**EE2310 C++程式設計 HW 4 (Chapter 7) due: 5/9/2019**

**Part 1, Choice(s) 選擇題 (35%)**

1. ADT stands for \_\_\_\_\_\_\_\_ Data Type.

A) Abstract

B) Action

C) Algorithmic

D) Automatic

E) None of the above

A

1. Objects are created from ADTs that encapsulate data and \_\_\_\_\_\_\_\_ together.

A) constants

B) functions

C) memory addresses

D) variables

E) None of the above

B

1. Objects permit data hiding. This means they can keep functions outside the class from \_\_\_\_\_\_\_\_ the data.

A) accessing

B) changing

C) knowing the names of the variables holding

D) deleting

E) doing all of the above

E

1. In OOP terminology, an object's member variables are often called its \_\_\_\_\_\_\_\_, and its member functions are sometimes referred to as its \_\_\_\_\_\_\_\_.

A) values, operators

B) data, activities

C) attributes, activities

D) attributes, methods

E) values, activities

D

1. When the body of a member function is defined inside a class declaration, it is called a(an) \_\_\_\_\_\_\_\_ function.

A) static

B) global

C) inline

D) conditional

E) constructor

C

1. A \_\_\_\_\_\_\_\_ is a member function that is automatically called when a class object is \_\_\_\_\_\_\_\_.

A) constructor, created

B) constructor, destroyed

C) destructor, created

D) destructor, destroyed

E) both A and D

E

1. A constructor may have a return type of \_\_\_\_\_\_\_\_.

A) int

B) bool

C) void

D) Any of the above

E) None of the above

E

1. When a constructor does not require that any arguments be passed to it, it is called a(n) \_\_\_\_\_\_\_\_ constructor.

A) empty

B) default

C) stand-alone

D) zero-element

E) useless

B

1. \_\_\_\_\_\_\_\_ member function may be called by a statement in a function that is outside the class.

A) A declared

B) A public

C) A private

D) An inline

E) Any

B

1. A C++ member function that uses, but does not change, the value of a member variable is called a(n) \_\_\_\_\_\_\_\_.

A) accessor

B) mutator

C) user

D) constant

E) constructor

A

1. Accessors are sometimes called \_\_\_\_\_\_\_\_ functions and mutators are sometimes called \_\_\_\_\_\_\_\_ functions.

A) set, get

B) get, set

C) public, private

D) private, public

E) regular, inline

B

1. If Circle is the name of a class, which of the following statements would create a Circle object named myCircle?

A) myCircle Circle;

B) myCircle Circle();

C) Circle myCircle;

D) Circle myCircle();

E) None of the above

C

1. If setRadius is a Circle class function and myCircle is a Circle object, which of the following statements would set myCircle's radius to 2.5?

A) setRadius(2.5);

B) myCircle.setRadius(2.5);

C) Circle.setRadius(2.5);

D) Circle(setRadius(2.5));

E) None of the above

B

1. When an object is passed \_\_\_\_\_\_\_\_ to a function, its members are not copied.

A) as an argument

B) by value

C) by reference

D) as a constant reference

E) by either method C or D above

E

1. Class declarations are usually stored \_\_\_\_\_\_\_\_.

A) on CDs

B) in their own header files

C) in .cpp files, along with function definitions

D) under pseudonyms

E) in system files

B

1. An object is a(n) \_\_\_\_\_\_\_\_ of a class.

A) example

B) instance

C) copy

D) attribute

E) member

B

1. The bundling of an object's data and procedures together is called \_\_\_\_\_\_\_\_.

A) OOP

B) encapsulation

C) data hiding

D) structuring

E) private access

B

1. If you do not declare an access specification, the default for members of a class is \_\_\_\_\_\_\_\_.

A) inline

B) public

C) private

D) global

E) shared

C

1. Public members of a class object can be accessed from outside the class by using the \_\_\_\_\_\_\_\_.

A) dot operator

B) get function

C) extraction operator

D) member access operator

E) class name

A

1. A C++ member function that sets or changes the value stored in a member variable is called a(n) \_\_\_\_\_\_\_\_.

A) accessor

B) mutator

C) user

D) getter

E) updater

B

1. \_\_\_\_\_\_\_\_ are sometimes called get functions and \_\_\_\_\_\_\_\_ are sometimes called set functions.

A) Accessors, mutators

B) Mutators, accessors

C) Public functions, private functions

D) Private functions, public functions

E) Value-returning functions, void functions

A

1. A constructor must have the same name as the \_\_\_\_\_\_\_\_.

A) first private data member

B) first public data member

C) class

D) first object of the class

E) function return type

C

1. A class may have \_\_\_\_\_\_\_\_ default constructor(s) and \_\_\_\_\_\_\_\_ destructor(s).

A) only one, only one

B) only one, more than one

C) more than one, only one

D) more than one, more than one

E) no, only one

A

1. The \_\_\_\_\_\_\_\_ directive prevents a header file from being included in a program more than once.

A) #include

B) #define

C) #ifndef

D) #endif

E) #exclude

C

1. When a member function is defined outside of the class declaration, the function name must be qualified with the class name, followed by \_\_\_\_\_\_\_\_.

A) a semicolon(;)

B) the scope resolution operator (::)

C) the public access specifier

D) the private access specifier

E) a tilde (~)

B

1. If Square is the name of a class, which of the following statements would create a Square object named box?

A) box Square();

B) box Square;

C) Square box();

D) Square box;

E) None of the above

D

1. If setSide is a Square class function and box is a Square object, which of the following statements would set the length of box's side to 5?

A) setSide(5);

B) box.setSide(5);

C) Square.setSide(5);

D) Square(setSide(5));

E) None of the above

B

1. A class can have a member variable that is an instance of another class. This is called \_\_\_\_\_\_\_\_.

A) object composition

B) object containment

C) chaining

D) encapsulation

E) None of the above

A

1. A structure variable is similar to a class object in which of the following way(s)?

A) It has member data that is usually private and accessed through public member functions.

B) Its data can be initialized with a constructor.

C) It can be passed to a function or returned from a function.

D) All of the above.

E) B and C, but not A.

E

1. The process of object-oriented analysis includes the following key steps \_\_\_\_\_\_\_\_.

A) create a problem description, find all the verbs in the description, then create the classes

B) identify the needed data members and member functions, then assign a class name

C) identify the private and public variables, then prototype the functions and write the code

D) write the main function, then determine which classes it will use

E) identify the needed classes, define their attributes and behaviors, and identify relationships between classes

E

31 Which of the following are legal access to the class or struct members? Assume each is outside of the class member definitions,

struct S class C class D

{ { {

int x; int x; public:

int y; int y; int x;

} private: int y;

S s; int z; private:

}; int z;

C c; };

D d;

A) s.x

B) c.x

C) d.x

D) c.z

E) d.z

A C

1. A constructor
2. can only be used to initialize
3. must initialize all member variables
4. can do anything any other method can do
5. usually initializes all, or most, member variables

CD

1. C++ allows the programmer to deal with data at a higher level of abstraction in part because related data of differing types can be treated as a unit in
2. a array variable
3. a structure variable
4. a function
5. a library
6. a class variable

BE

1. Given the program, which of the following class member accesses are legal?

#include <iostream>using namespace std;class DayOfYear{public: void input(); void output();// other public members private: int month; int day; // other private members

};int main(){

DayOfYear birthDay;

birthDay.input(); // A)

birthDay.day = 25; // B)

cout << birthDay.month; // C)

cout << birthDay.output(); // D)

if(birthDay.month == 1) // E)

cout << "January\n";

}

AD

1. When you defined a C++ class, which of the following should be part of the interface?
2. all declarations of private member variables
3. all declarations for public member functions
4. all explanatory comments for public member declarations.
5. all declarations for private member functions
6. all member function definitions (public or private)

BC

**Part 2, True/False 是非題 (5%)**

1. True/False: Object-oriented programming is centered around objects that include both data and the functions that operate on them.

T

1. True/False: A constructor is a public class function that is automatically invoked (i.e. called) whenever a class object is created.

T

1. True/False: Objects and structure variables can be both passed to functions and returned from functions.

T

1. True/False: A constructor is a public class function that gets called whenever you want to re-initialize an object's member data.

F

1. True/False: When an object is defined without an argument list for its constructor, the compiler automatically calls the object's default constructor, if there is one.

T

**Part 3, Coding 程式題(60%)**

1. **(10%) Car Class**

Write a class named **Car** that has the following member variables:

• year. An int that holds the car’s model year.

• make. A string that holds the make of the car.

• speed. An int that holds the car’s current speed.

In addition, the class should have the following member functions.

• Constructor. The constructor should accept the car’s year and make as arguments and assign these values to the object’s year and make member variables. The constructor should initialize the speed member variable to 0.

• Accessors (“i.e. get function”). Appropriate accessor functions should be created to allow values to be retrieved from an object’s year, make, and speed member variables.

• accelerate. The accelerate function should add 5 to the speed member variable each time it is called.

• brake. The brake function should subtract 5 from the speed member variable each time it is called.

Demonstrate the class in a program that creates a Car object, and then calls the accelerate function five times. After each call to the accelerate function, get the current speed of the car and display it. Then, call the brake function five times. After each call to the brake function, get the current speed of the car and display it.

1. **(10%) Population**

In a population, the birth rate and death rate are calculated as follows:

Birth Rate = Number of Births ÷ Population

Death Rate = Number of Deaths ÷ Population

For example, in a population of 100,000 that has 8,000 births and 6,000 deaths per year,

Birth Rate = 8,000 ÷ 100,000 = 0.08

Death Rate = 6,000 ÷ 100,000 = 0.06

Design a **Population** class that stores a current population, annual number of births, and annual number of deaths for some geographic area. The class should allow these three values to be set in either of two ways: by passing arguments to a three-parameter constructor when a new Population object is created or by calling the setPopulation, setBirths, and setDeaths class member functions. The class should also have getBirthRate and getDeathRate functions that compute and return the birth and death rates. Write a short program that uses the Population class and illustrates its capabilities.

*(Input Validation: If a population figure less than 2 is passed to the class, use a default value of 2. If a birth or death figure less than 0 is passed in, use a default value of 0.)*

1. **(10%) Inventory Class**

Design an Inventory class that can hold information for an item in a retail store’s inventory.

The class should have the following private member variables.

|  |  |
| --- | --- |
| **Variable Name** | **Description** |
| itemNumber | An int that holds the item’s number. |
| quantity | An int that holds the quantity of the item on hand. |
| cost | A double that holds the wholesale per-unit cost of the item |

The class should have the following public member functions.

|  |  |
| --- | --- |
| **Member Function** | **Description** |
| default constructor | Sets all the member variables to 0. |
| constructor #2 | Accepts an item’s number, quantity, and cost as arguments. Calls other class functions to copy these values into the appropriate member variables. Then calls the setTotalCost function. |
| setItemNumber | Accepts an int argument and copies it into the itemNumber member variable |
| setQuantity | Accepts an int argument and copies it into the quantity member variable. |
| setCost | Accepts a double argument and copies it into the cost member variable. |
| getItemNumber | Returns the value in itemNumber. |
| getQuantity | Returns the value in quantity. |
| getCost | Returns the value in cost. |
| getTotalCost | Computes and returns the totalCost. |

Demonstrate the class by writing a simple program that uses it. This program should validate the user inputs to ensure that negative values are not accepted for item number, quantity, or cost.

1. **(10%) Patient Fees**

Write a program that computes a patient’s bill for a hospital stay (including 1. the number of days spent in the hospital and day rate, 2. charges for hospital services (lab tests, surgery charges, etc., and 3. hospital medication charges). The different components of the program are

* The **PatientAccount class** will keep a total of the patient’s charges. It will also keep track of the number of days spent in the hospital. The group must decide on the hospital’s daily rate.
* The **Surgery class** will have stored within it the charges for at least five types of surgery. It can update the charges variable of the PatientAccount class.
* The **Pharmacy class** will have stored within it the price of at least five types of medication. It can update the charges variable of the PatientAccount class.
* The main program.

The main program will design a menu that allows the user to enter a type of surgery, enter one or more types of medication, and check the patient out of the hospital. When the patient checks out, the total charges should be displayed.

1. **(10%) Reverse digits of an integer**

Given a 32-bit signed integer, reverse digits of an integer by using class like bellow:

*Code*

class Solution {

public:

int reverse(int x) {

…

}

};

The input and output of the program might look like this :

**Example 1:**

**Input: 123**

**Output: 321**

**Example 2:**

**Input: -123**

**Output: -321**

**Example 3:**

**Input: 120**

**Output: 21**

**Note:**  
Assume we are dealing with an environment which could only store integers within the 32-bit signed integer range: [−231,  231− 1]. For the purpose of this problem, assume that your function returns 0 when the reversed integer overflows.

1. **(10%) Moving Inchworm**

Write a program that displays an inchworm on the left-hand side of the screen, facing right. Then slowly move him across the screen, until he disappears off the right-hand side. You may wish to do this in a loop so that after disappearing to the right, the worm appears again on the left. The diagram below shows how he may look at various points on the screen.

