匹配需求问题源码说明

1.基本说明

代码采用 C++编写，并采用了一些 C++11 的特性（主要是文件输入输出流部分），因此需要使用支持 C++11 的编译器编译，可使用 Visual Studio 2017 顺利编译并运行。

2.文件说明

Main.cpp ： 主函数

Exchange_Item.h ：（打款人/收款人）类定义和类成员函数声明

Exchange_Item.cpp ： 类成员函数的定义

requirematch.h ： 匹配需求算法核心函数的声明

requirematch.cpp ： 匹配需求算法核心函数的定义

CSVRow.h ： 从 CSV 读入数据的类定义

SubSetSUM.h ： 子集和问题算法的声明（匹配需求问题中没有用到）

SubSetSUM.cpp ： 子集和问题算法的定义（匹配需求问题中没有用到）

2.1 Exchange_Item.h

```
#pragma once
#ifndef EXCHANGEITEM
#define EXCHANGEITEM
#include<string>
#include<vector>
#include<map>
class Exchange_Item
{
public:
    Exchange_Item() : User_ID("000000"), money(0), initOrder(0),
res_money(0), weight(0), max_weight(5) {};//构造函数

    Exchange_Item(const std::string str, int m, int order, int res_m,
```

```
int w, int max_w) : User_ID(str), money(m), initOrder(order),
res_money(res_m), weight(w), max_weight(max_w) {};//构造函数
  //构造函数
    Exchange_Item(const std::string str, int m) :User_ID(str),
money(m), initOrder(0), res_money(m), weight(0), max_weight(4) {};
    //构造函数
    Exchange_Item(const std::string str, int m, int order) :
User_ID(str), money(m), initOrder(order), res_money(m), weight(0),
max_weight(4) {};

    //显示声明引用
    Exchange_Item(const Exchange_Item&) = default;
    Exchange_Item& operator=(const Exchange_Item&) = default;

    bool setID(std::string str);//设置ID
    std::string getID() const; //获取ID
    bool setMoney(int m);//设置金额
    int getMoney() const; //获取金额
    int getInitOrder() const; //获取初始顺序值
    bool setWeight(int w);//设置权重
    int getWeight() const; //获取权重
    bool weightIncrease();//权重+1
    bool weightIncrease(int i); //权重+i
    bool setResMoney(int res);//设置余额
    int getResMoney() const; //获取余额
    bool addExchangeInfo(std::string str, int m);//增加交易信息
    std::map<std::string, int> getExchangeInfo();//获取交易信息
    std::vector<std::string> getExchangeID();//获取交易信息中的ID
    std::vector<int> getExchangeMoney();//获取交易信息中的金额
    int maxWeight() const;//设置最大权重
    void print();//打印对象信息

private:
    std::string User_ID;
    int money;
    int initOrder;//初始顺序
    int res_money;//余额
    int weight;//权重
    int max_weight;//最大权重
    std::vector<std::string> exchange_ID; //交易信息中的ID
    std::vector<int> exchange_money; //交易信息中的金额
};
#endif // !EXCHANGEITEM ;
```

2.2 requirematch.h

```
bool compID(const Exchange_Item& Item1, const Exchange_Item& Item2);
//比较用户ID，也就是字符串比较，第一个比第二个小则返回true

bool compOrder(const Exchange_Item& Item1, const Exchange_Item&
Item2);
//比较用户原始输入顺序，第一个的order比第二个order小则返回true

bool compItemNoMaxWeight(const Exchange_Item& Item1, const
Exchange_Item& Item2);
//不考虑权重，仅仅比较余额，第一个余额小，则返回true

bool compItem(const Exchange_Item& Item1, const Exchange_Item& Item2);
//考虑权重进行比较

bool readCSVdata(ExContainer& Rers, const std::string csvfilename);
// 从CSV读取数据到list容器中
bool readCSVdata(ExContainer& Rers, const std::string csvfilename, int
startOrder);
// 从CSV读取数据到list容器中并给定初始顺序标号

size_t dealRepetition(ExContainer& Rers, ExContainer& Ters);
// 先排序，Rers和Ters中可以抵消的数据发生抵消并记录，返回抵消的笔数
difftype finishedNum(ExContainer& Rers);
// 获取liast中余额为0个个数

bool has2sum(Itertype& first, Itertype& last, Exchange_Item& target,
Itertype& it1, Itertype& it2);
// 在迭代器指定范围[first,last)内，查找是否有两个元素的余额的和与target的余额相
等
//如果有则将指向两元素迭代器赋给it1和it2，并返回true

void exchangeFunc(ExContainer& largeCont, ExContainer& smallCont,
    ExSaver & largeSaver, ExSaver & smallSaver);
//两个容器中元素发生交易，将完成交易的存储在另一容器并从原先容器中删除

std::vector<int> getExTimes(ExSaver& Rers);
// 返回每个元素的交易次数的容器

int sumExTimes(ExSaver& Rers);
```

```
// 返回总交易次数

ExSaver::iterator maxExTimes(ExSaver& Rers);
// 返回最大交易次数

void writeResult(ExSaver& Rers, std::string result_file);
//将结果写进CSV

void writeLog(std::string result_log, ExSaver& Rers, ExSaver& Ters,
double t);
//将运行时间，最大交易次数，总交易次数写进 log
```

2.3 SubSetSUM.h

```
void printSubsetSum(const std::vector<int> & w, const
std::vector<bool>& x);
//输出已经找到的符合要求的子集

bool subsetSum(const std::vector<int>& w, std::vector<bool> x, int
sum, int targetsum, size_t k);
//寻找子集和的函数，先寻找元素较多的子集

bool subsetSum(const std::vector<int>& w, std::vector<bool> x, int
targetsum, int n);
////寻找子集和的函数，先寻找元素较少的子集

void test_subsetsum();
//测试 printSubsetSum 函数，subsetSum 函数
```