Assignment 6

1. What’s the difference between struct and class?
2. What are the constructor and destructor? Their functions? How many different types of constructors and when will they being called?
3. If the constructor is defined as **Private**, what will happen?
4. Find the errors in the following program.

```
class   X
{
private:
    int   a=0, &b;
    const   int   c;
    void   setA(int   i){a=i;}
    X(int   i){a=i;}
public:
    int   X( ){a=b=c=0;}
    X(int   i, int   j, int   k){a=i; b=j; c=k;}
    static   void   setB(int   k){b=k;}
    setC(int   k) const {c=c+k;}
};
int   main( )
{
    X x1;
    X x2(3);
    X x3(1,2,3);
    x1.setA(3);
    return 0;
}
```

5. Read the following program and determine the output.

1)

```
    #include<iostream>
    #include<string>
    using namespace std;

    class X{
        int a;
        char *b;
        float c;
    public:
        X(int x1,char *x2,float x3):a(x1),c(x3){
            b=new char[sizeof(x2)+1];
            strcpy(b,x2);
        }
        X():a(0),b("X::X()"),c(10){ }
        X(int x1,char *x2="X::X(....)",int x3=10):a(x1),b(x2),c(x3){}
        X(const X&other){
            a=other.a;
            b="X::X(const X &other)";
            c=other.c;
        }
        void print(){
            cout<<"a="<<a<<"\t"<<"b="<<b<<"\t"<<"c="<<c<<endl;
        }
    };
    int    main(){
        X *A=new X(4,"X::X(int,char,float)",32);
        X B,C(10),D(B);
        A->print();  B.print();
        C.print();     D.print();
        return 0;
    }
```

2)

```
    #include<iostream>
    using namespace std;
    class Implementation{
    private:
        int value;
    public:
        Implementation(int v) {value = v;}
        void setValue(int v) {value = v;}
        int getValue() const {return value;}
      };
    class Interface{
    private:
        Implementation *ptr;
    public:
        Interface(int);
        void setValue(int);
        int getValue() const;
    };
    Interface::Interface( int v ) : ptr ( new Implementation(v) ) { }
    void Interface::setValue( int v ) { ptr->setValue(v); }
    int Interface::getValue() const { return ptr->getValue(); }
    int main(){
          Interface i(5);
          cout << i.getValue() << endl;
        i.setValue(10);
        cout << i.getValue() << endl;
        return 0;
}
```

3)

```
#include<iostream>
using namespace std;
class A{
    int x;
public:
    A():x(0){cout<<"constructor A() called..."<<endl;}
    A(int i):x(i){cout<<"X"<<x<<"\tconstructor..."<<endl;}
    ~A(){cout<<"X"<<x<<"\tdestructor..."<<endl; }
};
class B{
    int y;
    A X1, X2[3];
public:
    B(int j):X1(j),y(j){cout<<"B"<<j<<"\tconstructor..."<<endl;}
    ~B(){cout<<"B"<<y<<"\tdestructor..."<<endl;}
};
int   main(){
    A X1(1),X2(2);
    B B1(3);
}
```

4)

```
#include <iostream>
#include <cstring>
#include <assert.h>
using namespace std;

class Ctor{
private:
     char *p=nullptr;
public:
     Ctor(char* str=nullptr);
     Ctor(Ctor&& t);
     Ctor& operator=(Ctor&& t);
     Ctor(Ctor& t);
     Ctor& operator=(Ctor& t);
     ~Ctor();
};
```

```
Ctor::Ctor(char* str){
        if(str){
                this->p=new char[strlen(str)+1];
                strcpy(this->p,str);
        }
        cout << "1:Ctor(char *)" << endl;
}
Ctor::Ctor(Ctor&& t):p(move(t.p)){
        t.p=nullptr;
        cout << "2:Ctor(Ctor&&)" << endl;
}
Ctor& Ctor::operator=(Ctor&& t){
        this->p=move(t.p);
        t.p=nullptr;
        cout << "3:=(Ctor&&)" << endl;
        return *this;
}
Ctor::Ctor(Ctor& t){
        this->p=new char[strlen(t.p)+1];
        strcpy(this->p,t.p);
        cout << "4:Ctor(Ctor&)" << endl;
}
Ctor& Ctor::operator=(Ctor &t){
        if(this !=&t){
                delete[] this->p;
                if(t.p){
                        this->p=new char[strlen(t.p)+1];
                        strcpy(this->p,t.p);
                }
        }
        cout << "5:=(Ctor &)" << endl;
        return *this;
}
Ctor::~Ctor(){
        if(this->p){
                delete[] this->p;
                this->p=nullptr;
        }
        cout << "~Ctor" << endl;
}

int main(){
        Ctor c1("ok!"),c2("Hellow");
        Ctor c3(c1);
        c3 = c2;
        c3 = move(c2);
        Ctor c4(move(c1));
        return 0;
    }
```

6. The salary of the employee includes **Wage**, **Subsidy**, **Rent**, **WaterFee** and **ElecFee**. Design a class **Salary** to manage the salary of the employee. The form of the class illustrates as follows:

```
class    Salary{
private:
    double Wage, Subsidy, Rent, WaterFee and ElecFee;
public:
    Salary(……){initialization of each item of the salary};
    Salary(){initialization of each item of the salary to 0};
    void setXX(double f){xx=f;};
    double getXX{return XX;};
    double    RealSalary();              //Calculate the final payment
    ……
};
```

Where member function setXX() is used to set each item of the salary and getXX() is used to get each item of the salary. XX represent data members such as Wage, Subsidy, etc.

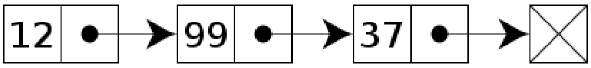
For example, data member Wage's member function is setWage() and getWage().

FinalPayment = Wage + Subsidy – Rent – WaterFee – ElecFee

Complete the design of the class **Salary**, write a main function to testify the member functions.

7. Design the class **Worker** which has name, age, dept, salary and other data members. Salary is the class **Salary** defined above. Complete the design of **Worker** class according to the requirement above, and count the total number of workers by using static data member.
8. Design an integer **List** class which can realize the operation of insert, delete, find, empty, print, etc.

1) Write the list.h file which concludes the declaration of the class.
2) Write the list.cpp file which concludes the definition of the member function.
3) Write a listMain.cpp program to testify your design.



Singly linked list

9. The program simulates the working of ATM:
    1) Create an Account array with 10 objects, where the id is from 0 to 9 and the balance is initialized to 1000 yuan.
    2) Prompt the user to type the valid id(if the id is invalid and then prompt

the user to type again);
Enter an id:

3) After accept the correct id and display the message: "Welcome consumerName!" and then display the following menu
Main Menu
1. Check balance
2. Withdraw
3. Deposit
4. Exit

4) For the above option
1 is used to check the balance and then show the main menu again.
2 is used to withdraw and 3 to deposit and the program should prompt to input the money and then back to the main menu again.
4 is used to exit from the program and display the message: "Thank you and goodbye!" and back to the following state
Main Menu
1. Check balance
2. Withdraw
3. Deposit
4. Exit

✧ Note:
For the first five questions(including program reading), please write your answer in the word document and save it with the name and format as StudentID_Name_6.pdf;
For the last four(6-9) program design, save the source code with the name StudentID_Name_6_1.rar, StudentID_Name_6_2.rar, StudentID_Name_6_3.rar and StudentID_Name_6_4.rar.

✧ Submit your assignment before the end of April 9(Tuesday) to the website!