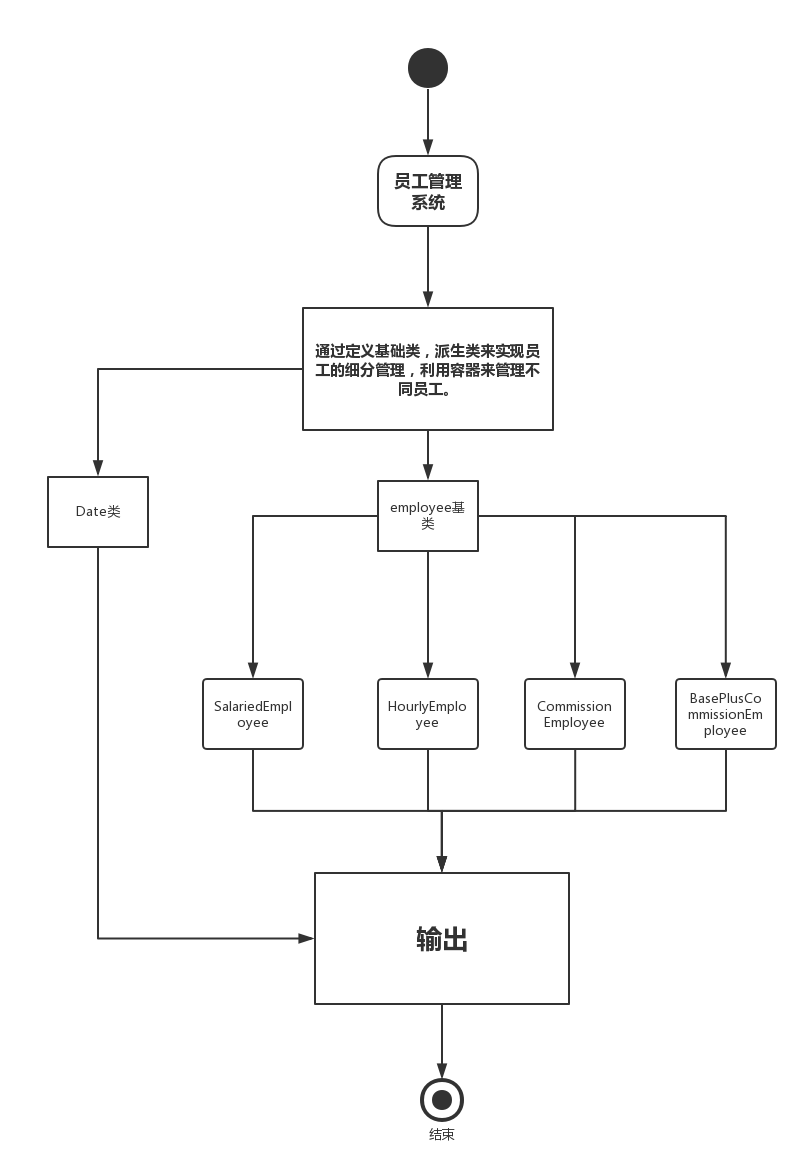
**《面向对象程序课程设计》上机实验报告**

班级：七班 学号：55160728 姓名：张叶钦

1. 实验题目

员工工资管理



1. 解决方案

通过定义基础类，派生类来实现员工的细分管理，利用容器来管理不同员工。

1. 程序清单

//类定义

#ifndef EMPLOYEE

#define EMPLOYEE

#include<iostream>

#include<string>

using namespace std;

class employee {

public:

//根据员工的名字和id构造员工对象

employee(string name,int id,int month): name(name),id(id),month(month){};

virtual int GetSalary() const {return 0;};//获取员工工资

string GetName() const { return name;}; //使外界得到员工姓名

int GetID()const{return id;};

protected:

string name;//员工姓名

int id;//员工ID

int year;//员工出生年份

int month;//员工出生月份

int day;//员工生日期

};

class SalariedEmployee:public employee{

public:

SalariedEmployee(string name,int id,int month):employee(name,id,month){};

virtual int GetSalary() const { return 500;}; //每周固定工资500

public:

void output(int m);

};

class HourlyEmployee:public employee{

public:

HourlyEmployee(string name,int id,int month):employee(name,id,month){};

virtual int GetSalary() const {

if(hours<40)

return hours\*2.5;

else

return 100+(hours-40)\*2.5\*1.5;

};

public:

void output(int m);

protected:

int hours;

};

class CommissionEmployee:public employee{

public:

CommissionEmployee(string name,int id,int month):employee(name,id,month){};

virtual int GetSalary() const { return Salesvolume\*2.5 ;};//假设佣金为2.5

void getsale(int s);

void output(int m);

protected:

int Salesvolume;

};

class BasePlusCommissionEmployee:public employee{

public:

BasePlusCommissionEmployee(string name,int id,int month):employee(name,id,month){};

virtual int GetSalary() const { return Salesvolume\*2.5+500 ;};//底薪为500

public:

void output(int m);

void getsale(int s);

protected:

int Salesvolume;

};

class Date{

public:

int year;

int month;

int day;

public:

Date();

Date(int a,int b,int c)

{

year=a;

month=b;

day=c;

}

void output();

};

#endif

//类的详细解释

#include<iostream>

#include"employee.h"

void SalariedEmployee::output(int m){

cout<<"Employee type:SalariedEmployee" <<endl;

cout<<"Employee name:"<<SalariedEmployee::GetName()<<endl;

cout<<"Employee ID:"<<SalariedEmployee::GetID()<<endl;

if(m==month){

cout<<"Happy birthday!\n";

cout<<"Employee salary:"<<SalariedEmployee::GetSalary()+100<<endl;

}

else

cout<<"Employee salary:"<<SalariedEmployee::GetSalary()<<endl;

}

void HourlyEmployee::output(int m){

cout<<"Employee type:HourlyEmployee" <<endl;

cout<<"Employee name:"<<HourlyEmployee::GetName()<<endl;

cout<<"Employee ID:"<<HourlyEmployee::GetID()<<endl;

if(m==month){

cout<<"Happy birthday!\n";

cout<<"Employee salary:"<<HourlyEmployee::GetSalary()+100<<endl;

}

else

cout<<"Employee salary:"<<HourlyEmployee::GetSalary()<<endl;

}

void CommissionEmployee::output(int m){

cout<<"Employee type:CommissionEmployee" <<endl;

cout<<"Employee name:"<<CommissionEmployee::GetName()<<endl;

cout<<"Employee ID:"<<CommissionEmployee::GetID()<<endl;

if(m==month){

cout<<"Happy birthday!\n";

cout<<"Employee salary:"<<CommissionEmployee::GetSalary()+100<<endl;

}

else

cout<<"Employee salary:"<<CommissionEmployee::GetSalary()<<endl;

}

void CommissionEmployee::getsale(int s){

Salesvolume=s;

}

void BasePlusCommissionEmployee::output(int m){

cout<<"Employee type:BasePlusCommissionEmployee" <<endl;

cout<<"Employee name:"<<BasePlusCommissionEmployee::GetName()<<endl;

cout<<"Employee ID:"<<BasePlusCommissionEmployee::GetID()<<endl;

if(m==month){

cout<<"Happy birthday!\n";

cout<<"Employee salary:"<<BasePlusCommissionEmployee::GetSalary()+100<<endl;}

else

cout<<"Employee salary:"<<BasePlusCommissionEmployee::GetSalary()<<endl;

}

void BasePlusCommissionEmployee::getsale(int s){

Salesvolume=s;

}}}//主函数

#include <iostream>

#include"employee.h"

int main ()

{

int i;

//string name;

//int id;

cout<<"Please choose the type of employee:\n";

cout<<"1.周薪雇员 2.时薪雇员 3.佣金雇员 4.带底薪佣金雇员\n";

cin>>i;

SalariedEmployee a("zhangsan",0001);

Date a1(1998,1,1);

HourlyEmployee b("Lisi",0002);

Date b1(1999,2,3);

CommissionEmployee c("Wangwu",0003);

Date c1(1980,9,7);

c.getsale(100);

BasePlusCommissionEmployee d("AA",0004);

Date d1(1982,6,4);

d.getsale(100);

switch(i){

case 1:

a.output();

a1.output();

break;

case 2:

b.output();

b1.output();

break;

case 3:

c.output();

c1.output();

break;

case 4:

d.output();

d1.output();

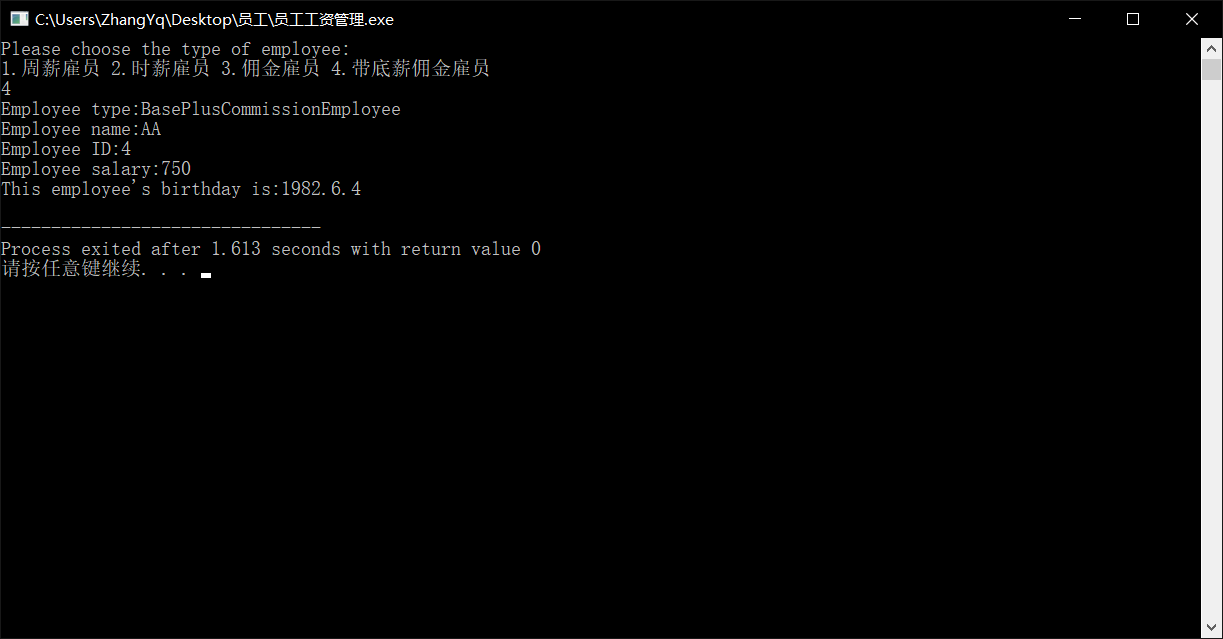
break;

}

return 0;

}

程序运行结果



1. 体会与总结

通过次程序，初步了解容器类的定义与使用。通过对不同类的不同与相同之处，选择合适的类的定义与继承，简化程序代码。提高工作效率。