/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

1）判断是否足够支付，不能则取消交易

2）根据零钱情况调整，零钱不足，则以稍高的价计算，需要找零

3）交易币以大面值为先，逐级判断

此程序有较大缺陷，有时可以不需要找零，而这里没有做到

到位的判断，可能是判断的起点有问题

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include<iostream.h>

struct money

{ //结构体——钱

int count; //货币数量

int value; //货币面值

};

money wallet[7]; //建立钱包

float price; //价格，需要支付的钱

int total; //钱包总额

int balance; //余额

int initiate()

{

//以下7行进行面值初始化

wallet[0].value = 1;

wallet[1].value = 2;

wallet[2].value = 5;

wallet[3].value = 10;

wallet[4].value = 20;

wallet[5].value = 50;

wallet[6].value = 100;

//总额、余额初始化

total = 0;

balance = 0;

int i;

for( i=0;i<7;i++ )

{

cout<<"请输入面值为"

<<wallet[i].value

<<"的货币数量:";

cin >>wallet[i].count; //设置钱币数量

}

cout<<"你知道要付多少钱吗";

cin >>price; //获取价格

for( i=0;i<7;i++ ) //计算钱包总额

total += wallet[i].count \* wallet[i].value;

return 0;

}

int enough()

{ //判断钱是否足够

price = (float) (price>(int)price ? (int)price+1:(int)price);

balance = total - (int)price; //计算余额

if( total>=(int)price )

return 1;

return 0;

}

int change()

{ //判断零钱情况

//零钱不足,则按稍高的价格进行计算

int change = 0;

int sum = 0;

sum += wallet[0].count \* wallet[0].value;

sum += wallet[1].count \* wallet[1].value;

change = (int) price%5;

if( sum>=change )

{

if( change%2 && wallet[0].count==0 )

{

price -= change;

price += 5;

}

}

else

{

price -= change;

price += 5;

}

sum += wallet[2].count \* wallet[2].value;

sum += wallet[3].count \* wallet[3].value;

sum += wallet[4].count \* wallet[4].value;

change = (int) price%50;

if( sum<change )

{

price -= change;

price += 50;

}

return 0;

}

int pay()

{

int count[7] = {0}; //需要的钱币数量

cout<<"你共有"<<total<<"元钱"<<endl;

cout<<"需要支付"<<price<<"元"<<endl;

if( !enough() ) //如果不够

{

cout<<"钱不够，我不卖了。。"<<endl;

return 0; //取消支付

}

change(); //根据零钱情况调整

cout<<"需要";

if( price>=100 )

{ //计算100币值的数量

count[6] = (int)price/100;

if( count[6]>wallet[6].count )

count[6] = wallet[6].count;

wallet[6].count -= count[6];

total -= count[6]\*100;

cout<<"\t100元纸币\t"

<<count[6]<<"张"<<endl;

price -= 100\*count[6];

}

if( price>=50 )

{ //计算50币值的数量

count[5] = (int)price/50;

if( count[5]>wallet[5].count )

count[5] = wallet[5].count;

wallet[5].count -= count[5];

total -= count[5]\*50;

cout<<"\t50元纸币\t"

<<count[5]<<"张"<<endl;

price -= 50\*count[5];

}

if( price>=20 )

{ //计算20币值的数量

count[4] = (int)price/20;

if( count[4]>wallet[4].count )

count[4] = wallet[4].count;

wallet[4].count -= count[4];

total -= count[4]\*20;

cout<<"\t20元纸币\t"

<<count[4]<<"张"<<endl;

price -= 20\*count[4];

}

if( price>=10 )

{ //计算10币值的数量

count[3] = (int)price/10;

if( count[3]>wallet[3].count )

count[3] = wallet[3].count;

wallet[3].count -= count[3];

total -= count[3]\*10;

cout<<"\t10元纸币\t"

<<count[3]<<"张"<<endl;

price -= 10\*count[3];

}

if( price>=5 )

{ //计算5币值的数量

count[2] = (int)price/5;

if( count[2]>wallet[2].count )

count[2] = wallet[2].count;

wallet[2].count -= count[2];

total -= count[2]\*5;

cout<<"\t5元纸币 \t"

<<count[2]<<"张"<<endl;

price -= 5\*count[2];

}

if( price>=2 )

{ //计算2币值的数量

count[1] = (int)price/2;

if( count[1]>wallet[1].count )

count[1] = wallet[1].count;

wallet[1].count -= count[1];

total -= count[1]\*2;

cout<<"\t2元纸币 \t"

<<count[1]<<"张"<<endl;

price -= 2\*count[1];

}

if( price>=1 )

{ //计算1币值的数量

count[0] = (int)price;

wallet[0].count -= count[0];

total -= count[0];

cout<<"\t1元纸(硬)币\t"

<<count[0]<<"张(枚)"<<endl;

}

cout<<"你还剩"<<balance<<"元"<<endl; //报告余额

return 0;

}

int main()

{

initiate();

pay();

return 0;

}

