# 2019

## 一、编程计算 1/1!-1/3!+1/5!-...+(-1)(n+1)/(2n-1)!

源码

#include <iostream>

using namespace std;

int function(int n) {//求n阶导

if (n == 1)

return 1;

return n \* function(n - 1);

}

int main() {

double sum = 0;

int n = 1;

cin >> n;

int t = -1;

int j = 1;

for (int i = 1; i < n + 1; i++) {

t = -t;

j = t \* function(2 \* i - 1) \* j;

//cout << j << endl;

sum += 1.0 / j;

}

cout << sum << endl;

}

## 二、3位老师对某次数学竞赛进行了预测。他们的预测如下：

甲说：学生A得第一名，学生B得第三名。

乙说：学生C得第一名，学生D得第四名。

丙说：学生D得第二名，学生A得第三名。

竞赛结果表明，他们都说对了一半，说错了一半，并且**无并列名次**，试编程输出A、B、C、D各自的名次。

问题分析：用数1，2，3，4分别代表学生a，b，c，d获得的名次。问题就可以利用三重循环把所有的情况枚举出来。

int main() {

int a, b, c, d;

for(a=1;a<5;a++)

for(b=1;b<5;b++)

if(a!=b)

for (c = 1; c < 5; c++) {

if (c != a && c != b) {

d = 10 - a - b - c;

if(a!=d&&b!=d&&c!=d)

if ((a == 1 || b == 3) && (c == 1 || d == 4) && (d == 2 || a == 3))

cout << a << " " << b << " " << c << " " << d << endl;

}

}

}

结果：

A:4

B:3

C:1

D:2

## 三、给定链表节点的定义 struct Node{ int data; Node \*next; } 请编写一个函数，用递归的方式：对两个有序链表合并成一个有序链表。

这题可能是近年来最难的一题了

#include<iostream>

using namespace std;

struct Node{

int data;

Node \*next;

};

Node\* Sort(Node \*L1,Node \*L2){

Node \*p;

if(L1 == NULL)

return L2;

if(L2 == NULL)

return L1;

if(L1->data < L2->data){

p=L1;

p->next = Sort(L1->next,L2);

}

else

{

p = L2;

p->next = Sort(L2->next,L1);

}

return p;

}

void print(Node \*p){

Node \*t=p;

while(t){

cout<<t->data<<" ";

t=t->next;

}

cout<<endl;

}

int main(){

Node \*l1 = new Node();

Node \*l2 = new Node();

Node \*p;

p=l1;

cout<<"input l1:";

for(int i=0;i<5;i++)

{

cin>>p->data;

if(i!=4)

{

p->next = new Node();

p=p->next;

}

}

p=l2;

cout<<"input l2:";

for(int i=0;i<5;i++)

{

cin>>p->data;

if(i!=4)

{

p->next = new Node();

p=p->next;

}

}

print(l1);

print(l2);

Node \*l3 = Sort(l1,l2);

print(l3);

}

结果：

input l1:1 3 4 5 8

input l2:3 6 9 10 11

1 3 4 5 8

3 6 9 10 11

1 3 3 4 5 6 8 9 10 11

## 四、酒店场景。

要求：

定义一个客人类 Guest。包含成员属性：编号 Num、姓名 Name、房费 Fee、当前酒店入住人数 Count。

其中编号 Num 需要程序自动生成。

现在要求实现以下 Guest 的成员函数：构造函数、Show()显示 Guest 的信息、 GetCount()返回当前酒店入住的人数、GetTotalIncome()返回当前酒店的总收入。

并定义 3 个 Guest 对象来对成员函数进行测试。

源码：

Guest.h

class Guest {

private:

static int num;

static int count;

string name;

double Fee;

static double totalIncome;

public:

Guest(const string& name, const double& Fee) :name(name), Fee(Fee) { num++; count++; totalIncome += Fee; }

~Guest() {

count--;

}

void show() const {

cout << num << ' ' << count << " " << name << ' ' << Fee << endl;

}

static int getCount() {

return count;

}

static double GetTotalIncome() {

return totalIncome;

}

};

int Guest::num = 0;

int Guest::count = 0;

double Guest::totalIncome = 0;

int main()

{

Guest g1("abc", 100), g2("bcd", 200), g3("def", 300);

g1.show(); g2.show(); g3.show();

cout << Guest::GetTotalIncome() << endl;

cout << Guest::getCount() << endl;

}

## 五、抽象类 Shape

要求：它拥有一系列虚函数：Input()输入类需要的信息、Show() 显示类的信息、Perimeter()计算周长、Area()计算面积。

先定义 Circle、Square、 Triangle 来继承 Shape 并实现其虚函数。要求创建 Circle、Square、Triangle 的对象，用基类指针指向这些对象，并调用成员函数

Shape.h

#pragma once

#include <iostream>

#include <cmath>

using namespace std;

class Shape //抽象类（拥有纯虚函数）

{

public:

virtual void Input() = 0;//纯虚函数

virtual void Show() = 0;

virtual double Perimeter() = 0;

virtual double Area() = 0;

};

class Circle :public Shape {//继承抽象类

private:

double r;

public:

void Input();

void Show();

double Perimeter();

double Area();

};

class Square :public Shape {

private:

double length;

double width;

public:

void Input();

void Show();

double Perimeter();

double Area();

};

class Triangle :public Shape {

private:

double A;

double B;

double C;

public:

void Input();

void Show();

double Perimeter();

double Area();

};

Shape.cpp

#include "Shape.h"

void Circle::Input() {

cout << "Circle R:";

cin >> this->r;

while (r <= 0) {

cout << "R is error,please input again, R=";

cin >> this->r;

}

}

void Circle::Show() {

cout << "R=" << this->r << endl;

}

double Circle::Perimeter() {

return 2 \* r \* 3.1415926;

}

double Circle::Area() {

return 3.1415926 \* r \* r;

}

void Square::Input() {

cout << "Square length and width=";

cin >> this->length >> this->width;

while (this->length <= 0 || this->width <= 0) {

cout << "error" << endl;

cout << "Square length and width=";

cin >> this->length >> this->width;

}

}

void Square::Show() {

cout << "length=" << this->length << ",width=" << this->width << endl;

}

double Square::Perimeter() {

return 2 \* (this->width + this->length);

}

double Square::Area() {

return this->width \* this->length;

}

void Triangle::Input() {

cout << "Triangle A, B and C=";

cin >> this->A >> this->B >> this->C;

while ((A + B) <= C || (A + C) <= B || (B + C) <= A || A <= 0 || B <= 0 || C <= 0) {

cout << "error!" << endl;

cout << "Triangle A, B and C=";

cin >> this->A >> this->B >> this->C;

}

}

void Triangle::Show() {

cout << "A=" << this->A << ",B=" << this->B << ",C=" << this->C << endl;

}

double Triangle::Perimeter() {

return A + B + C;

}

double Triangle::Area() {

//海伦公式

double p = 0.5 \* (A + B + C);

return sqrt(p \* (p - A) \* (p - B) \* (p - C));

}

Main.cpp

#include <iostream>

#include "Shape.h"

using namespace std;

int main() {

Shape\* base;//抽象类定义为指针

Circle circle;

base = &circle;//子类赋值给父类

Shape\* circle1 = new Circle;//这一步等同于上面两步

Square square;

Triangle triangle;

base->Input();

base->Show();

cout << "Perimeter=" << base->Perimeter() << endl;

cout << "Area=" << base->Area() << endl;

}

参考：https://blog.csdn.net/weixin\_44190648/article/details/122148318

# 2018

## 一、输出A-Z的ascii码

源码

#include <iostream>

using namespace std;

int main() {

//输出A - Z的ascii码

char ch = 'A';

while ('A' <= ch && ch <= 'Z')

{

cout << ch << " " << (int)ch << endl;

ch += 1;//注意不是+'1',两个意义完全不一样

}

}

## 二、计算一元二次方程的两个根

源码:

#include <iostream>

#include <cmath>

#include <iomanip>//精度控制

using namespace std;

int main() {

double a = 0., b = 0., c = 0.;//a\*x\*x+b\*x+c=0

double x1 = 0., x2 = 0.;

cin >> a >> b >> c;

double delat = b \* b - 4 \* a \* c;

if (delat < 0) {

cout << "no solution" << endl;

return 0;

}

x1 = (-b + sqrt(delat)) / 2.0 / a;

x2 = (-b - sqrt(delat)) / 2.0 / a;

cout << fixed << setprecision(3) << x1 << endl;//精确到小数点后三位

cout << fixed << setprecision(3) << x2 << endl;

}

## 三、输出9\*9乘法表

源码：

#include <iostream>

using namespace std;

int main() {

for (int i=1; i < 10; i++) {

for (int j=1; j < i+1; j++) {

cout << j << "\*" << i << "=" << i \* j << "\t";

}

cout << endl;

}

}

结果：

1\*1=1

1\*2=2 2\*2=4

1\*3=3 2\*3=6 3\*3=9

1\*4=4 2\*4=8 3\*4=12 4\*4=16

1\*5=5 2\*5=10 3\*5=15 4\*5=20 5\*5=25

1\*6=6 2\*6=12 3\*6=18 4\*6=24 5\*6=30 6\*6=36

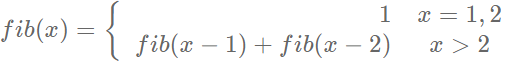
1\*7=7 2\*7=14 3\*7=21 4\*7=28 5\*7=35 6\*7=42 7\*7=49

1\*8=8 2\*8=16 3\*8=24 4\*8=32 5\*8=40 6\*8=48 7\*8=56 8\*8=64

1\*9=9 2\*9=18 3\*9=27 4\*9=36 5\*9=45 6\*9=54 7\*9=63 8\*9=72 9\*9=81

## 四、写个递归函数，求斐波那契

现在要求输入一个正整数 n ，请你输出斐波那契数列的第 n 项



源码：

#include <iostream>

using namespace std;

int Fibonacci(int n) {

if (n == 1 || n == 2)

return 1;

return Fibonacci(n - 1) + Fibonacci(n - 2);

}

int main() {

int n = 4;

cout << Fibonacci(n) << endl;

}

## 五、写一个函数，数组排序

由于是回忆版，没有要求排序是小到大还是大到小，同时也没有说明排序是排序方法，因此在此会使用三种排序方式来回答该问题

源码

#include<iostream>

#include<stdlib.h>

#include<time.h>

#define LEN 10

using namespace std;

int main(){

void quickSort(int Arr[],int left,int right);

void bubbleSort(int Arr[]);

void SelectSort(int Arr[]);

int Arr[LEN];

srand(time(NULL));

cout<<"Quick Sort"<<endl;

cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl;

cout<<"before"<<endl;

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

Arr[i] = rand()%98+1;

cout<<Arr[i]<<" ";

}

cout<<endl<<"after:"<<endl;

quickSort(Arr,0,LEN-1);

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

cout<<Arr[i]<<" ";

}

cout<<endl<<endl<<"Bubble Sort"<<endl;

cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl;

cout<<"before"<<endl;

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

Arr[i] = rand()%98+1;

cout<<Arr[i]<<" ";

}

cout<<endl<<"after:"<<endl;

bubbleSort(Arr);

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

cout<<Arr[i]<<" ";

}

cout<<endl<<endl<<"Select Sort"<<endl;

cout<<"\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*"<<endl;

cout<<"before"<<endl;

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

Arr[i] = rand()%98+1;

cout<<Arr[i]<<" ";

}

cout<<endl<<"after:"<<endl;

SelectSort(Arr);

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

cout<<Arr[i]<<" ";

}

return 0;

}

void quickSort(int Arr[],int left,int right){

if(left>=right)

return;

int i,j,base;

i = left;

j = right；

base = Arr[left];

while(i<j){

while(Arr[j]>=base && i<j)

j--;

while(Arr[i]<=base && i<j)

i++;

if( i < j ){

int t = Arr[i];

Arr[i] = Arr[j];

Arr[j] = t;

}

}

Arr[left] = Arr[i];

Arr[i] = base;

quickSort(Arr,left,i-1);

quickSort(Arr,i+1,right);

}

void bubbleSort(int Arr[]){

for(int i=0;i<LEN-1;i++)

for(int j=0;j<LEN-1-i;j++){

if(Arr[j]>Arr[j+1]){

int t = Arr[j];

Arr[j] = Arr[j+1];

Arr[j+1] = t;

}

}

}

void SelectSort(int Arr[]){

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

int min=i;

for(int j=i;j<LEN;j++){

if(Arr[j]<Arr[min])

min=j;

}

int t=Arr[min];

Arr[min]=Arr[i];

Arr[i]=t;

}

}

结果：

Quick Sort

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

before

24 8 45 47 23 70 37 96 9 16

after:

8 9 16 23 24 37 45 47 70 96

Bubble Sort

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

before

98 64 35 29 8 3 54 22 48 96

after:

3 8 22 29 35 48 54 64 96 98

Select Sort

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

before

56 67 61 4 53 14 14 96 79 74

after:

4 14 14 53 56 61 67 74 79 96

## 六、c=a/b 捕获异常

参考：<https://blog.csdn.net/weixin_44190648/article/details/122322641>

源码:

int divide(int x, int y) {

if (y == 0)

throw y;

return x / y;

}

int main() {

try {

divide(10, 0);

}

catch (int exception) {

cout << "被除数为" << exception << endl;

}

catch (...) {

cout << "未知异常" << endl;

}

}

#include<iostream>

using namespace std;

double division(double a,double b){

if(b==0)

throw "error";

return a/b;

}

int main(){

double a,b;

while(1){

cout<<"a/b, please input a b:";

cin>>a>>b;

try{

double result=division(a,b);

cout<<a<<"/"<<b<<"="<<result<<endl;

break;

}catch(const char \*msg){

cout<<"error!!"<<endl;

}

}

}

结果:

a/b, please input a b:3 0

error!!

a/b, please input a b:3 1.5

3/1.5=2

## 七、写一个类或结构体，包括学号姓名语数外成绩，输入n，新建n个，然后分别输入

struct score

{

double mid;

double final;

score(double mid\_=0,double final\_=0):mid(mid\_),final(final\_) {}

};

class Student

{

public:

Student(const string& name\_,const string& id\_,score math\_):name(name\_),id(id\_),math(math\_) {}

~Student(){};

void show() {

cout << name << ' ' << id << ' ' << math.mid << ' ' << math.final << endl;

}

private:

string name;

string id;

score math;

};

int main()

{

int n;

cin >> n;

vector<Student> v;

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

string name = "abc";

string id = "1";

score math(100,200);

Student s1(name, id, math);

v.push\_back(s1);

v[i].show();

}

}

## 八、用7的类，写三个函数，一个是将类用二进制追加到文件，一个是读取，还有一个是备份，把student.dat备份到student.bak

文件操作参考：<https://blog.csdn.net/weixin_44190648/article/details/122358591>

<https://blog.csdn.net/weixin_44190648/article/details/122362581>

#include <iostream>

#include <fstream>

#include <string>

using namespace std;

string filename1 = "C:\\Users\\19a405\\Desktop\\data.txt";

string filename2 = "C:\\Users\\19a405\\Desktop\\data1.txt";

class student {

friend ostream& operator<<(ostream& os, const student& st);//友元类

private:

int no;

string name;

public:

student(const int no = 0, const string& name = "") :no(no), name(name) {}

};

ostream& operator<<(ostream& os, const student& st) {//输出函数重载

os << st.no << '\t' << st.name;

return os;

}

bool Write(const string& filename, const student& st1) {

//c\_str()，把string转为char

//文件写入，二进制形式，追加

ofstream of(filename.c\_str(), ios\_base::out | ios\_base::app | ios\_base::binary);

if (of.is\_open() == false) {

cout << "文件" << filename << "打开失败！" << endl;

return false;

}

of.write((char\*)&st1, sizeof(student));//写入

of.close();

return true;

}

bool Read(const string& filename) {

ifstream ifile(filename.c\_str(), ios\_base::in);

if (ifile.is\_open() == false) {

cout << "文件" << filename << "打开失败！" << endl;

return false;

}

//读取全部内容

/\*char ch;

while (ifile.get(ch))

cout << ch;\*/

student st;

while (ifile.read((char\*)&st, sizeof(student))) {

cout << "\*\*\*\*\*\*\*\*\*\*\*" << endl;

cout << st << endl;//输出重载

}

ifile.close();

return true;

}

bool Backup(const string& filename1, const string& filename2) {

ifstream ifile(filename1, ios\_base::binary | ios\_base::in);

if (ifile.is\_open() == false) {

cout << "文件" << filename1 << "打开失败！" << endl;

return false;

}

ofstream of(filename2, ios\_base::binary | ios\_base::out | ios\_base::app);

if (of.is\_open() == false) {

cout << "文件" << filename2 << "打开失败！" << endl;

}

student st;

while (ifile.read((char\*)&st, sizeof(student))) {

of.write((char\*)&st, sizeof(student));

}

ifile.close();

of.close();

return true;

}

int main()

{

student st1(123, "abc"), st2(456, "def"), st3(789, "ghl");

Write(filename1, st1);

Write(filename1, st2);

Write(filename1, st3);

Backup(filename1, filename2);

Read(filename2);

}

# 2017

## 一.输入一个数字，为其高，一个符号\*，输出该符号组成的平行四边形形状

源码

#include<iostream>

using namespace std;

int main(){

int n;

cout<<"please input a number:";

cin>>n;

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

for(int j=n-i-1;j>=0;j--){

cout<<" ";

}

for(int k=0;k<n;k++){

cout<<"\*";

}

cout<<endl;

}

return 0;

}

结果：

please input a number:5

\*\*\*\*\*

\*\*\*\*\*

\*\*\*\*\*

\*\*\*\*\*

\*\*\*\*\*

## 二、猴子第一天摘了若干个桃子，当即吃了一半，还不解馋，又多吃了一个；第二天，吃剩下的桃子的一半，还不过瘾，又多吃了一个；以后每天都吃前一天剩下的一半多一个，到第10天想再吃时，只剩下一个桃子了。问第一天共摘了多少个桃子？

源码：

(递归)

//day10=day9-day9/2-1,所以day9=(day10+1)\*2

int peach(int day) {

if (day == 1)//第十天

return 1;

return 2 \* (peach(day-1) + 1);

}

void main() {

cout<<peach(10);

}

(for循环)

//day10=day9-day9/2-1,所以day9=(day10+1)\*2

void main() {

int sum = 1;

for (int i = 0; i < 9; i++) {

sum = (sum + 1) \* 2;

}

cout << sum << endl;

}结果：

1534

## 三、输入一系列的数字，输出他们正负数个数，输入0截止，要用函数实现

源码

int positive\_Num = 0;

int negative\_Num = 0;

void count\_Num(int n) {

if (n > 0)

positive\_Num++;

else if (n < 0)

negative\_Num++;

}

void main() {

int n;

while (cin>>n){

if (n == 0)

break;

count\_Num(n);

}

cout << "positive\_Num:" << positive\_Num << endl;

cout << "negative\_Num:" << negative\_Num << endl;

}

## 四、是用一个类来记录学生成绩和他班级，里面有几个函数，静态成员和动态成员都有

因为题目是回忆版的，题目概述不清，根据之前看过的教材大概自己设置几个函数与静态成员

#include<iostream>

#include<string>

using namespace std;

class Student{

public:

Student(string n,int c,double sco){

name=n;

Class=c;

score=sco;

num++;

total+=sco;

average=total/(double)num;

}

static double Got\_Total(){

return total;

}

static int Got\_num(){

return num;

}

static double Got\_Ave(){

return average;

}

void display(); //显示个人信息

void Change\_Score(double n); //更改分数

void Change\_Class(int c); //更改班级

private:

string name;

int Class;

double score;

static double total;

static int num;

static double average;

};

void Student::display(){

cout<<"Name:"<<name<<endl;

cout<<"Class:"<<Class<<endl;

cout<<"Score:"<<score<<endl;

}

void Student::Change\_Class(int c){

Class=c;

cout<<"After Change:"<<endl;

display();

}

void Student::Change\_Score(double s){

total-=score;

score=s;

total+=score;

average=total/(double)num;

cout<<"After Change:"<<endl;

display();

}

double Student::total = 0;

int Student::num = 0;

double Student::average = 0;

int main(){

Student Stu[5]={

Student("Obama",1,99.0),

Student("Mike",1,77.5),

Student("Jackson",2,87.0),

Student("Cherry",3,93.5),

Student("Oscar",4,89.0)

};

for(int i=0;i<5;i++)

{

cout<<endl;

Stu[i].display();

}

cout<<endl<<"Total:"<<Student::Got\_Total()<<endl;

cout<<"Numbers of Student:"<<Student::Got\_num()<<endl;

cout<<"Average:"<<Student::Got\_Ave()<<endl;

cout<<endl;

Stu[0].Change\_Score(100);

cout<<endl<<"Total:"<<Student::Got\_Total()<<endl;

cout<<"Numbers of Student:"<<Student::Got\_num()<<endl;

cout<<"Average:"<<Student::Got\_Ave()<<endl;

}

结果：

Name:Obama

Class:1

Score:99

Name:Mike

Class:1

Score:77.5

Name:Jackson

Class:2

Score:87

Name:Cherry

Class:3

Score:93.5

Name:Oscar

Class:4

Score:89

Total:446

Numbers of Student:5

Average:89.2

After Change:

Name:Obama

Class:1

Score:100

Total:447

Numbers of Student:5

Average:89.4

## 五、重载+，++，=等简单运算符

这题跟上面一样也是不太清楚，所以我以时间为例写一个重载运算符

参考：<https://blog.csdn.net/weixin_44190648/article/details/122018049>

#include<iostream>

using namespace std;

class Time{

private:

int sec;

int min;

public:

Time(int m=0,int s=0){

min=m;

sec=s;

}

void operator = (Time a){

min=a.min;

sec=a.sec;

}

void display(){

cout<<min<<":"<<sec<<endl;

}

friend Time operator +(Time &a,Time &b);

Time operator ++ (); //前置

Time operator ++ (int); //后置

};

Time operator +(Time &a,Time &b){

Time t;

t.sec=a.sec+b.sec;

t.min=a.min+b.min;

if(t.sec>=60)

{

t.sec-=60;

t.min++;

}

return t;

}

Time Time::operator++(){

this->sec++;

if(this->sec>=60)

{

this->sec-=60;

this->min++;

}

return \*this;

}

Time Time::operator++(int){

Time t=\*this;

this->sec++;

if(this->sec>=60)

{

this->sec-=60;

this->min++;

}

return t;

}

int main(){

Time a;

cout<<"a:";

a.display();

cout<<"++a:";

++a;

a.display();

cout<<endl;

cout<<"a++ (b) :";

Time b=a++;

b.display();

cout<<"a:";

a.display();

cout<<"a+b:";

Time c;

c=a+b;

c.display();

}

结果：

a:0:0

++a:0:1

a++ (b) :0:1

a:0:2

a+b:0:3

————————————————

版权声明：本文为CSDN博主「SampsonTse」的原创文章，遵循CC 4.0 BY-SA版权协议，转载请附上原文出处链接及本声明。

原文链接：https://blog.csdn.net/SampsonTse/article/details/104440280

# 2015

## 1.大小写字母转换

#include<stdio.h>

void main()

{

char x='a';

scanf("%c",&x);

if( x>='A' && x<='Z') { x=x+32; }

else if( x>='a' && x<='z') { x=x-32;  }

printf("%c\n",x); }

}

## 2.回文数判断

输入n，若不是回文数，则将n反转和n求和得到新的n，再判断n是否为回文数，问几次这样的操作可以使得最开始的n变成回文数。

注意 ，判断回文数只要翻转 后时候与源数字一样即可判断

#include<stdio.h>

#include<stdlib.h>

#include<math.h>

#include <algorithm>

#include<iostream>

#include<string.h>

using namespace std;

int change(int x)

{

int sum=0;

while(x)

{

sum=sum\*10+x%10;

x/=10;

}

return sum;

}

int judge(int a)

{

int b=change(a);

if(a-b==0)

return 1;

else

return 0;

}

int main()

{

int x;

while(scanf("%d",&x)==1)

{

int i=0;

while(1)

{

i++;

x+=change(x);

if(judge(x))

break;

}

printf("%d %d\n",i,x);

}

}

#include <iostream>

using namespace std;

int reverse(int n) {

int temp = 0;

while (n / 10 != 0) {

temp = 10 \* temp + n % 10;

n /= 10;

}

temp = 10 \* temp + n % 10;

return temp;

}

void main() {

int n;

cin >> n;

int temp = reverse(n);

int times = 0;

while (n != temp) {

n += temp;

temp = reverse(n);

times++;

}

cout << times << endl;

}

## 3.约瑟夫环

#include <iostream>

#include<stdio.h>

#include<algorithm>

#include<string.h>

using namespace std;

int judge(int a[],int n)

{

int sum=0;

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

if(a[i]==1) sum++;

return sum;

}

int main()

{

int i,j;

int n,m; //n个人，从1报数到m

scanf("%d %d",&n,&m);

int \*p=(int \*)malloc(n\*sizeof(int));

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

\*(p+i)=1;

int baoshu=0;

for(i=0;;i++)

{

if(p[i%n]==1)

{

baoshu++;

if(baoshu%m==0)

p[i%n]=0;

}

if(judge(p,n)==1)

break;

}

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

if(p[i]==1)

printf("%d\n",i+1);

}

数学方法

#include<stdio.h>

int main()

{

int n,m,s=0;

scanf("%d%d",&n,&m);

int i;

for(i=2;i<=n;i++)

s=(s+m)%i;

printf("%d\n",s+1);

}

# 2014

## 一、判断输入的数是否为素数

bool isPrime(int n) {

if (n == 1)

return false;

for (int i = 2; i < sqrt(n) + 1; i++) {

if (n % i == 0)

return false;

}

return true;

}

## 二、打印1~100之间的素数

#include <iostream>

#include <cmath>

#include <iomanip>

using namespace std;

void printPrime() {

int num[101]; //定义一个数组存放1~100

for (int i = 1; i < 101; i++) {

num[i] = i; //初始化数组

}

num[1] = -1; //首先1肯定不是素数，置为-1

for (int i = 2; i < sqrt(100); i++) { //遍历

for (int j = i + 1; j < 101; j++) {

if (num[i] != -1 && num[j] != -1 && num[j] % num[i] == 0) { //可以整除就不是素数

num[j] = -1; //不是素数置为-1

}

}

}

int count = 0; //统计个数

for (int i = 1; i < 101; i++) {

if (num[i] != -1) {

count++;

cout<<setw(5)<<num[i]; //打印

if (count % 5 == 0) {

cout << endl;

}

}

}

cout << endl;

cout << "count=" << count;

}

int main() {

printPrime();

return 0;

}

## 三、在一个有序数列中插入一个数，使得数列依然有序，并且把最大的那个数剔除出队列

比如 3,5,9,12， 插入6，然后得 3,5,6,9

#include <iostream>

using namespace std;

int main() {

int num[100]; //定义一个较大的数组

int len; //输入数字序列长度

cout << "请输入数字序列的长度";

cin >> len;

cout << "请输入长度为" << len << "的数字序列";

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

cin >> num[i]; //输入数字序列

}

cout << "请输入你要插入的数字";

int k;

cin >> k; //输入要插入的数字

for (int i = len - 1; i >= 0; i--) { //从最后一个开始遍历

if (k < num[i]) { //如果num[i]>k,则num[i]后移一位

num[i + 1] = num[i];

}

else {

num[i + 1] = k; //如果num[i]>k,则将k插入到num[i+1]，并结束循环

break;

}

}

//不修改len的长度就自动把最后一个数删掉了

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

cout << num[i] << " "; //打印

}

return 0;

}

# 2013

## 一、编写一个程序，给出方程3x+2y-7z=5（其中，0≤x,y,z≤10），求满足方程的所有x，y和z，输出之。

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

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

for (int k = 0; k <= 10; k++) {

if (3 \* i + 2 \* j - 7 \* k - 5 == 0)

cout << i << ' ' << j << ' ' << k << endl;

}

## 二、键盘输入一个整数，判断每一个数字（0-9）在这个整数中重复出现的次数，输出重复出现的数字及其出现次数。比如：输入3321181，则输出3出现了2次、1出现了3次；输入2186，则输出没有重复数字。

#include <iostream>

using namespace std;

int main() {

int num[10] = {0};//记录每位出现次数

//非静态的数组，只要初始化一个，剩余的会自动初始化为0

int k; //输入

cin >> k;

while (k / 10 != 0) { //判断是否k为1位数

int temp = k % 10; //获得最末尾的数字

num[temp]++; //数组+1

k /= 10;

}

num[k]++; //最后一位单独处理

bool flag = false; //标记是否没有重复数字

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

if (num[i] > 1) { //存在重复数字

flag = true; //标志位修改

cout << i << "出现了" << num[i] << "次" << endl;

}

}

if (flag == false) {

cout << "没有重复数字"<<endl;

}

return 0;

}

## 三、类Person是一个描述人员信息的数据结构体，包括姓名（不定长）、性别、年龄。利用该结构体创建数组emp[5]，调用自身的Get()方法可以输入人员的信息，并通过Show()方法显示输入的信息。请编写程序完成上述功能。

#include <iostream>

#include <string>

using namespace std;

class Person { //类的定义

private:

string name;

string sex;

int age;

public:

void Get();

void Show();

};

void Person::Get() { //Get方法具体实现

cin >> name >> sex >> age;

}

void Person::Show() { //Show方法具体实现

cout << "name:" << name << endl;

cout << "sex:" << sex << endl;

cout << "age:" << age << endl;

}

int main() {

Person emp[5];

for (int i = 0; i < 5; i++) {

cout << "请输入第" << i + 1 << "个人的名字、性别、年龄";

emp[i].Get();

}

for (int i = 0; i < 5; i++) {

cout << "第" << i + 1 << "个人" << endl;

emp[i].Show();

}

}

————————————————

版权声明：本文为CSDN博主「XHHP」的原创文章，遵循CC 4.0 BY-SA版权协议，转载请附上原文出处链接及本声明。

原文链接：https://blog.csdn.net/weixin\_41799019/article/details/113736123