北京邮电大学 2020——2021 学年第二学期

《C++程序设计》春季期中-课程作业

作业中不填写姓名、学号等能显露个人身份的信息	
注意事项	一、学生必须独立完成本作业。 **二、任何雷同或者有抄袭嫌疑的作业，会导致提交者的总评成绩为 0 分。** 三、学生可以通过阅读、查找纸质资料或者网上电子资料解答题目。但是，不允许**【抄袭】** 四、作业必须以 PDF 格式提交。作业文件命名方法如下： 1. PDF 文件命名为“MT-学号经过 MD5 算法转换后的 32 位小写字符串.pdf” 例如：学号为 2017211000 的同学，学号经过 MD5 算法变换后的 32 位小写字符串是“ad31152bdfa630e14613e8f3102fabc6” 所以该同学的 PDF 文件名为： MT-ad31152bdfa630e14613e8f3102fabc6.pdf 2. PDF 文件上传爱课堂期中教学单元中的作业提交部分 3. 一周后开启互评，每人必须完成互评才能得分 4. MD5 算法转方法 Linux 下： echo -n 2017211000 | md5sum # 其中 | 是竖线管道符 使用 https://www.sojson.com/encrypt_md5.html 转换学号 使用 https://md5jiami.51240.com/ 转换学号

Question 1 (50 points)

Create two classes called Traveler and Pager without default constructors, but with constructors that take an argument of type string, which they simply copy to an internal string variable. For each class, write the correct copy-constructor. Now inherit a class BusinessTraveler from Traveler and give it a member object of type Pager. Write the correct default constructor, a constructor that takes a string argument, a copy-constructor, and an assignment operator.

Question 2 (50 points)

Write a class with one virtual function and one non-virtual function. Inherit a new class, make an object of this class, and upcast to a pointer of the base-class type. Use the clock( ) function found in <ctime> (you'll need to look this up in your local C library guide) to measure the difference between a virtual call and non-virtual call. You'll need to make multiple calls to each function inside your timing loop in order to see the difference.

评分标准

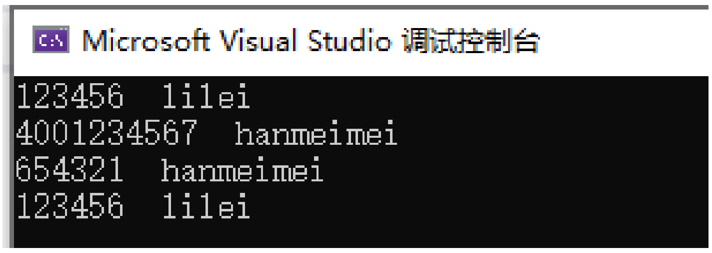
1. 输出结果和工程项目截图 10%。截图应清晰，大小合适；
2. 实现程序的基本要求：50%；
3. 代码要有必要的注释，可参考 doxygen、javadoc 或者 QT 类型的注释；代码应符合 C++11/14/17 标准；遵循合理的编码规范：20%；
4. 代码结构清晰、缩进合理；文档干净整洁；作业文档中的代码字体大小合理规整，代码有语法加亮 20%；
5. **【特别强调】**代码的注释和可读性是判分的重要依据。可读性差的代码，哪怕结果正确，也只能得到 50%的总分；
6. **【特别强调】**任何抄袭或者疑似抄袭的代码，都将导致你的作业被二次复审，并且无法通过本课程期末考核（第二年重修）。请保存好自己的作业，不要给任何人参考。

注意事项

1. 提交的文档请转为 pdf 格式
2. 请保留作业题目（本页之前的内容不要更改/删除）
3. 请保持作业回答干净整洁。

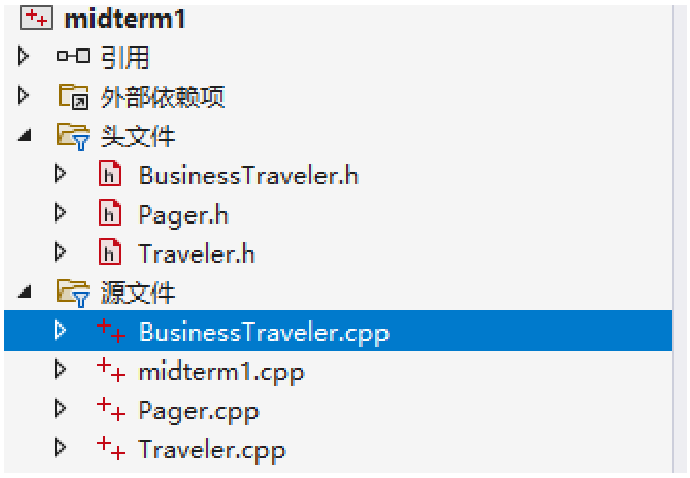
1. 作业 1

1.1 输出结果截图


Microsoft Visual Studio 调试控制台

```
123456  lilei
4001234567  hanmeimei
654321  hanmeimei
123456  lilei
```

1.2 工程项目截图


```
midterm1
  引用
  外部依赖项
  头文件
    BusinessTraveler.h
    Pager.h
    Traveler.h
  源文件
    BusinessTraveler.cpp
    midterm1.cpp
    Pager.cpp
    Traveler.cpp
```

1.3 Traveler.h

```cpp
#pragma once
#include<string>
using std::string;
class Traveler {
private:
    string NameOfTraveler;
public:
Traveler() = delete;
Traveler(string NameOfTraveler_);
Traveler(Traveler& traveler_);
string getNameOfTraveler()const;
void setNameOfTraveler(string NameOfTraveler_);
};
```

1.4. Traveler.cpp

```
#include"Traveler.h"
Traveler::Traveler(string NameOfTraveler_) {
    this->NameOfTraveler = NameOfTraveler_;
}

Traveler::Traveler(Traveler& traveler_) {
    this->NameOfTraveler = traveler_.getNameOfTraveler();
}

string Traveler::getNameOfTraveler()const {
    return this->NameOfTraveler;
}

void Traveler::setNameOfTraveler(string NameOfTraveler_) {
    this->NameOfTraveler = NameOfTraveler_;
}
```

1.5 Pager.h

```
#pragma once
class Pager {
private:
    string NumberOfPager;
public:
    Pager() = delete;

    Pager(string NumberOfPager_);

    Pager(Pager& pager_);

    string getNumberOfPager()const;

    void setNumberOfPager(string& NumberOfPager_);
};
```

1.6 Pager.cpp

```
#include<string>
```

```
#include"Pager.h"
using std::string;
Pager::Pager(string NumberOfPager_) {
        this->NumberOfPager = NumberOfPager_;
    }

Pager::Pager(Pager& pager_) {
        this->NumberOfPager = pager_.getNumberOfPager();
    }

string Pager::getNumberOfPager()const {
    return this->NumberOfPager;
}

void Pager::setNumberOfPager(string& NumberOfPager_) {
    this->NumberOfPager = NumberOfPager_;
}
```

1.7 BusinessTraveler.h

```
#pragma once
#include"Pager.h"
#include"Traveler.h"

class BusinessTraveler :public Traveler {
private:
    Pager PagerOfBusinessTraveler{ "4001234567" };
public:
    BusinessTraveler();

    BusinessTraveler(string& NameOfBusinessTraveler_);

    BusinessTraveler(string& NameOfBusinessTraveler_, string&
NumberOfPagerOfBusinessTraveler_);

    BusinessTraveler(BusinessTraveler& businesstraveler_);

    string BusinessTravelerToString()const;

    void setPagerOfBusinessTraveler(const Pager& pager_);

    BusinessTraveler& operator=(const BusinessTraveler& bt_);
};
```

1.8 BusinessTraveler.cpp

```
#include"BusinessTraveler.h"

BusinessTraveler::BusinessTraveler() :Traveler("xiaoming") {};

BusinessTraveler::BusinessTraveler(string&
NameOfBusinessTraveler_) :Traveler(NameOfBusinessTraveler_) {};

BusinessTraveler::BusinessTraveler(string& NameOfBusinessTraveler_, string&
NumberOfPagerOfBusinessTraveler_)
    :Traveler(NameOfBusinessTraveler_) {
    this->PagerOfBusinessTraveler = Pager(NumberOfPagerOfBusinessTraveler_);
}

BusinessTraveler::BusinessTraveler(BusinessTraveler& businesstraveler_)
    :Traveler(businesstraveler_.getNameOfTraveler()) {
    this->PagerOfBusinessTraveler =
Pager(businesstraveler_.PagerOfBusinessTraveler.getNumberOfPager());
}

string BusinessTraveler::BusinessTravelerToString()const {
        return this->PagerOfBusinessTraveler.getNumberOfPager() + "  " +
this->getNameOfTraveler();
}

void BusinessTraveler::setPagerOfBusinessTraveler(const Pager& pager_) {
    this->PagerOfBusinessTraveler = pager_;
}

BusinessTraveler& BusinessTraveler::operator=(const BusinessTraveler& bt_) {
    this->setNameOfTraveler(bt_.getNameOfTraveler());
    this->PagerOfBusinessTraveler =
Pager(bt_.PagerOfBusinessTraveler.getNumberOfPager());
    return *this;
}
```

1.9 midterm1.cpp

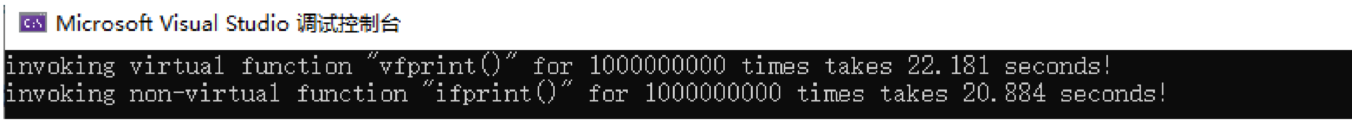
```
#include<iostream>
#include<string>
#include"Traveler.h"
#include"Pager.h"
#include"BusinessTraveler.h"

using std::cout;
using std::cin;
using std::string;

int main(void) {
    //test the function in the class
    string name1{ "lilei" }, number1{ "123456" }, name2{ "hanmeimei" },
number2{ "654321" };
    BusinessTraveler b1{ name1,number1 };//test the constructor that takes two
string arguments
    cout << b1.BusinessTravelerToString() << '\n';
    BusinessTraveler b2{ name2 }; //test the constructor that takes a string
argument
    cout << b2.BusinessTravelerToString() << '\n';
    Pager p2{ number2 };//test the constructor that takes a string argument
    b2.setPagerOfBusinessTraveler(p2);
    cout << b2.BusinessTravelerToString() << '\n';
    b2 = b1; //test the overloading assignment operator function
    cout << b2.BusinessTravelerToString() << '\n';
    return 0;
}
```

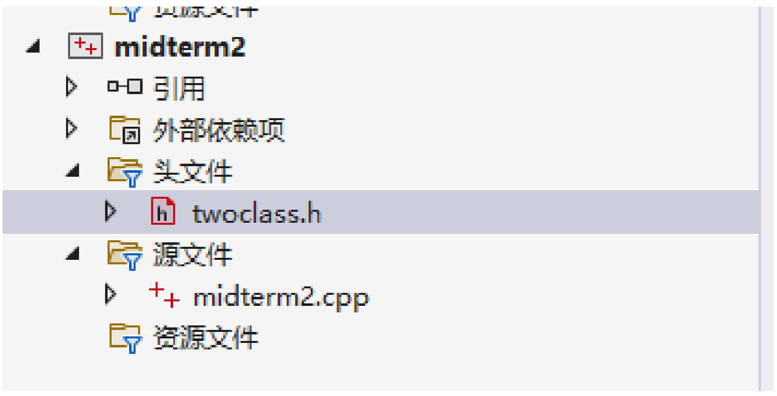
2.作业2

2.1 输出结果截图


Microsoft Visual Studio 调试控制台

```
invoking virtual function "vfprint()" for 1000000000 times takes 22.181 seconds!
invoking non-virtual function "ifprint()" for 1000000000 times takes 20.884 seconds!
```

2.2 工程项目截图



2.3 twoclass.h

```
#pragma once
class father {
private:
public:
    int a{ 0 };
    father() = default;
    virtual void vfprint(int i) {
        a = i;
    }
    void ifprint(int i) {
        a = i;
    }
};

class son :public father {
private:
public:
    int b{ 0 };
    son() = default;
    void vfprint(int i)override {
        b = i;
    }
};
```

2.4 Midterm2.cpp

```
#include<iostream>
#include<ctime>
#include"twoclass.h"
using std::cout;

int main(void) {
    clock_t start1, start2, end1, end2;
    double duration1, duration2;
    son* s1 = new son();
    father* f1=dynamic_cast<father*>(s1);

    start1 = clock();
    for(int i=0;i< 1000000000;i++)
        f1->vfprint(i);
    s1->a = s1->b;
    end1 = clock();

duration1 = (double)(static_cast<double>(end1) -static_cast<double>(start1)) /
CLK_TCK;
    cout << "invoking virtual function \"vfprint()\"
for 1000000000 times takes " << duration1 << " seconds!\n";

    start2 = clock();
    for (int i = 0; i < 1000000000; i++)
        f1->ifprint(i);
    s1->a = s1->b;
    end2 = clock();
    duration2 = (double)(static_cast<double>(end2) -
static_cast<double>(start2)) / CLK_TCK;
    cout << "invoking non-virtual function \"ifprint()\"
for 1000000000 times takes " << duration2 << " seconds!\n";

    return 0;
}
```