單選題，每題5分

1. Which of the following assignments would be a compilation error?

a. Assigning the address of a base-class object to a base-class pointer.

b. Assigning the address of a base-class object to a derived-class pointer.

c. Assigning the address of a derived-class object to a base-class pointer.

d. Assigning the address of a derived-class object to a derived-class pointer.

2. Assuming the following pseudocode for the Fibonacci series, what is the value of the 5th Fibonacci number (*fibonacci (5)*)?

fibonacci(0) = 0

fibonacci(1) = 1

fibonacci(n) = fibonacci(n – 1) + fibonacci(n – 2)

a. 1.

b. 3.

c. 5.

d. 7.

3. Which of the following representations is erroneous?

a.22A b.$(110)_2$ c.$(EF)_{16}$ d.$(141)_8$

4. In which of the following addition problems (using two's complement notation) does an overflow error occur?

a.0011+1010

b.0100+0100

c.1100+1100

d.0101+1000

5. Storing the binary number 10110.100011 in Excess_127 (single precision) representation will be:

a. 0 10000011 01101000110000000000000
b. 0 10000100 10110100011000000000000
c. 0 00000100 01101000110000000000000
d. 0 000001011 0110100011000000000000

6. Which of the following statement is **False**?

a. An array is a random-access structure.
b. A sequential list is a random-access structure.
c. A linked list is a random-access structure.
d. A stack is not a random-access structure.
e. None of the above

7. Recursion is memory-intensive because:

a. Recursive functions tend to declare many local variables.
b. Previous function calls are still open when the function calls itself and the activation records of these previous calls still occupy space on the call stack.
c. Many copies of the function code are created.
d. It requires large data values.

注意：背面有試題

8. Every object of the same class:
   a. Gets a copy of every member function and member variable.
   b. Gets a copy of every member variable.
   c. Gets a copy of every member function.
   d. Shares pointers to all member variables and member functions.

9. Which of the following is not true of object-oriented design?
   a. OOD takes advantage of inheritance relationships.
   b. OOD encapsulates attributes and operations into objects.
   c. OOD focuses on actions (verbs).
   d. Each class can be used to create multiple objects.

10. Polymorphism is implemented via:
   a. Member functions.
   b. virtual functions and dynamic binding.
   c. inline functions.
   d. Non-virtual functions.

11. Abstract classes:
   a. Contain at most one pure virtual function.
   b. Can have objects instantiated from them if the proper permissions are set.
   c. Cannot have abstract derived classes.
   d. Are defined, but the programmer never intends to instantiate any objects from them.

12. A stack is initially empty, then the following commands are performed:

```
push 5
push 7
pop
push 10
push 5
pop
```

Which of the following is the correct stack after those commands (assume the top of the stack is on the *left*)?
   a. 5 10 7 5.
   b. 5 10.
   c. 7 5.
   d. 10 5.

13. Given that the line

```
delete newPtr;
```

just executed, what can you conclude?
   a. The memory referenced by newPtr is released only if it is needed by the system.
   b. The pointer newPtr is of type void *.
   c. The pointer newPtr only exists if there was an error freeing the memory.
   d. The pointer newPtr still exists.

14. Given the class definition:

```
class CreateDestroy
{
public:
    CreateDestroy() { cout << "constructor called, "; }
    ~CreateDestroy() { cout << "destructor called, "; }
```

注意：背面有試題

```
};
```

What will the following program output?

```
int main()
{
    CreateDestroy c1;
    CreateDestroy c2;
    return 0;
}
```

a. constructor called, destructor called, constructor called, destructor called, .

b. constructor called, destructor called, .

c. constructor called, constructor called, .

d. constructor called, constructor called, destructor called, destructor called, .

15. For a non-empty linked list, select the code that should appear in a function that adds a node to the end of the list. newPtr is a pointer to the new node to be added and lastPtr is a pointer to the current last node. Each node contains a pointer nextPtr.

a. lastPtr->nextPtr = newPtr;
   lastPtr = newPtr.

b. lastPtr = newPtr;
   lastPtr->nextPtr = newPtr.

c. newPtr->nextPtr = lastPtr;
   lastPtr = newPtr.

d. lastPtr = newPtr;
   newPtr->nextPtr = lastPtr

16. A queue performs the following commands (in pseudo-code):

enqueue 4, 6, 8, 3, 1

dequeue three elements

enqueue 3, 1, 5, 6

dequeue two elements

What number is now at the front of the queue?

a. *3.* b. *4.* c. *5.* d. *6.*

17. If you add the following nodes to a binary search tree in the order they appear (left-to-right):

6 34 17 19 16 10 23 3

what will be the output of a postorder traversal of the resulting tree?

a. 3 10 16 23 19 17 34 6.

b. 3 6 17 16 10 19 23 34.

c. 6 3 34 17 16 10 19 23.

d. 10 16 23 19 17 34 3 6.

18. What is the code for a loop that iterates from the end of a string toward the beginning?

a.
```
string::reverse_iterator i = s.begin()
while ( i != s.end() )
{
    cout << *i;
    ++i;
}.
```

b.
```
string::reverse_iterator i = s.rbegin()
while ( i != s.rend() )
{
    cout << *i;
    ++i;
}.
```

c.
```
string::reverse_iterator i = s.end()
while ( i != s.begin() )
{
    cout << *i;
    --i;
}.
```

d.
```
string::reverse_iterator i = s.rbegin()
while ( i != s.rend() )
{
    cout << *i;
    --i;
}.
```

19. Class templates:

a. May include the statement template< typename Type > anywhere.

b. Must put template< typename Type > before the class definition.

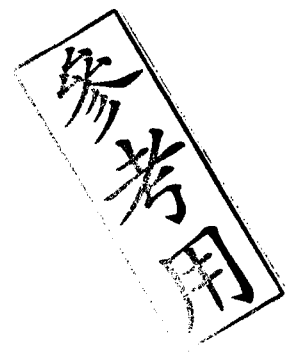
c. Must include template< typename Type > inside the class definition.

d. Have the option of including the optional statement template< typename Type >.

注意：背面有試題

20. What mistakes prevents the following class declaration from functioning properly as an abstract class?

```
class Shape
{
public:
        virtual double print() const;
        double area() const { return base * height; }
private:
        double base;
        double height;
};
```



a. There are no pure virtual functions.
b. There is a non-virtual function.
c. private variables are being accessed by a public function.
d. Nothing, it functions fine as an abstract class.