时，根据班主任的建议，直接将同学拉入黑名单，另一个是根据同学在组内的行为，由黄老师将其拉入。

黑名单的减小也有两个时刻，一个是黄老师将某位同学拉出，还有一个是黑名单已经满了，且需要拉入新的人员，此时，在黑名单中时间最长的成员，自动地被拉出黑名单。

输入：

输入时，先输入当前任务类型：

任务类型为1时，将加入新的学生记录；此时将输入学生的学号（假定新加入的学生肯定没有加入过），如果该学生应加入黑名单，则接着会多输入一个"999";

任务类型为2时，将某已有学生加入黑名单；此时将直接输入学生学号（假定不会将黑名单中已有的学生，再次加入黑名单）；

任务类型为3时，将某些学生移出黑名单；此时将直接输入学生学号（假定这个学号肯定在黑名单中）。

任务类型为0时，结束输入。

输出：

在执行类型2和3任务时，将输出当前的黑名单中的学生学号，用空格间隔，输出顺序为学生被加入黑名单的顺序，先加入者在前。如果黑名单为空，输出"NULL BLACKLIST!".

函数接口定义：

```
请补全类Group
```

裁判测试程序样例：

```
#include <iostream>
using namespace std;
const int N=3;
class Group{
    private:
        int num;
        static int blackList[N];
        static int size;
    public:
        Group();
        Group(int num, bool bSign);
        static void addToList(int num);
        static void removeFromList(int num);
        static void displayList();
};
void Group::displayList(){
```

```
    if(size==0) cout<<"NULL BLACKLIST!"<<endl;
        else{
            for(int i=0;i<size-1;i++) cout<<blackList[i]<<' ';
            cout<<blackList[size-1]<<endl;
        }
}
/* 请在这里填写答案 */

int main(){
    int i, j, k, num, task, count=0;
    Group g[100];
    cin>>task;
    while(task!=0){
        switch(task){
            case 1: cin>>num>>k;
                      if(k==999) {
                        g[count++]=Group(num, true);
                        cin>>task;
                      }else{
                            g[count++]=Group(num, false);
                            task = k;
                         }
                    break;
            case 2: cin>>num;
                      Group::addToList(num);
                      Group::displayList();
                      cin>>task;
                      break;
            case 3: cin>>num;
                      Group::removeFromList(num);
                      Group::displayList();
                      cin>>task;
                      break;
        }
    }
    return 0;
}
```

输入样例：

```
1 102
1 345 999
1 123
2 102
1 333
2 333
2 123
3 102
3 123
3 333
0
```

输出样例：

```
345 102
345 102 333
102 333 123
333 123
333
NULL BLACKLIST!
```

编程题

7-1 2018final世界杯也疯狂之球队积分计算

这是一个世界杯赛场，有球队类，其中，队名、记录每场比分的数组、累计积分 是其三个私有数据成员；有两个成员函数，分别是根据各场比分计算累计积分、以及计算净胜球总和。请添加适当的构造函数。 积分计算规则是：胜者得三分，负者不得分，打平双方各得一分。

输入格式:

输入仅一行，表示这个球队信息，分别是 1或2（1表示该队是主队，2表示该队是客队）、若干场比分（主队进球数:客队进球数，场数小于20）。每项信息间以空格隔开。

输出格式:

输出一行，分别是该队的累计积分和净胜球数，两项信息间以空格隔开。

输入样例:

在这里给出一组输入。例如：

```
1 3:5 5:0 2:2 3:7
```

输出样例:

在这里给出相应的输出。例如：

```
4 -1
```

7-2 2018Final简单字符串string处理（三人行）

凤湖中学在这学期流行一个名为“三人行”英文诗赛。比赛中，"老师"先说一个英语单词作为主题词，然后三位“路人"各说一句英语诗，看谁的诗更符合主题词。以主题词中的字母在诗中的出现次数作为符合度。在评价符合度时：不区分大小写；只关注英语字母(a-z)，忽略其它字母。如果某句诗所含英语字母数大于20个，则分别用该诗句的前20个英语字母和后20个英语字母来计算符合度，并取两者之较大值作为实际的符合度。

输入格式:

输入一共有4行，第1行是一个主题词，接下来的3行，各输入一句英语诗句。

输出格式:

输出一共有3行，是各句诗的符合度。

输入样例:

```
soon
Some tours are doomed to return, Some doors are destined to open then to close.
Days fly by. Soon I need to remove this page, and hang up a new year.
Summer isn't far behind, And another year has passed.
```

输出样例:

```
8
7
5
```

7-3 2018Final友元函数（体检表）

体育已经成为一门重要的中考科目，但身体存在某些疾病的同学可以免考。

为此，凤湖中学每年需要整理一下毕业班同学（学生数不超过300人）的体检表中的体检结论，筛选出需要免考的同学，建议其申请免考。

要求建立一个类Exam，每个Exam类对象存储一位同学的体检信息，包括私有数据成员：

string num;//学号 int disease[10];//疾病信息 int count;//疾病数

该类还包括一些必要的公有成员函数。

此外，该类还有一个友元函数void print(Exam*e, int count, int *d) 用于输出需要申请免考的学生信息，这里e是一个指向Exam类对象数组元素的指针，count是学生数量，d是一个指向值得关注病病数组元素的指针。

输入格式:

首先输入学生的体检信息，每位学生一行，以0结束输入。首先输入学生的学号（000-999，每位同学的学号各不相同）；接着输入疾病数T(T<=10)；然后是T个疾病信息，每种疾病用一个正整数编号(100-999)表示，每种疾病对应的编号不同。 在输入学生体检信息后，输入一行值得关注的疾病编号（相邻的编号以单个空格隔开），最后一个编号为0，标识结束输入病病编号。

输出格式:

输出由于患有值得关注疾病，需要申请免考的同学的学号，以及所患疾病编号（在存在多个值得关注的疾病时，仅输出编号最小的那个），每位同学一行，按照输入顺序输出。

输入样例:

```
103 0
109 1 101
002 3 105 101 107
825 4 108 775 109 104
0
105 101 0
```

输出样例:

```
109 101
002 101
```

7-4 2018Final多态(拗九节)

怡山小学生物组和艺术组的同学们今年拗九节为社区养老院的爷爷奶奶们送去了慰问演出。生物组的每位同学均带去了若干只猫或狗去表演节目，而艺术组的同学则唱歌或者跳舞。去养老院前，每位同学都事先预计了表演要持续的时间。不过，在献爱心时，他们都不愿意输给同学，一旦发现自己的节目预期持续时间短于已经表演过的同类节目的最长时间，则会临时延长表演时间，至同类节目的最长时间。

精彩的表演过程被全程录下来，每个节目一段，爷爷奶奶们会不断地点播重放这些节目，增添了不少欢乐。

现给出以下基类框架：

class Group

{ protected:

```
    int length; //时间长度
```

public:

```
  virtual void play()=0;//重放节目
```

};

以Group为基类（如果觉得有必要，可以向Group类中加入若干成员函数），构建出BioGroup和ArtGroup两个类，分别描述生物组和艺术组的表演。

要求主函数中有一个基类Group指针数组，数组元素不超过20个。

Group *g[20];

主函数根据输入的信息，相应建立BioGroup或ArtGroup类对象，对于BioGroup类对象要能给出参与表演的动物（cat/dog）及其数量，和表演的时间（以秒为单位）； 对于ArtGroup类对象要能给出表演的类型（dance/sing），和表演的时间。

输入格式:

首先输入表演信息

每一行为一位同学的表演信息：

其中第1项为组别, B为生物组，A为艺术组，如果输入为E，则表示结束输入表演信息。

对于生物组同学来说，接下来依次输入参与表演的动物(C或D, C指cat，D指dog), 动物的数量（不小于1的正整数），预期表演的时间长度T（正整数）。

对于艺术组同学来说，接下来输入表演的形式(S或D， S指sing, D指dance)，预期表演的时间长度T（正整数）

第一行的表演节目编号为1,第二行为2，依此类推。

接着输入点播要求：

点播要求为一行以空格隔开的若干个数字，最后一个数字是0。除0以外的每个数字均为节目编号（假设所给的节目编号对应的表演肯定存在）。

输出格式:

点播节目的信息。每行一个。按点播次序给出。

输入样例:

```
B C 2 5
B C 1 3
B D 2 2
A S 2
A S 8
A S 4
A D 1
B C 2 7
E
1 6 2 7 4 0
```

输出样例:

```
2 cats, 5 seconds
sing, 8 minutes
1 cat, 5 seconds
dance, 1 minute
sing, 2 minutes
```

说明：

节目1：B C 2 5意味着2只cats表演5秒

节目2: B C 1 3意味着1只cat表演3秒，但是以前cat类节目的最长时间为5秒，大于3秒，所以此时实际节目为1只cat表演5秒

节目3:B D 2 2意味着2只dogs表演2秒

节目4：A S 2意味着sing2分钟

节目5:A S 8意味着sing8分钟

节目6:A S 4意味着sing4分钟，但是此前sing类节目的最长时间为8分钟，大于4分钟，所以此时实际节目为sing8分钟。

节目7：A D 1意味着dance1分钟

节目8：B C 2 7意味着2只cats表演7秒

E意味着结束输入节目信息

1 6 2 7 4 0

意味着依次输出1,6,2,7,4节目的具体信息，每行一个节目。

7-5 2018final复数求模的类模板

有一个复数的类模板，有两个私有数据成员，分别是 实部和虚部。有一个成员函数是求该复数的模。请添加适当的构造函数。

输入格式:

输入仅一行，分别是三个数，以空格间隔。 第一个数是1或2或3（1表示int型，2表示float型，3表示double型），第二个数是该复数的实部，第三个数是该复数的虚部。

输出格式:

输出仅一行，输出该复数的模（in型的复数，输出int型的模; float型的复数，输出float型的模；double型的复数，输出double型的模）。

输入样例:

在这里给出一组输入。例如：

```
2 2.5 -3.1
```

输出样例:

在这里给出相应的输出。例如：

```
3.98246
```

7-6 我的支付宝和余额宝

支付宝`AliPay`和余额宝`AliFund`是一对好兄弟，他们来自同一个父类Account。

已知类Account是支付宝`AliPay`和余额宝`AliFund`的虚基类，包括两个protected成员数据：

```
long ID;//账号
string name;//用户名
支付宝AliPay是类Account的保护派生类，包括两个新增protected成员数据：
double amount;//支付宝账户中的金额
int out-limit;//支付宝转账上限，资金转出不得超出账户中金额也不能超出上限
//当转账要求超出上述限制时，自动转出最大允许金额
```

余额宝`AliFund`是类Account的保护派生类，包括三个新增protected成员数据：

```
double amount;//余额宝账户中的金额
double rate;//余额宝账户中资金年利率
int in-limit;//余额宝转账下限，资金转入余额宝时不得少于该下限，上不封顶
//当转入资金不足下限金额时，自动忽略该资金转入操作，资金退回
```

为了实现支付宝账户和余额宝账户间的资金收集和转账等功能，现以`AliPay`和`AliFund`为基类，创建一个淘宝账户类My_Ali，包括以下新增数据成员：

```
bool auto_collect_flag;//资金自动收集标志
int threshold;//资金自动收集阈值
/**/若资金自动收集标志为true，当用户支付宝账户金额超过资金自动收集阈值threshold时，超过且符合支付宝账户资金转出限制和余额宝资金转入下限的部分资金将自动转入余额宝中*/*/
```

增加合适的成员函数，可以实现支付宝账户和余额宝账户之间的资金转账和支付宝账户向余额宝账户的资金自动收集功能。

生成以上各类并编写主函数，根据输入的初始账户信息，建立用户对象，根据用户的账户操作信息，输出用户最后的资金信息。

测试输入包含一个测试用例:

第一行为初始账户信息，第一个数据是账号，第二个字符串是用户名，第三个数据是支付宝初始金额，第四个数据是支付宝转账上限，第五个数据是余额宝初始金额，第六个数据是余额宝资金年利率，第七个数据是余额宝转账下限，第八个数据是资金自动收集标志，字符Y代表设置资金自动收集，字符N代表不设置资金自动收集。若设置资金自动收集，第九个数据是资金自动收集阈值，否则，第九个数据不存在。 随后每一行代表用户资金账户的一次操作，第一个数字代表当前操作的账户类型，1代表支付宝账户，2代表余额宝账户，第二个数字代表具体操作，1代表账户资金增加，2代表账户金额减少，3代表支付宝和余额宝之间的资金转移，第三个数字代表操作金额。 最后输入0 0 0时结束。 注意：当设置资金自动收集为真时，支付宝账户资金每次增加时会自动触发资金自动收集。

输入样例:

```
100001 Zhangsan 1000 30000 50000 0.047 5000 Y 3000
1 1 1500
1 2 500
1 3 500
1 1 7000
2 3 500
0 0 0
```

输出样例:

```
Account for Zhangsan
Total: 59000
AliPay: 3500
AliFund: 55500 with rate 4.7%
```