1.如果Action中存在多个方法时，我们可以使用!+方法名调用指定方法。

public class HelloWorldAction{

private String message;

....

public String execute() throws Exception{

this.message = "我的第一个struts2应用";

return "success";

}

public String other() throws Exception{

this.message = "第二个方法";

return "success";

}

}

设访问上面action的URL路径为： /struts/test/helloworld.action

要访问action的other() 方法，则采用：

/struts/test/helloworld!other.action

1. 数组赋给指针

//把整个数组的值赋给指针

#include <iostream>

**using** **namespace** std**;**

int main**(){**

loop**:**

int a**[**5**];**

int **\***p **=** a**;**

cout **<<** "please 输入数组的值：" **<<** endl**;**

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

cin **>>** **\*** **(**p **+** i**);**

**}**

**for** **(**p **=** a**;**p **<** a **+** 5**;**p**++){**

cout **<<** **\***p **<<**" "**;**

**}**

cout**<<**endl**;**

**goto** loop**;**

**return** 0**;**

**}**

1. 链表

#include <iostream>

using namespace std;

struct NODE{

int date; //一个数据

struct NODE \* next; //一个指向下一个结点的指针 这行的struct可有可无

};

NODE \* head;

NODE \* Create(){

NODE \* p = NULL;

NODE \* q = NULL;

head = NULL;

int x;

cin>>x;

while (x != 0){

p = new NODE;

p -> date = x;

if(head == NULL){

head = p;

}

else {

q -> next = p;

}

q = p;

cin >> x;

} //end while

if (head != NULL){

q -> next = NULL;

}

return (head);

} //end Create()

void displayList (NODE \*head){

while (head != NULL){

cout << head -> date <<" ";

cout << head -> next <<" ";

head = head -> next;

}

}

int main(){

displayList(Create());

cout << endl;

return 0;

}

1. 链表 用别名表达

#include <iostream>

using namespace std;

typedef struct NODE{

int date; //一个数据

struct NODE \* next; //一个指向下一个结点的指针 这行的struct可有可无

}List; //List 为结构体NODE的别名

List \* head;

List \* Create(){

List \* p = NULL;

List \* q = NULL;

head = NULL;

int x;

cin>>x;

while (x != 0){

p = new List;

p -> date = x;

if(head == NULL){

head = p;

}

else {

q -> next = p;

}

q = p;

cin >> x;

} //end while

if (head != NULL){

q -> next = NULL;

}

return (head);

} //end Create()

void displayList (List \*head){

while (head != NULL){

cout << head -> date <<" ";

cout << head -> next <<" ";

head = head -> next;

}

}

int main(){

displayList(Create());

cout << endl;

return 0;

}

1. 计算器代码收录

#include<iostream>

#include<vector>

#include<string>

#include<cmath>

#include<iomanip>

using namespace std;

string skipblacks(string str) //忽略空格

{

string return\_str="";

int length=str.length();

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

if(str[i]!=' ')

return\_str+=str[i];

return return\_str;

}

string str\_dele(string s,int m) //删除字符串str的第m个字符

{

int len=s.length();

char temp\_str[100];

if(m>len)

cout<<"删除的位置不对!"<<endl;

else

{

int j = 0;

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

{

if(i==m-1)

continue;

else

temp\_str[j++]=s[i];

}

}

for(int i=0,j=0;j<len-1;)

s[j++]=temp\_str[i++];

string sn = "";

for(int i = 0;i<s.size()-1;++i)

sn += s[i];

return sn;

}

vector<double> renew(vector<double> num,double tmpresult,int i)

{

vector<double> vnew;

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

vnew.push\_back(num[j]);

vnew.push\_back(tmpresult);

if(i < num.size()-2)

for(int j = i + 2;j<num.size();++j)

vnew.push\_back(num[j]);

return vnew;

}

vector<char> clearchch(vector<char> vch) //去除乘号除号

{

vector<char> nvch;

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

if(vch[i] != '\*'&&vch[i] != '/')

nvch.push\_back(vch[i]);

return nvch;

}

int main()

{

string str;

vector<double> result;//保存结果

while(getline(cin,str))

{

if(str == "0")

break;

vector<char> vch;

vector<int> vi;

string s = skipblacks(str);//忽略字符串中的空格

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

{

if((s[i]=='+')|(s[i]=='-')|(s[i]=='\*')|(s[i]=='/'))

vch.push\_back(s[i]); //保存运算符哈

else if(s[i]>='0'&&s[i]<='9')

vi.push\_back(s[i]-'0'); //保存数字

}

string ns = s;

vector<double> vpos;//记录运算符号在原字符串中的位置

vector<double> vnum;//记录每个数的位数

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

{

size\_t pos = ns.find(vch[i]);

ns = str\_dele(ns,pos+1);

vpos.push\_back(pos+i+1);

}

vnum.push\_back(vpos[0]-1);

for(int i = 0;i<vpos.size()-1;++i)

vnum.push\_back(vpos[i+1]-vpos[i]-1);

double tmpsum = 0;

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

tmpsum += vnum[i];

double tmps = s.size()-vch.size()-tmpsum;

vnum.push\_back(tmps);

vector<double> num;//将vi转化为数字

double tmpn = 0;

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

{

double sum = 0;

tmpn += vnum[i];

for(int j = 0;j<vnum[i];++j)

sum += vi[j+tmpn-vnum[i]]\*pow(10,vnum[i]-1-j);

num.push\_back(sum);

}

vector<double> vnew = num;

int ccnt = 0;

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

{

if((vch[i] == '\*')|(vch[i] == '/'))

{

double tmpresult;

if(vch[i] == '\*')

tmpresult = num[i]\*num[i+1];

else

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

vnew = renew(vnew,tmpresult,i-ccnt);//更新结果

++ccnt; //统计已经计算过多少个\*和/。

}

}

vector<char> nvch = clearchch(vch);//去除乘号和除号只剩下加减号

double restmp = vnew[0];

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

{

if(nvch[i] == '+')

restmp = restmp + vnew[i+1];

else

restmp = restmp - vnew[i+1];

}

result.push\_back(restmp);

}

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

cout<<fixed<<setprecision(2)<<result[i]<<endl; //保留两位小数

system("pause");

return 0;

}

1. <https://zhidao.baidu.com/question/457931418227164765.html>

改变dev输出控制台颜色