**七、程序改错**

1. 在下面的定义中，NODE是链表结点的结构，appendToList则是一函数，其功能是：在list所指向的链表的末尾添加一个新的值为x的结点，并返回表头指针。函数中有两处错误，指出错误所在行的行号并提出改正意见。

struct NODE{

int data;

NODE \*next;

};

NODE\* appendToList(NODE \*list, int x){ //1行

NODE \*p=new int; //2行

p->data=x; //3行

p->next=NULL; //4行

if(list==NULL) return p; //5行

NODE \*p1=list; //6行

while(p1->next!=NULL) p1=p1->next; //7行，链表非空先找到表尾

p1=p; //8行，让原表尾指针指向新添加的结点

return list;

}

错误行的行号为\_\_\_\_\_\_和\_\_\_\_\_\_\_\_。

分别改正为\_\_\_\_\_\_\_\_\_\_\_\_\_\_和\_\_\_\_\_\_\_\_\_\_\_\_\_\_。

2. 在下面的定义中，NODE是链表结点的结构，addToList则是一函数，其功能是：将一个值为x的新结点添加到以plist为表头指针的链表的首部(即第一个结点的前面)并返回表头指针。函数中有两处错误，指出错误所在行的行号并提出改正意见。

struct NODE{

int data;

NODE \*next;

};

NODE\* adndToList(NODE \* plist, int x){ //1行

NODE \*p; //2行

\*p=new NODE; //3行

p->data=x; //4行

p->next=NULL; //5行

plist=p; //6行

return p; //7行

}

错误行的行号为\_\_\_\_\_\_和\_\_\_\_\_\_\_\_。

分别改正为\_\_\_\_\_\_\_\_\_\_\_\_\_\_和\_\_\_\_\_\_\_\_\_\_\_\_\_\_。

3. 假定要求下面程序的输出结果为“11/15”,其主函数中存在着三行语句错误，请指出错误语句行的行号并改正错误行。

#include<iostream.h>

class Franction { //定义分数类

int nume; //定义分子

int deno; //定义分母

public:

//把\*this化简为最简分数，具体定义在另外文件中实现

void FranSimp();

//返回两个分数\*this和x之和，具体定义在另外文件中实现

Franction FranAdd(const Franction& x);

//置分数的分子和分母分别0和1

void InitFranction() {nume=0; deno=1;}

//置分数的分子和分母分别n和d

void InitFranction(int n, int d) {nume=n; deno=d;}

//输出一个分数

void FranOutput() {cout<<nume<<'/'<<deno<<endl;}

};

void main() //1行

{ //2行

Franction a,b,c; //3行

a.InitFranction(6,15); //4行

b.InitFranction(1); //5行

c.InitFranction(); //6行

c=FranAdd(a,b); //7行

cout<<c.nume<<’/’<<c.deno<<endl; //8行

} //9行

错误行的行号为\_\_\_\_\_\_、\_\_\_\_\_\_\_\_和\_\_\_\_\_\_\_\_。

分别改正为\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_、\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_和\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_。

4. 假定要求下面程序的输出结果为“23/20”,其主函数中存在着三条语句错误，请指出错误语句行的行号并改正。

#include<iostream.h>

class Franction { //定义分数类

int nume; //定义分子

int deno; //定义分母

public:

//把\*this化简为最简分数，具体定义在另外文件中实现

void FranSimp();

//返回两个分数\*this和x之和，具体定义在另外文件中实现

Franction FranAdd(const Franction& x);

//置分数的分子和分母分别0和1

void InitFranction() {nume=0; deno=1;}

//置分数的分子和分母分别n和d

void InitFranction(int n, int d) {nume=n; deno=d;}

//输出一个分数

void FranOutput() {cout<<nume<<'/'<<deno<<endl;}

};

void main() //1行

{ //2行

Franction \*a=new Franction; //3行

Franction \*b=new Franction; //4行

a->InitFranction(6,15); //5行

b.InitFranction(3,4); //6行

Franction c; //7行

c.InitFranction(); //8行

c=a.FranAdd(b); //9行

cout<<c.FranOutput()<<endl; //10行

} //11行

错误行的行号为\_\_\_\_\_\_、\_\_\_\_\_\_\_\_和\_\_\_\_\_\_\_\_。

分别改正为\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_、\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_和\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_。

5. 下面是一个类的定义，存在着3处语法错误，请指出错误行的行号并改正。

class CE { //1行

private: //2行

int a,b; //3行

int getmin() {return (a<b? a:b);} //4行

public //5行

int c; //6行

void SetValue(int x1,int x2, int x3) { //7行

a=x1; b=x2; c=x3; //8行

}; //9行

int GetMin(); //10行

}; //11行

int GetMin() { //12行

int d=getmin(); //13行

return (d<c? d:c); //14行

} //16行

错误行的行号为\_\_\_\_\_\_、\_\_\_\_\_\_\_\_和\_\_\_\_\_\_\_\_。

分别改正为\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_、\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_和\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_。

6. 下面程序段第4-10行中存在着三条语句的语法错误，请指出错误语句的行号并改正。

class A { //1行

int a,b; //2行

const int c; //3行

public //4行

A():c(0);a(0);b(0) {} //5行

A(int aa, int bb) c(aa+bb); {a=aa; b=bb;} //6行

}; //7行

A a,b(1,2); //8行

A \*x=&a, &y=b; //9行

A \*z=new A, w[10]; //10行

错误行的行号为\_\_\_\_\_\_、\_\_\_\_\_\_\_\_和\_\_\_\_\_\_\_\_。

分别改正为\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_、\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

和\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_。

Public: A():c(0),a(0),b(0) {}

A(int aa, int bb): c(aa+bb) {a=aa; b=bb;}

7. 下面程序段第4-9行中存在着三条语句错误，请指出错误语句的行号并说明原因。

class A { //1行

int a,b; //2行

const int c; //3行

public: //4行

A() {a=b=c=0;} //5行

A(int aa, int bb):c(aa+bb) {a=aa; b=bb;} //6行

}; //7行

A a,b(1,2,3); //8行

A x(2,3), y(4); //9行

错误行的行号为\_\_\_\_\_\_、\_\_\_\_\_\_\_\_和\_\_\_\_\_\_\_\_。

错误原因分别为\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_、\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_和\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_。

8. 下面程序段第10-17行中存在着三条语句错误，请指出错误语句的行号并说明原因。

class A { //1行

int a; //2行

public: //3行

A(int aa=0):a(aa){} //4行

}; //5行

class B { //6行

int a,b; //7行

const int c; //8行

A d; //9行

public: //10行

B():c(0) {a=b=0;} //11行

B(int aa, int bb):d(aa+bb) { //12行

a=aa; b=bb; c=aa-bb; //13行

} //14行

} //15行

B a,b(1,2); //16行

B x=a,y(b),z(1,2,3),; //17行

错误行的行号为\_\_\_\_\_\_、\_\_\_\_\_\_\_\_和\_\_\_\_\_\_\_\_。

错误原因分别为\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_、\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

和\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_。

9. 假定要求下面程序输出结果为“d=800,f=60”，在第4-23行中存在着三条语句错误，请指出错误语句的行号并改正。

#include<iostream.h>

class A { //1行

int a[10]; int n; //2行

public: //3行

A(int aa[], int nn): n(nn) { //4行

for(int i=0; i<n; i++) aa[i]=a[i]; //5行

} //6行

int Get(int i) {return a[i];} //7行

int SumA(int n); //8行

}; //9行

int A::SumA(int n) { //10行

int s=0; //11行

for(int j=0; j<n; j++) s+=a[j]; //12行

return s; //13行

} //14行

void main() { //15行

int a[]={2,5,8,10,15,20}; //16行

A x(a,6); //17行

int d=1; //18行

for(int i=0; i<4; i++) d\*=x.a[i]; //19行

int f=SumA(6); //20行

cout<<"d="<<d<<’,’; //21行

cout<<"f="<<f<<endl; //22行

} //23行

错误行的行号为\_\_\_\_\_\_、\_\_\_\_\_\_\_\_和\_\_\_\_\_\_\_\_。5 19 20

分别改正为\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_、\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_和\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_。

10. 下面是分数类fract的定义及测试主程序，在类定义及其友元函数定义中有两处错误，更正错误后程序应显示41/28，请指出错误所在行的行号并给出改正意见。

class fract{

int den; //分子

int num; //分母

public:

fract(int d=0,int n=1):den(d),num(n){} //1行

friend fract &operator+=(fract,fract&); //2行

void show(){ cout<<den<<'/'<<num;} //3行

}; //4行

friend fract &operator+=(fract f1,fract f2) //5行

{ //7行

f1.den=f1.den\*f2.num+f1.num\*f2.den; //8行

f1.num\*=f2.num; //9行

return f1; //10行

}

void main(){

fract fr(3,4);

fr+=fract(5,7);

fr.show();

}

错误行的行号为\_\_\_\_\_\_和\_\_\_\_\_\_\_\_。

分别改正为\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

和\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_。