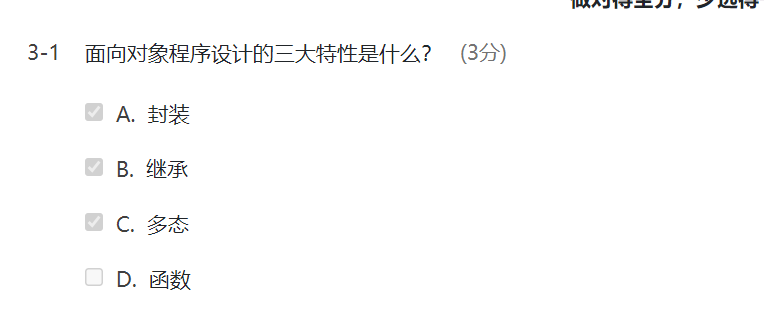
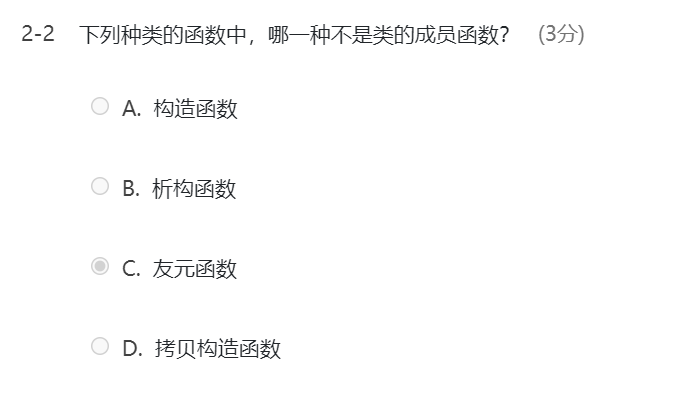
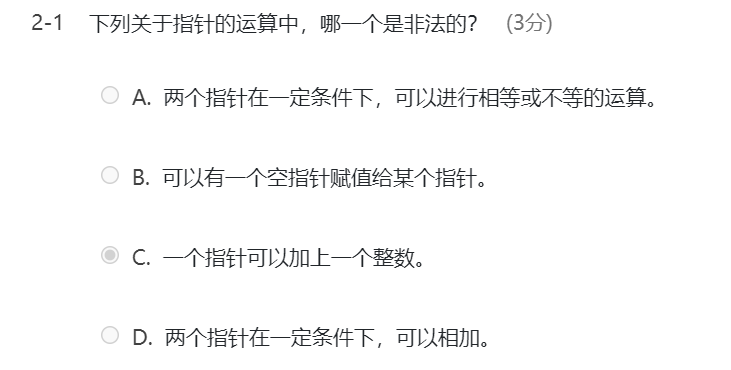
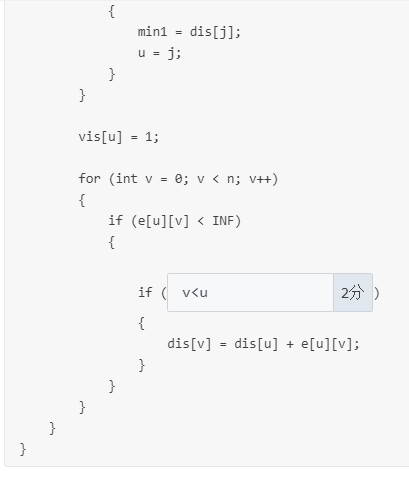
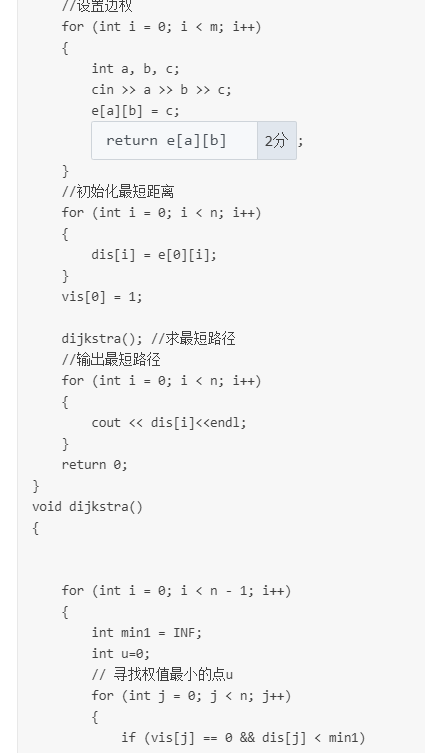
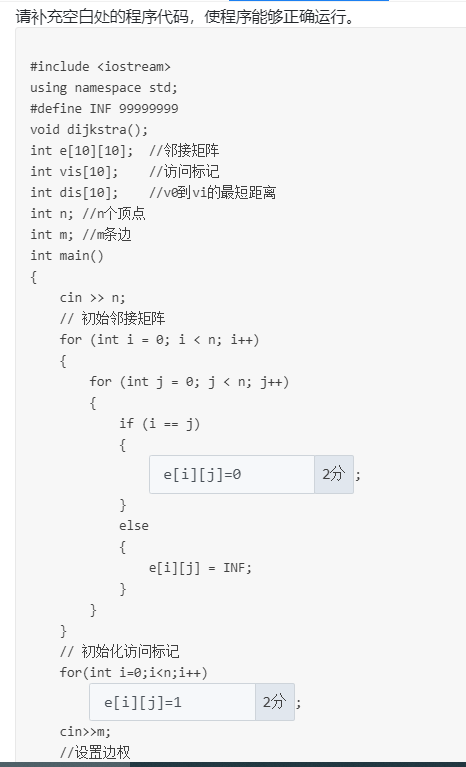
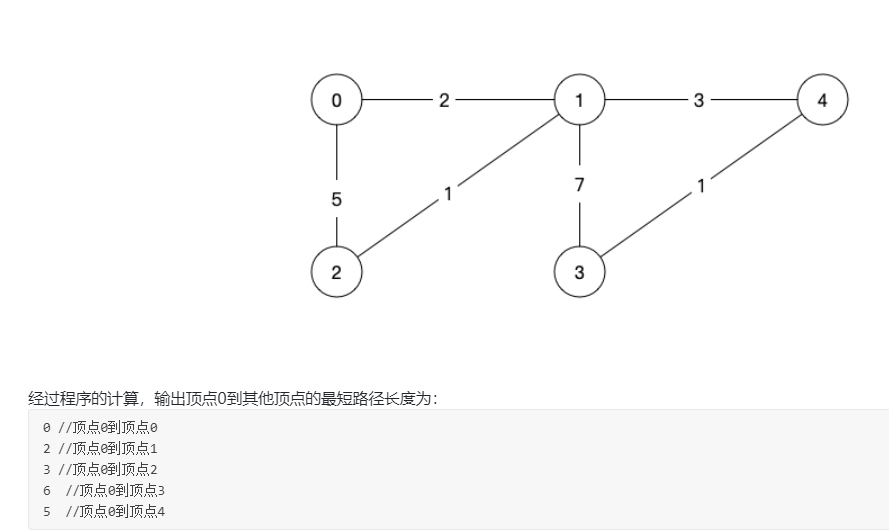
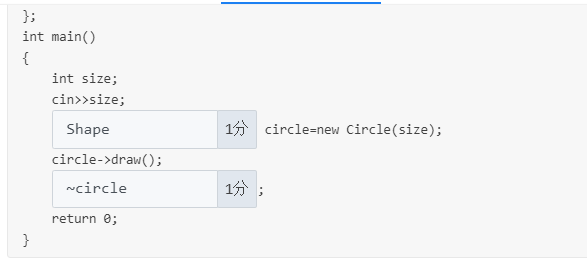
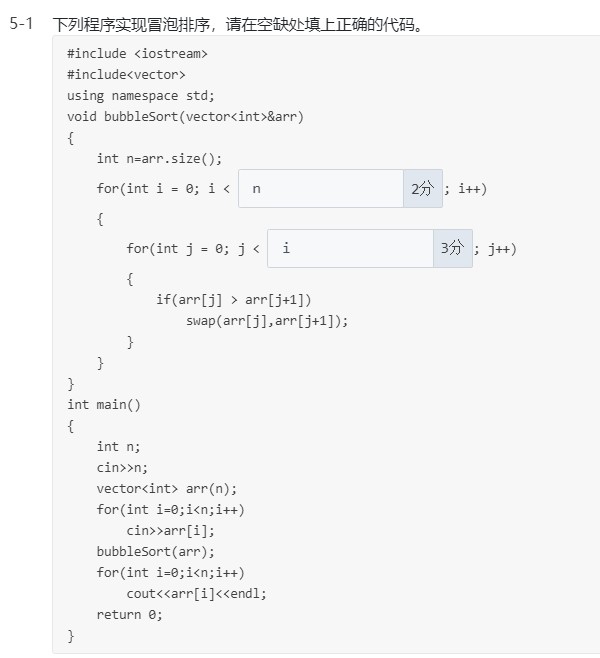
## 第一次测试：







#include <iostream>

using namespace std;

template<typename ElemType>

class ArrayList

{

private:

ElemType \*elems; //数据元素

int sizeOfElems; //数据个数

int sizeOfAllocatedMemory; //分配内存大小

static const int INC=10;//内存增量

void reallocate(); //重新分配内存

public:

ArrayList(int size=0); //构造函数

ArrayList(const ArrayList<ElemType>& list); //拷贝构造函数

~ArrayList(); //析构函数

int size() const; //返回数据个数

void add(int index, ElemType e); //插入数据

void add(ElemType e); //插入数据

void remove(int index); //移除数据

void remove(); //移除数据

void clear(); //清除全部数据

ElemType& operator[](int index); //返回数据

ArrayList<ElemType>& operator=(const ArrayList<ElemType>& list); //=运算符重载

friend



2分

(ostream& os, const ArrayList<ElemType>& list) //<<运算符重载

{

for(int i=0;i<list.size();i++)

cout<<list.elems[i]<<"\t";

return os;

}

};

template<typename ElemType>

ArrayList<ElemType>::ArrayList(int size)

{

elems=new ElemType[size];

sizeOfElems=0;

sizeOfAllocatedMemory=size;

}

template<typename ElemType>

ArrayList<ElemType>::ArrayList( const ArrayList<ElemType>& list )

{

elems=new ElemType[list.sizeOfAllocatedMemory];

sizeOfAllocatedMemory=list.sizeOfAllocatedMemory;

sizeOfElems=list.sizeOfElems;

for(int i=0;i<list.sizeOfElems;i++)



1分

;

}

template<typename ElemType>

ArrayList<ElemType>::~ArrayList()

{

delete[] elems;

}

template<typename ElemType>

int ArrayList<ElemType>::size() const

{

return sizeOfElems;

}

template<typename ElemType>

void ArrayList<ElemType>::reallocate()

{

int newSizeOfAllocatedMemory=sizeOfAllocatedMemory+INC;

ElemType \*newElems=new ElemType[newSizeOfAllocatedMemory];

for(int i=0;i<sizeOfElems;i++)

{

newElems[i]=elems[i];

}

delete []elems;

elems=newElems;

sizeOfAllocatedMemory=newSizeOfAllocatedMemory;

}

template<typename ElemType>

void ArrayList<ElemType>::add( int index, ElemType e )

{

if (index>sizeOfElems) {

throw string("index out of bounds!");

}

int newSizeOfArray=sizeOfElems+1;

if(sizeOfAllocatedMemory<newSizeOfArray)

{

reallocate();

}

for(int i=sizeOfElems-1;i>=index;i--)

{

elems[i+1]=elems[i];

}

elems[index]=e;

sizeOfElems=newSizeOfArray;

}

template<typename ElemType>

void ArrayList<ElemType>::add( ElemType e )

{

add(this->sizeOfElems,e);

}

template<typename ElemType>

void ArrayList<ElemType>::remove( int index )

{

int newSizeOfElems=sizeOfElems-1;

if(index>newSizeOfElems || index<0)

{

throw string("index out of bounds!");

}

if(sizeOfElems==0)

{

throw string("There is no element to remove!");

}

for(int i=index;i<newSizeOfElems;i++)

{

elems[i]=elems[i+1];

}

sizeOfElems=newSizeOfElems;

}

template<typename ElemType>

void ArrayList<ElemType>::remove()

{

remove(0);

}

template<typename ElemType>

void ArrayList<ElemType>::clear()

{



1分

;

}

template<typename ElemType>

ElemType& ArrayList<ElemType>::operator[]( int index )

{

if(index>sizeOfElems-1 || index<0)

{

throw string("Index out of bounds");

}

return



2分

;

}

template<typename ElemType>

ArrayList<ElemType>& ArrayList<ElemType>::operator=( const ArrayList<ElemType>& list )

{

delete []elems;

elems=new ElemType[list.sizeOfAllocatedMemory];

sizeOfAllocatedMemory=list.sizeOfAllocatedMemory;



2分

;

for(int i=0;i<list.sizeOfElems;i++)

elems[i]=list.elems[i];

return \*this;

}

int main(int argc, const char \* argv[]) {

try

{

ArrayList<int> l1;

l1.add(0,10);

l1.add(0,20);

l1.add(0,30);

l1.add(40);

cout<<"l1:"<<l1<<endl;

ArrayList<int> l2(l1);

l2.remove(1);

cout<<"l2:"<<l2<<endl;

l2.remove();

cout<<"l2:"<<l2<<endl;

ArrayList<int> l3=l2;

cout<<"l3:"<<l3<<endl;

cout<<"The size of l3 is "<<l3.size()<<endl;

l3[0]=1000;

cout<<"l3:"<<l3<<endl;

l3.clear();

cout<<"l3:"<<l3<<endl;

}

catch (string e)

{

cout<<"Exception："<<e<<endl;

}

return 0;

}

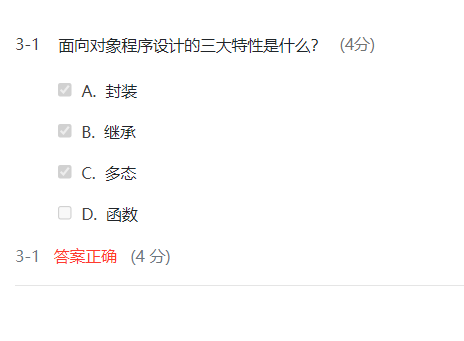


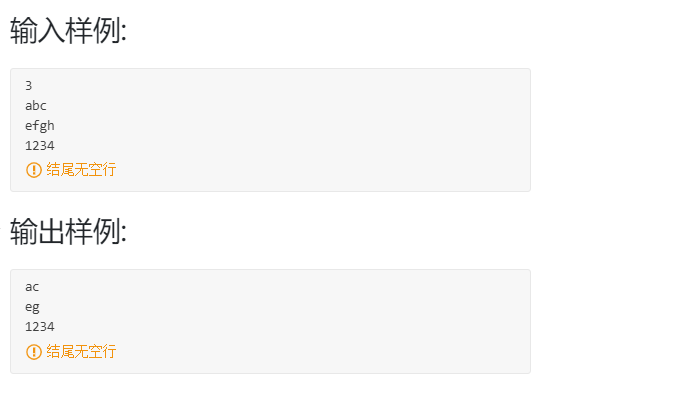




## 第二次测试







#include<bits/stdc++.h>

using namespace std;

int main()

{

int n;

cin>>n;

while(n--)

{

char aa[1000000],bb[1000000];

cin>>aa;

int flg=0;

int dp=0;

int i;

int length=strlen(aa);

for(i=0;i<length;i++)

{

if(('a'<=aa[i]&&aa[i]<='z')||('A'<=aa[i]&&aa[i]<='Z'))

{flg=1;}

if(i%2==0)

{

bb[dp]=aa[i];

dp++;

}

}

if(flg==1)

{

cout<<bb<<endl;

}

else

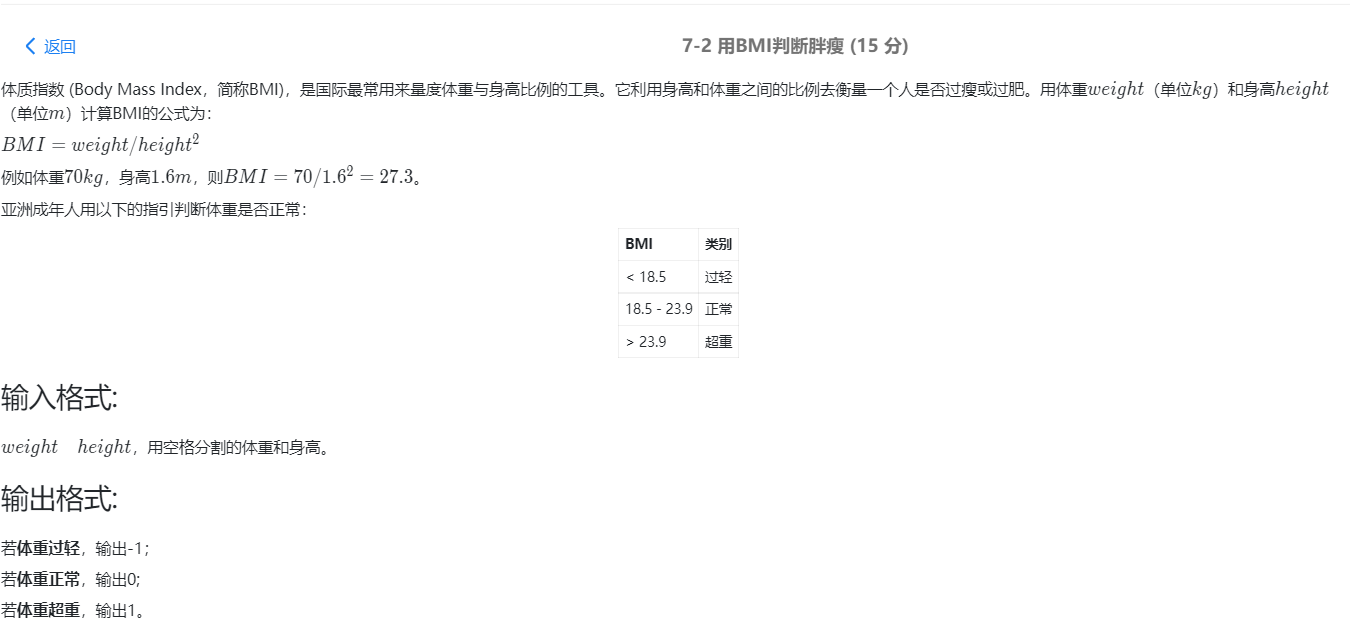
{

cout<<aa<<endl;

}

}

}





#include<bits/stdc++.h>

using namespace std;

int main()

{

float height;

float weight;

float BMI;

cin>>weight>>height;

BMI=weight/(height\*height);

if(BMI<18.5)

{cout<<"-1";}

else if(BMI>23.9)

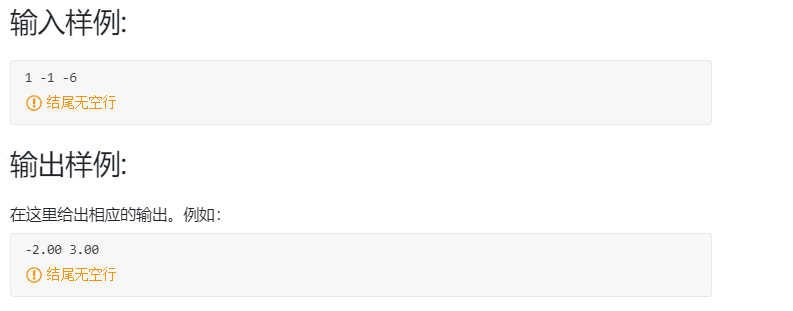
{cout<<"1";}

else cout<<"0";

return 0;

}





#include<bits/stdc++.h>

using namespace std;

int main()

{

float a,b,c;

float x1,x2;

float delta;

cin>>a>>b>>c;

delta=(b\*b-4\*a\*c);

if(delta>0)

{

x1=(-b-sqrt(delta))/(2\*a);

x2=(-b+sqrt(delta))/(2\*a);

if (x1<=x2)

printf("%.2f %.2f",x1,x2);

}

else

return 0;

}



#include<bits/stdc++.h>

using namespace std;

int main()

{

int a,b,c;//weight of each good

int m1,n1,w1,m2,n2,w2,m3,n3,w3;

cin>>a>>b>>c;

m1=a+b;

n1=c;

if(m1>=n1){

w1=m1-n1;

}

else {

w1=n1-m1;

}

m2=a+c;

n2=b;

if(m2>=n2){

w2=m2-n2;

}

else

{

w2=n2-m2;}

m3=b+c;

n3=a;

if(m3>=n3){

w3=m3-n3;

}

else {

w3=n3-m3;

}

if((w1<=w2)&&(w1<=w3))

{

cout<<w1;}

else if((w2<=w1)&&(w2<=w3))

{

cout<<w2;}

else if((w3<=w1)&&(w3<=w2))

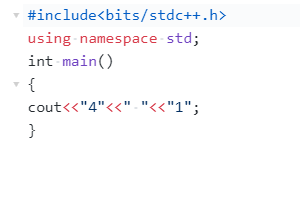
{

cout<<w3;}

}







**强行cout可能有分**