CSE 108 Practice problems

Namespace, Inheritance and Static class members

1. Create two classes with the name "pool" under two different namespaces - "infrastructure" and "sports".

The pool class under infrastructure namespace should have four private variables and two public functions as follows:

```
private variables:
int height, width, depth;
char painted_color[10];
public functions:
void set_properties(int h, int w, int d, char *pc)
void show() // This function will show the dimention (height x
width x depth), and the underlying painted_color information
```

The pool class under sports namespace should have two private variables and two public functions as follows:

```
private variables:
char table_ingredient[20];
char table_color[10];
public functions:
void set_properties(char *ti, char* tc)
void show() // This function will show the pool table
ingredient and table color
```

Write down the c++ code along with suitable main function. In the main function, just create the objects of each classes and call the member functions of the objects. You can give the hardcoded information into the set_properties functions. For example, `set_properties(1,2,3,"Blue")` can be used for the pool class object of the infrastructure namespace.

2. See the Problem2.cpp file. Write an appropriate definition of the employee class to meet the expected output as given below. Do not modify anything inside the main function, and try to write an efficient code.

```
===Showing details information of employee: A===
Showing personalInformation:
Name: A
Employee Id: 1
Gender: m
Showing Departmental Information:
Department Name: CSE
Assigned work: Office Cleaning
===Showing details information of employee: B===
Showing personalInformation:
Name: B
Employee Id: 2
Gender: f
Showing Departmental Information:
Department Name: EEE
Assigned work: Typing
```

Figure: Expected output of Problem 2

3. See the Problem3.cpp file. Write an appropriate definition of the serverMaintenanceOfficer class to meet the expected output as given below. Do not modify anything inside the main function, and try to write an efficient code. During implementation, you should reuse the definition of employee class that you have written in Problem2.

```
===Showing details information of employee: A===
Showing personalInformation:
Name: A
Employee Id: 1
Gender: m
Showing Departmental Information:
Department Name: CSE
Assigned work: Office Cleaning
===Showing details information of employee: B===
Showing personalInformation:
Name: B
Employee Id: 2
Gender: f
Showing Departmental Information:
Department Name: EEE
Assigned work: Typing
===Showing details information of maintenance officer: C===
Showing personalInformation:
Name: C
Employee Id: 11
Gender: m
Showing Departmental Information:
Department Name: ME
Assigned work: Office Hour Maintenance
Per hour salary: 1200
===Showing details information of maintenance officer: D===
Showing personalInformation:
Name: D
Employee Id: 12
Gender: f
Showing Departmental Information:
Department Name: CE
Assigned work: Off Hour Maintenance
Per hour salary: 1250
```

Figure: Expected output of Problem 3

4. See the Problem4.cpp file. Modify the inventory class and add required code fragments to generate the expected output as given below. Do not modify anything inside the main function, and try to write an efficient code.

```
Parameter-less constructor for inventory class object
Parameter-less constructor for inventory class object
Parameter-less constructor for inventory class object
Available in inventory: 15
Destructor for inventory class object
Destructor for inventory class object
Destructor for inventory class object
```

Figure: Expected output of Problem 4

5. See the Problem5.cpp file. Write an appropriate definition of the buyer class and the seller class to meet the format of the example output as given below. Do not modify anything inside the main function, and try to write an efficient code. During implementation, you should reuse the definition of inventory class that you have written in Problem 4.

```
Parameter-less constructor for inventory class object
Parameter-less constructor for buyer class object
Parameter-less constructor for inventory class object
Parameter-less constructor for buyer class object
Parameter-less constructor for inventory class object
Parameter-less constructor for seller class object
Parameter-less constructor for inventory class object
Parameter-less constructor for seller class object
Enter option 1 to buy, option 2 to sell, and other to exit
Option: 1
Quantity: 10
Person (1 or 2): 1
So far bought quantity: 10
Available in inventory: 10
Option: 1
Quantity: 5
Person (1 or 2): 2
So far bought quantity: 15
Available in inventory: 15
Option: 2
Quantity: 8
Person (1 or 2): 1
So far sold quantity: 8
Available in inventory: 7
Option: 2
Quantity: 4
Person (1 or 2): 2
So far sold quantity: 12
Available in inventory: 3
Option: 5
Destructor for seller class object
Destructor for inventory class object
Destructor for seller class object
Destructor for inventory class object
Destructor for buyer class object
Destructor for inventory class object
Destructor for buyer class object
Destructor for inventory class object
```

Figure: Example output of Problem 5