第四次实验报告

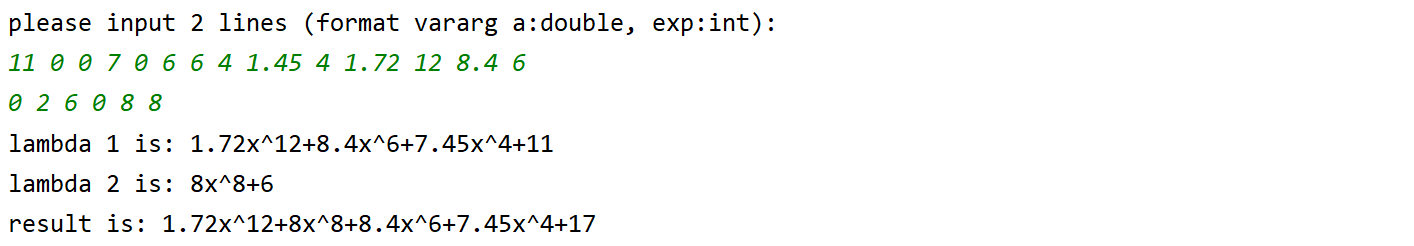
实验题目

用list迭代器实现多项式的合并

实验过程

1. 输入的处理，起初使用解析表达式的方法(例如输入为11x^0+0x^7+4)，但发现操作过于复杂，改为使用vararg\*的方式进行赋值
2. 对输入表达式的调整，包括排序和合并。先使用stl::sort()使表达式从大到小排列，在使用append方法合并同类项。
3. 输出表达式的处理，使用单元分割的方法进行输出。
4. 合并表达式的函数join，使用了归并排序的方法进行表达式的合并。并使用了range\_for。

效果截图



代码

#include <iostream>

#include <string>

#include <sstream>

#include <list>

using namespace std;

*/\*\**

*\* 一个单项式*

*\*/*

class math\_one{

public:

math\_one(): a(0.0),exp(0){}

math\_one(double a, int exp): a(a),exp(exp){}

double a;

int exp;

friend ostream&operator << (ostream& oStream, math\_one& value){

oStream << value.a;

if(value .exp >= 1){

oStream << 'x';

}

if(value.exp >= 2){

oStream << '^' << value.exp;

}

return oStream;

}

};

void print(list<math\_one\*>\* l){

if(l->empty()){

cout << '0';

} else {

for(auto v: \*l){

if(v->a > 0 && v!= \*l->begin()){

cout << '+';

}

cout << \*v;

}

}

cout << endl;

}

bool compare(math\_one\* first, math\_one\* second){

return (first->exp > second->exp);

}

void append(list<math\_one\*>\* l, math\_one\* e){

int b\_exp = -1;

if(!l->empty()){

b\_exp = l->back()->exp;

}

if(e->a == 0){

} else if(b\_exp == e->exp){

l->back()->a+= e->a;

} else {

l->push\_back(e);

}

}

list<math\_one\*>\* format(list<math\_one\*>\* l){

auto nl = new list<math\_one\*>(\*l);

nl->sort(compare);

*//cout << "r"; print(nl);*

l->clear();

*// int b\_exp = -1;*

for(auto v: \*nl){

append(l, v);

*// if(v->a == 0){*

*//*

*// } else if(v->exp == b\_exp){*

*// l->back()->a+= v->a;*

*// } else {*

*// l->push\_back(v);*

*// b\_exp = v->exp;*

*// }*

}

*//\*l = \*nl;*

return l;

}

list<math\_one\*>\* join(list<math\_one\*>\* a, list<math\_one\*>\* b){

auto l = new list<math\_one\*>();

auto p = b->begin();

for(auto v: \*a){

while(p!= b->end() && compare(\*p,v)){

append(l,\*p);

*//l->push\_back(\*p);*

++p;

}

append(l, v);

*//l->push\_back(v);*

}

while(p != b->end()){

append(l,\*p);

*//l->push\_back(\*p);*

++p;

}

return format(l);

}

list<math\_one\*> \* parse(const string& input){

auto l = new list<math\_one\*>;

double a; int exp;

istringstream iStrStream = istringstream(input);

while(!iStrStream.eof() && iStrStream >> a && iStrStream >> exp){

l->push\_back(new math\_one(a, exp));

}

return l;

}

int main(){

cout << "please input 2 lines (format vararg a:double, exp:int): " << endl;

string inputA, inputB;

getline(cin,inputA);

getline(cin,inputB);

list<math\_one\*> \*first, \*second, \*sub;

cout << "lambda 1 is: ";

first = format(parse(inputA));

print(first);

cout << "lambda 2 is: ";

second = format(parse(inputB));

print(second);

sub = join(first, second);

cout << "result is: ";

print(sub);

delete first;

delete second;

delete sub;

return 0;

}