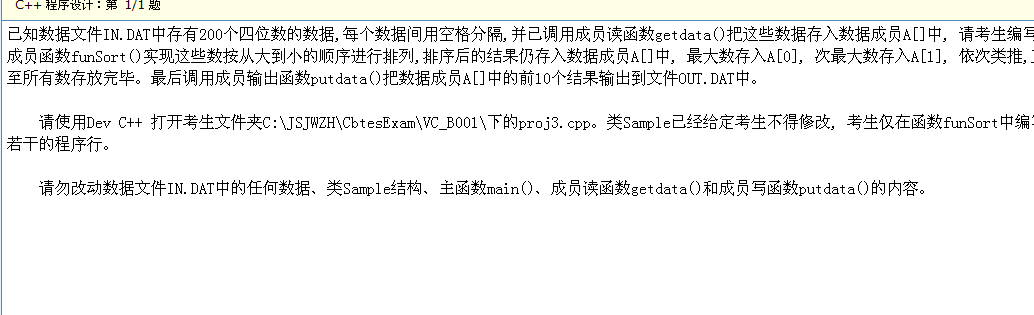
**设计题**



从大到小

#include <iostream>

#include <fstream>

using namespace std;

#define Max 200

class Sample

{

int A[Max] ;

public:

Sample() {}

int getNumber(Sample s, int k) { return s.A[k] ; }

void getdata(Sample &s) ; // 读数据函数

void funSort(Sample &s) ; // 要求考生编写的函数

void putdata(Sample &s) ; // 写结果函数

};

void Sample::funSort(Sample &s)

{

int i,j;

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

{

for(j=0;j<Max-i;j++)

{

if(A[j]<A[j+1])

{

int tmp=A[j];

A[j]=A[j+1];

A[j+1]=tmp;

}

}

}

}

int main()

{

int i;

Sample s ;

s.getdata(s) ;

cout << "读取的数据如下:" << endl ;

for(int i = 0 ; i < Max ; i++) {

cout << s.getNumber(s, i) << " " ;

if((i + 1) % 10 == 0) cout << endl ;

}

cout << endl ;

s.funSort(s) ;

s.putdata(s) ;

cout << "输出结果:" << endl ;

for(i = 0 ; i < 10 ; i++)

cout << s.getNumber(s, i) << " " ;

cout << endl ;

}

//\*\*\*\*\*\*\*\*\*\*\*以下请考生不要修改\*\*\*\*\*\*\*\*\*\*\*\*

void Sample::getdata(Sample &s) // 从数据文件中读取数据

{

fstream fp ;

fp.open(".\\in.dat", ios::in) ;

for(int i = 0 ; i < Max ; i++) fp >> s.A[i] ;

fp.close() ;

}

void Sample::putdata(Sample &s) // 写结果文件

{

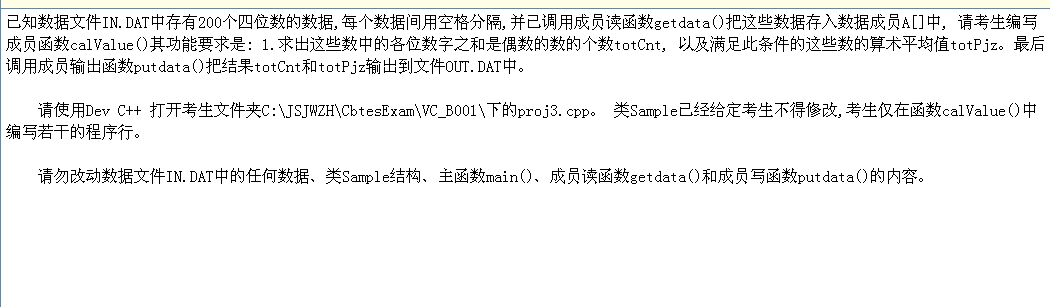
fstream fp ;

fp.open(".\\out.dat", ios::out) ;

for(int i = 0 ; i < 10 ; i++) fp << s.A[i] << endl ;

fp.close() ;

}



各位数字之和是偶数的个数，算数平均值

#include <iostream>

#include <fstream>

using namespace std;

#define Max 160

class Sample

{

int A[Max] ;

int totCnt ;

double totPjz ;

public:

Sample() { totCnt = 0 ; totPjz = 0.0 ; }

void getNumber(Sample s, int &num, double &pjz) { num = s.totCnt ; pjz = s.totPjz ; }

int getNumber(Sample s, int k) { return s.A[k] ; }

void getdata(Sample &s) ; // 读数据函数

void calValue(Sample &s) ; // 要求考生编写的函数

void putdata(Sample &s) ; // 写结果函数

};

void Sample::calValue(Sample &s)

{

int i;

int n,m,p,q;

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

{

if (A[i]>0)

{

n=A[i]%10;

m=n%100/10;

p=n/100%10;

q=n/1000;

if ((n+m+p+q)%2==0)

{

s.totCnt++;

s.totPjz+=A[i];

}

}

}

if(s.totCnt==0)

{

s.totPjz=0;

}

else

{

s.totPjz/=s.totCnt;

}

}

int main()

{

int num ; double pjz ;

Sample s ;

s.getdata(s) ;

cout << "读取的数据如下:" << endl ;

for(int i = 0 ; i < Max ; i++) {

cout << s.getNumber(s, i) << " " ;

if((i + 1) % 10 == 0) cout << endl ;

}

cout << endl ;

s.calValue(s) ;

s.putdata(s) ;

cout << "输出结果:" << endl ;

s.getNumber(s, num, pjz) ;

cout << num << endl << pjz << endl ;

}

//\*\*\*\*\*\*\*\*\*\*\*以下请考生不要修改\*\*\*\*\*\*\*\*\*\*\*\*

void Sample::getdata(Sample &s) // 从数据文件中读取数据

{

fstream fp ;

fp.open(".\\in.dat", ios::in) ;

for(int i = 0 ; i < Max ; i++) fp >> s.A[i] ;

fp.close() ;

}

void Sample::putdata(Sample &s) // 写结果文件

{

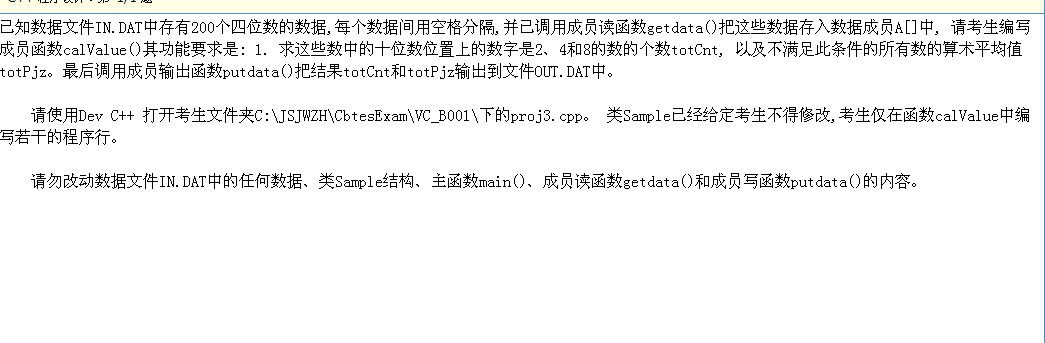
fstream fp ;

fp.open(".\\out.dat", ios::out) ;

fp << s.totCnt << endl << s.totPjz << endl ;

fp.close() ;

}



在十位数位置是2.4.8的数的个数，算术平均值

#include <iostream>

#include <fstream>

using namespace std;

#define Max 160

class Sample

{

int A[Max] ;

int totCnt ;

double totPjz ;

public:

Sample() { totCnt = 0 ; totPjz = 0.0 ; }

void getNumber(Sample s, int &num, double &pjz) { num = s.totCnt ; pjz = s.totPjz ; }

int getNumber(Sample s, int k) { return s.A[k] ; }

void getdata(Sample &s) ; // 读数据函数

void calValue(Sample &s) ; // 要求考生编写的函数

void putdata(Sample &s) ; // 写结果函数

};

void Sample::calValue(Sample &s)

{

int i;

int n;

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

{

n = (s.A[i] / 10) % 10;

if (n == 2 || n == 4 || n == 8)

{

s.totCnt++;

}

else

{

s.totPjz += s.A[i];

}

}

s.totPjz /= Max - s.totCnt;

}

int main()

{

int num ; double pjz ;

Sample s ;

s.getdata(s) ;

cout << "读取的数据如下:" << endl ;

for(int i = 0 ; i < Max ; i++) {

cout << s.getNumber(s, i) << " " ;

if((i + 1) % 10 == 0) cout << endl ;

}

cout << endl ;

s.calValue(s) ;

s.putdata(s) ;

cout << "输出结果:" << endl ;

s.getNumber(s, num, pjz) ;

cout << num << endl << pjz << endl ;

}

//\*\*\*\*\*\*\*\*\*\*\*以下请考生不要修改\*\*\*\*\*\*\*\*\*\*\*\*

void Sample::getdata(Sample &s) // 从数据文件中读取数据

{

fstream fp ;

fp.open(".\\in.dat", ios::in) ;

for(int i = 0 ; i < Max ; i++) fp >> s.A[i] ;

fp.close() ;

}

void Sample::putdata(Sample &s) // 写结果文件

{

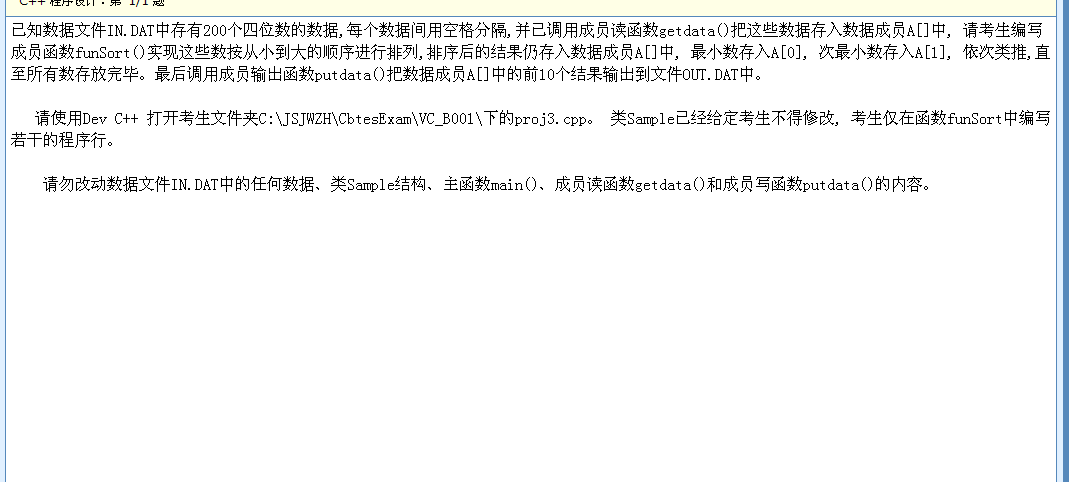
fstream fp ;

fp.open(".\\out.dat", ios::out) ;

fp << s.totCnt << endl << s.totPjz << endl ;

fp.close() ;

}



从小到大

#include <iostream>

#include <fstream>

using namespace std;

#define Max 200

class Sample

{

int A[Max] ;

public:

Sample() {}

int getNumber(Sample s, int k) { return s.A[k] ; }

void getdata(Sample &s) ; // 读数据函数

void funSort(Sample &s) ; // 要求考生编写的函数

void putdata(Sample &s) ; // 写结果函数

};

void Sample::funSort(Sample &s)

{

int i,j;

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

{

for(j=0;j<Max-i;j++)

{

if(A[j]>A[j+1])

{

int tmp=A[j];

A[j]=A[j+1];

A[j+1]=tmp;

}

}

}

}

int main()

{

int i;

Sample s ;

s.getdata(s) ;

cout << "读取的数据如下:" << endl ;

for(int i = 0 ; i < Max ; i++) {

cout << s.getNumber(s, i) << " " ;

if((i + 1) % 10 == 0) cout << endl ;

}

cout << endl ;

s.funSort(s) ;

s.putdata(s) ;

cout << "输出结果:" << endl ;

for(i = 0 ; i < 10 ; i++)

cout << s.getNumber(s, i) << " " ;

cout << endl ;

}

//\*\*\*\*\*\*\*\*\*\*\*以下请考生不要修改\*\*\*\*\*\*\*\*\*\*\*\*

void Sample::getdata(Sample &s) // 从数据文件中读取数据

{

fstream fp ;

fp.open(".\\in.dat", ios::in) ;

for(int i = 0 ; i < Max ; i++) fp >> s.A[i] ;

fp.close() ;

}

void Sample::putdata(Sample &s) // 写结果文件

{

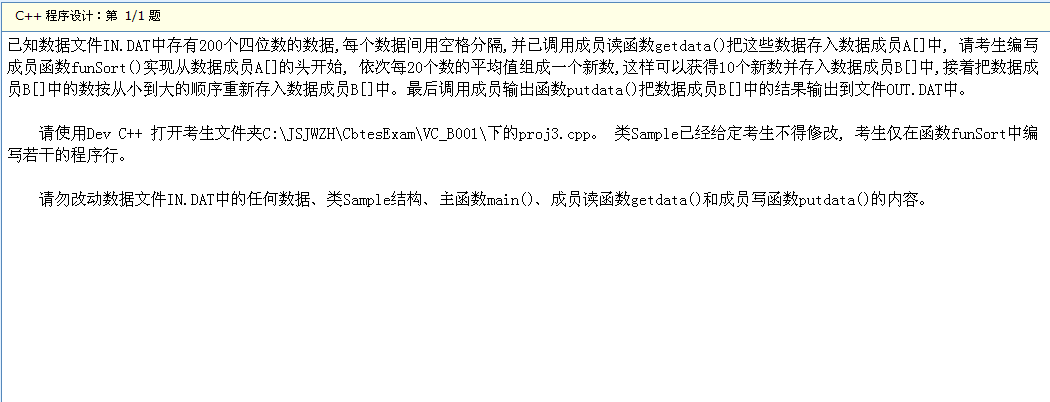
fstream fp ;

fp.open(".\\out.dat", ios::out) ;

for(int i = 0 ; i < 10 ; i++) fp << s.A[i] << endl ;

fp.close() ;

}



#include <iostream>

#include <fstream>

using namespace std;

#define Max 200

class Sample

{

int A[Max] ;

double B[10] ;

public:

Sample()

{

for(int i = 0 ; i < 10 ; i++) B[i] = 0 ;

}

int getNumberA(Sample s, int k) { return s.A[k] ; }

double getNumberB(Sample s, int k) { return s.B[k] ; }

void getdata(Sample &s) ; // 读数据函数

void funSort(Sample &s) ; // 要求考生编写的函数

void putdata(Sample &s) ; // 写结果函数

};

void Sample::funSort(Sample &s)

{

int i = 0,j;

int sum = 0;

int t = 0;

while(i<Max)

{

for(j = i;j<i+20;j++)

{

sum += s.A[j];

}

s.B[t] = (double) sum / 20;

sum = 0;

t++;

i+=20;

}

for(int n = 0;n<10;n++)

{

for(int m = n+1;m < 10; m++)

{

double temp = 0;

if(s.B[n] > s.B[m])

{

temp = s.B[n];

s.B[n] = s.B[m];

s.B[m] = temp;

}

}

}

}

int main()

{

int i;

Sample s ;

s.getdata(s) ;

cout << "读取的数据如下:" << endl ;

for(int i = 0 ; i < Max ; i++) {

cout << s.getNumberA(s, i) << " " ;

if((i + 1) % 10 == 0) cout << endl ;

}

cout << endl ;

s.funSort(s) ;

s.putdata(s) ;

cout << "输出结果:" << endl ;

for(i = 0 ; i < 10 ; i++)

cout << s.getNumberB(s, i) << " " ;

cout << endl ;

}

//\*\*\*\*\*\*\*\*\*\*\*以下请考生不要修改\*\*\*\*\*\*\*\*\*\*\*\*

void Sample::getdata(Sample &s) // 从数据文件中读取数据

{

fstream fp ;

fp.open(".\\in.dat", ios::in) ;

for(int i = 0 ; i < Max ; i++) fp >> s.A[i] ;

fp.close() ;

}

void Sample::putdata(Sample &s) // 写结果文件

{

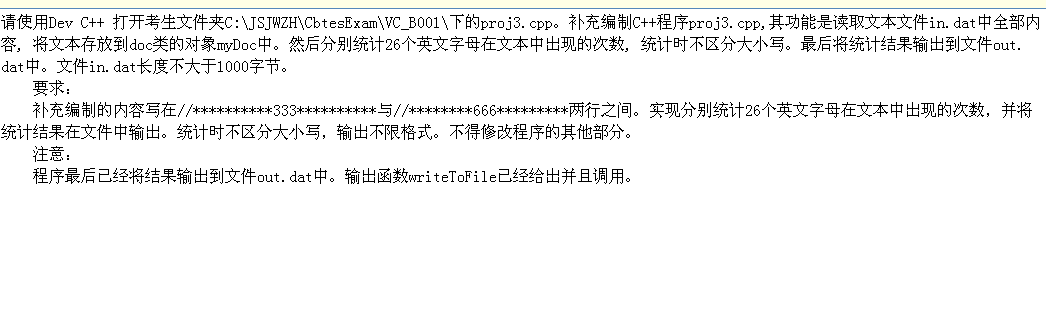
fstream fp ;

fp.open(".\\out.dat", ios::out) ;

for(int i = 0 ; i < 10 ; i++) fp << s.B[i] << endl ;

fp.close() ;

}



不大于1000，26个英文单词出现的次数

// proj3.cpp

#include<iostream>

#include<fstream>

#include<cstring>

using namespace std;

class doc

{

private:

char \*str; //文本字符串首地址

int counter[26]; //用于存放26个字母的出现次数

int length; //文本字符个数

public:

//构造函数，读取文件内容，用于初始化新对象。filename是文件名字符串首地址。

doc(const char \*filename);

void count(); //统计26个英文字母在文本中出现的次数，统计时不区分大小写。

~doc();

void writeToFile(const char \*filename);

};

doc::doc(const char \*filename)

{

ifstream myFile(filename);

int len=1001,tmp;

str=new char[len];

length=0;

while((tmp=myFile.get())!=EOF)

{

str[length++]=tmp;

}

str[length]='\0';

myFile.close();

for(int i=0; i<26; i++)

counter[i]=0;

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*333\*\*\*\*\*\*\*\*\*\*\*

void doc::count()

{

char \*s=str;

int index;

while(\*s)

{

if(\*s>='a'&&\*s<='z'||\*s>='A'&&\*s<='Z')

{

index=\*s>='a'?\*s-'a':\*s-'A';

counter[index]++;

}

s++;

}

for(int i=0;i<26;i++)

{

cout<<(char)('a'+i)<<":"<<counter[i]<<" ";

}

}

//\*\*\*\*\*\*\*\*\*\*666\*\*\*\*\*\*\*\*\*

doc::~doc()

{

delete [] str;

}

void doc::writeToFile(const char \*filename)

{

ofstream outFile(filename);

for(int i=0; i<26; i++)

outFile<<counter[i]<<endl;

outFile.close();

}

int main()

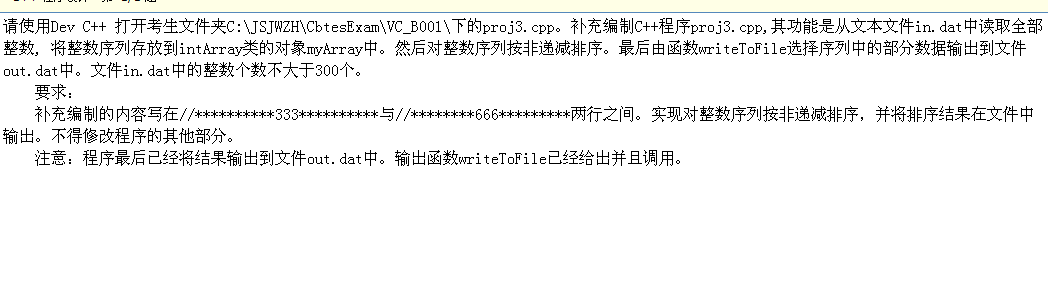
{

doc myDoc(".\\in.dat");

myDoc.count();

myDoc.writeToFile(".\\out.dat");

}



不大于300，非递减排序

// proj3.cpp

#include<iostream>

#include<fstream>

#include<cstring>

using namespace std;

class intArray

{

private:

int \*array; //整数序列首地址

int length; //序列中的整数个数

public:

//构造函数，从文件中读取数据用于初始化新对象。参数是文件名。

intArray(const char \*filename);

void sort(); //对整数序列按非递减排序

~intArray();

void writeToFile(const char \*filename);

};

intArray::intArray(const char \*filename)

{

ifstream myFile(filename);

int len=300;

array=new int[len];

length=0;

while(myFile>>array[length++]);

length--;

myFile.close();

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*333\*\*\*\*\*\*\*\*\*\*\*

void intArray::sort(){

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

for (int j = i; j < length; ++j)

if (array[i] > array[j])

{

int temp;

temp = array[i];

array[i] = array[j];

array[j] = temp;

}

for (int a = 0; a < length; ++a)

cout<< array[a]<< " ";

}

//\*\*\*\*\*\*\*\*\*\*666\*\*\*\*\*\*\*\*\*

intArray::~intArray()

{

delete [] array;

}

void intArray::writeToFile(const char \*filename)

{

int step=0;

ofstream outFile(filename);

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

{

outFile<<array[i]<<endl;

step++;

}

outFile.close();

}

int main()

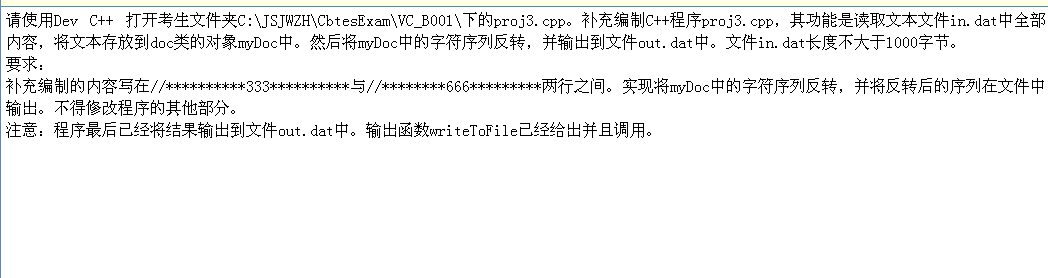
{

intArray myArray(".\\in.dat");

myArray.sort();

myArray.writeToFile(".\\out.dat");

}



不大于1000，字符序列反转

// proj3.cpp

#include<iostream>

#include<fstream>

#include<cstring>

using namespace std;

class doc

{

private:

char \*str; //文本字符串首地址

int length; //文本字符个数

public:

//构造函数，读取文件内容，用于初始化新对象。filename是文件名字符串首地址。

doc(const char \*filename);

void reverse(); //将字符序列反转

~doc();

void writeToFile(const char \*filename);

};

doc::doc(const char \*filename)

{

ifstream myFile(filename);

int len=1001,tmp;

str=new char[len];

length=0;

while((tmp=myFile.get())!=EOF)

{

str[length++]=tmp;

}

str[length]='\0';

myFile.close();

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*333\*\*\*\*\*\*\*\*\*\*\*

void doc::reverse()

{

int i=0,j=length-1;

char t;

while(i<j)

{

t=str[i];

str[i]=str[j];

str[j]=t;

i++,j--;

}

cout<<str;

}

//\*\*\*\*\*\*\*\*\*\*666\*\*\*\*\*\*\*\*\*

doc::~doc()

{

delete [] str;

}

void doc::writeToFile(const char \*filename)

{

ofstream outFile(filename);

outFile<<str;

outFile.close();

}

int main()

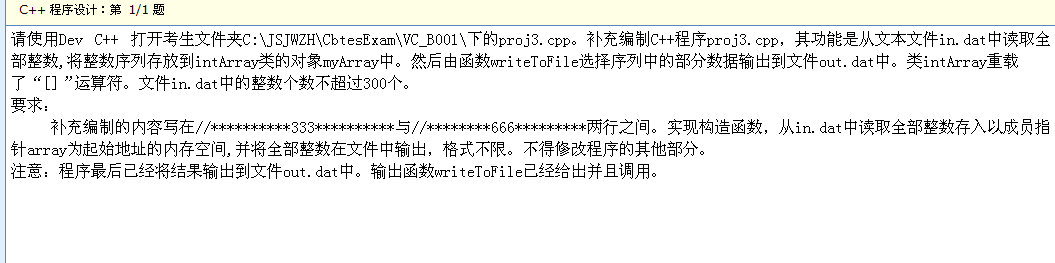
{

doc myDoc(".\\in.dat");

myDoc.reverse();

myDoc.writeToFile(".\\out.dat");

}



不超过300，重载[]运算符

// proj3.cpp

#include<iostream>

#include<fstream>

#include<cstring>

using namespace std;

class intArray

{

private:

int \*array; //整数序列首地址

public:

int length; //序列中的整数个数

//构造函数，从文件中读取数据用于初始化新对象。参数是文件名。

intArray(const char \*filename);

int & operator[](int i); //重载运算符，实现对象赋值。

~intArray();

};

//\*\*\*\*\*\*\*\*\*\*\*\*\*333\*\*\*\*\*\*\*\*\*\*\*

intArray::intArray(const char \*filename)

{

int len=300;

length=0;

array=new int[len];

ifstream myfile(filename);

while(myfile>>array[length++]);

length--;

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

cout<<array[i]<<" ";

cout<<endl;

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*666\*\*\*\*\*\*\*\*\*\*\*

int& intArray::operator[](int i)

{

return array[i];

}

intArray::~intArray()

{

delete [] array;

}

void writeToFile(const char \*filename, intArray& myArray)

{

int step=0;

ofstream outFile(filename);

for(int i=0; i<myArray.length; i=i+step)

{

outFile<<myArray[i]<<'\n';

step++;

}

outFile.close();

}

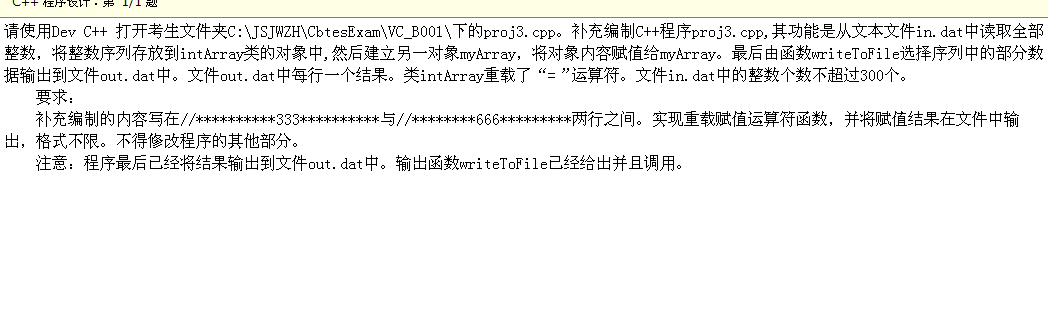
int main()

{

intArray myArray(".\\in.dat");

writeToFile(".\\out.dat",myArray);

}



// proj3.cpp

#include<iostream>

#include<fstream>

#include<cstring>

using namespace std;

class intArray

{

private:

int \*array;

int length;

public:

intArray(const char \*filename);

intArray();

intArray & operator=(const intArray & src);

~intArray();

void writeToFile(const char \*filename);

};

intArray::intArray()

{

length=10;

array=new int[length];

}

intArray::intArray(const char \*filename)

{

ifstream myFile(filename);

array=new int[300];

length=0;

while(myFile>>array[length++]);

length--;

myFile.close();

}

//\*\*\*\*\*\*\*\*\*\*\*\*\*333\*\*\*\*\*\*\*\*\*\*\*

intArray &intArray::operator=(const intArray & src)

{

int n = 0; //保存传入的 \*arrayP对象 array的长度

int i = 0;

while(src.array[i] != 0) //记录传入的 \*arrayP对象 array的长度

{

n ++;

i ++;

}

array=new int[300];

for(int j = 0;j<n;j++)

array[j] = src.array[j]; //将\*arrayP对象 的成员\*array传给myArray的 \*array

length = n; // 将\*arrayP对象 的成员length传给myArray的 length

}

//\*\*\*\*\*\*\*\*\*\*666\*\*\*\*\*\*\*\*\*

intArray::~intArray()

{

delete [] array;

}

void intArray::writeToFile(const char \*filename)

{

int step=0;

ofstream outFile(filename);

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

{

outFile<<array[i]<<endl;

step++;

}

outFile.close();

}

int main()

{

intArray \*arrayP=new intArray(".\\in.dat");

intArray myArray;

myArray=\*arrayP;

delete arrayP;

myArray.writeToFile(".\\out.dat");

}