C++ Programming Final exam. Jan. 19, 2000.

I. (40%) ³ æ Ã (Single choice problem) (Each subproblem 4%)

Ans: 1. b 2. c 3. b 4. d 5. a 6. b 7. b 8. a 9. d 10. a

1. What kind of syntax in the constructor is used to initialize constant members of a class? (a) assignment statement (b) member initializer (c) call by value (d) call by reference.
2. A static data member in a class represents (a) function-wide (b) object-wide (c) class-wide information.
3. If a member initializer is not provided for a member object of a class, the object's (a) copy constructor (b) default constructor (c) conversion constructor is called when instantiating an object of the class.
4. Why does C++ provide standard template library? (a) to support polymorphism (b) to support inheritance (c) to support data abstraction (d) to support program reuse.
5. C++ views a file as (a) a sequence of bytes (b) a sequence of fields (c) a sequence of records.
6. When overloading ( ), [ ], ->, or any assignment operator, the operator overloading function must be declared as a (a) static member (b) class member (c) friend function.
7. An abstract base class contains (a) exactly one (b) one or more (c) at most one pure virtual function(s).
8. Operator << is used for multiple purposes in C++ — as the stream-insertion operator and as the left-shift operator. This is an example of (a) operator overloading (b) operator overriding (c) polymorphism (d) inheritance.
9. Which declaration has any error?
   (a) char *suit[4] = {"Hearts", "Diamonds", "Clubs", "Spades"};
   (b) char str[] = {"I am students"};
   (c) char *str = {"I am students"};
   (d) char (*suit)[4] = {"Hearts", "Diamonds", "Clubs", "Spades"};

10. What is the output generated by the following program? (a) 1 -1 (b) 1 1 (c) –1 1 (d) -1 -1

```
#include <iostream>
using namespace std;
class Test {
public:
   Test( ) { p = new int;}
   void set(int a ) { *p = a;}
   int get( ) {return *p;}
private:
   int *p;
};

int main( ) {
  Test test1, test2;
  test1.set(1);
  cout << test1.get( ) << " ";
  test1=test2;
  test2.set(-1);
  cout << test1.get( ) << endl;
  return 0;
}
```

II. (60%) Â ª Ã D(Brief Explanation)
(Each subproblem10%)

11. When should we define a copy constructor for a class? Explain.

Ans: Copy constructors are invoked whenever a copy of an object is needed such as in call-by-value, when returning an object by value from a called function, or when initializing an object to be copy of another object of the same class. Usually, if a class has a pointer member, we should define copy constructor and overload the assignment operators for safe.

12. An overload of the assignment operator for the class Array has the following code, typically returning the object— *this— by reference. Explain the reason for returning the object by reference.

```
// ptr is a member data of the type int *
// size is a member data of the type int
const Array &Array::operator=( const Array
&right )
{
    if ( &right != this ) {
        if ( size != right.size ) {
            delete [] ptr;
            size = right.size;
            ptr = new int[ size ];
            assert( ptr != 0 );
        }
        for ( int i = 0; i < size; i++ )
            ptr[ i ] = right.ptr[ i ];
    }
    return *this;
}
```

Ans: The use of `this` pointer in statement "return *this;" enable cascading member function calls together. Returning the object by reference is more efficient than returning the object by value.

13. Correct the errors in the following class declaration.

```
// Declaration of the Time class.
#ifndef TIME_H
#define TIME_H
class Time {
public:
    Time( int = 0, int = 0, int = 0 );
    Time( );
    void ~Time( );
    void setTime( int, int, int );
    void printMilitary();
    void printStandard();
private:
    int hour;
    int minute;
    int second;
}
#endif
```

Ans:

1) `Time( );` should be removed or be defined with some other formal arguments that can be distinguished from the default constructor `Time( int = 0, int = 0, int = 0 );` Otherwise there would be a link time error because the compiler cannot distinguish between the following two constructors.
`Time( int = 0, int = 0, int = 0 );`
`Time( );`
2) void `~Time( );` should be `~Time( );`
3) `}` should be `};`

14. Correct the errors in the following program.

```
class Time {
   public:
      void resetx( ) { x=0;}
   private:
      int x;
};
int main( ) {
   Time t1;
   t1 -> resetx( );
   func1(&t1);
}
void func1( Time& ref) {
     ref->resetx( );
}
```

Ans: The correct program should be

```
void func1( Time& ref);
class Time {
   public:
      void resetx( ) { x=0;}
   private:
      int x;
};
int main( ) {
   Time t1;
   t1.resetx( );
   func1(t1);
   return 0;
}
void func1( Time& ref) {
     ref.resetx( );
}
```

15. Explain why a function that accepts a 2-dimenional array as argument can be declared as the following, where the first dimension size of array wDeck is not specified.

```
void shuffle( int wDeck[][13] )
{
     int row, column;

     for ( int card = 1; card <= 52; card++ ) {
          do {
               row = rand() % 4;
               column = rand() % 13;
          } while( wDeck[ row ][ column ] != 0 );

          wDeck[ row ][ column ] = card;
     }
}
```

Ans: It is because C/C++ organizes the memory allocated to 2-dimensional arrays using row-wise manner. That is, the elements are stored in the following order for the array a[3][4].

a[0][0] address low
a[0][1]
a[0][2]
a[0][3]
a[1][0]
a[1][1]
a[1][2]
a[1][3]
a[2][0]
a[2][1]
a[2][2]
a[2][3] address high

In general, the address for a[i][j] can be calculated using the pointer arithmetic as follows if the second dimensional size is known to be row_size.

a+i+j*row_size

16. What is the output for the following program?

```
#include <iostream>
using namespace std;
class Tbase {
public:
Tbase( ) { cout << "Tbase: " << endl; }
virtual ~Tbase( ) { cout << "~Tbase: " <<
endl; }
};

class Tderived : public Tbase {
public:
Tderived( ) { cout << "Tderived: " << endl; }
~Tderived( ) { cout << "~Tderived: " << endl; }
};

int main ( ) {
   Tbase *pTbase=new Tderived( );
   delete pTbase;
   return 0;
}
```

Ans:
Polymorphism makes the destructor call to the destructor of appropriate derived class. And then the base class's destructor is called automatically. So we have the output as:

Tbase:
Tderived:
~Tderived:
~Tbase: