计算机科学系　　2013 学年度第三学期

《　程序设计 II　》期末考试试题(A)：参考答案

任课教师：吴维刚　刘聪　考试形式：　闭卷　考试时间：2 小时

1. Single Choice (20points)

Please choose the choice that best completes the statements in the question

ABBDC　BCDAC　DCAAB　BDADC

2. Output analysis (20points, 5points for each)

Please write down the printout of the following programs.

1) A program about constructor and destructor

constructor is called.
copy constructor is called.
a1=1
S 10
a2=2
S 10

2) A program about inheritance

2
6
4
6

3) A program about polymorphism

Animal Constructor.
Cat Constructor.
Animal Constructor.
Dog Constructor.
I am a Cat
Mew Mew
I am a Dog
Wong Wong
I am a Cat
I am a Dog

4) A program about template and exception

2.5
1.5
0.5
caught: EmptyStackException

3. Error Correction (20points)

In each program, there are 5 errors, i.e. totally 15 errors. You need to find out at least 10 of them, 2 points for each one.

1). Class and object

```
1)  #include <iostream>
2)  using namespace std;
3)
4)   class MyClass
5)   {
6)     public:
7)          MyClass(int x){
8)               member=x;
9)          }
10)         getMember() {                    // int getMember(){
11)              return member;
12)         }
13)         void setMember(int m){
14)              member=m;
15)          }
16)         void ~MyClass(){}                // ~MyClass(){}
17)    private:
18)          int member=0;                   // int member;
19)  };
20)
21)  int main(){
22)      MyClass obj1;                       // MyClass obj1(0);
23)      MyClass obj2(3);
24)      obj1.setMember(1);
25)      cout<<obj2.member<<endl;            // obj2.getMember()
26)      return 0;
27)  }
```

2). Inheritance and polymorphism

```
1)  #include <iostream>
2)  using namespace std;
3)
4)  template<class TYPE>
5)  class BASE
6)  {
7)    public:
8)      void show(TYPE obj){
9)          cout<<obj<<endl;
10)     }
11) }                                                    //};
12)
13) template <class TYPE, class TYPE1>
14) class DERIVED: :public BASE<TYPE1> {          //::->:
15) public:
16)    void show(TYPE obj1, TYPE1 obj2){
17)          cout<<obj1<<endl;
18)          BASE::show(obj2);          //BASE<TYPE1>::show(obj2)
19)    }
20) };
21)
22) int main(){
23)       DERIVED<char *, double> obj;
24)       BASE<char *, double> *pBase=&obj;      // BASE<double>
25)       DERIVED<char *, double> *pDerived=pBase;    //=&obj;
26)       obj.show("Pi is", 3.14);
27)       return 0;
28) }
```

3). A program handling a class for string text

```
1)  #include <iostream>
2)  #include <cstring>
3)  using namespace std;
4)
5)  const int S = 100;
6)
7)  class String{
8)  private:
9)      char array[S];
10)     int size;
```

```
11)  public:
12)          String(){
13)                                                          //size = 0;
14)                  cin >> array;
15)                  while (array[size])   ++size;
16)          }
17)          String(char *t){
18)                  array = t;                              //strcpy(array, t);
19)          }
20)          ~String(){
21)                  delete [] array;                        //delete this statement
22)          }
23)          int getSize(){ return size; }
24)          String &operator += (String str){
25)                  int count = str.getSize();
26)
27)                  if (size + count > S){
28)                          cout << "not enough space.\n";
29                           return *this;
30                   }
31                   for (int i = 0; i < count; i++)
32                           array[size + i] = str.array[count];          //str.array[i];
33
34                   array[size + count] = '\0';
35
36                                                           //return *this;
37           }
38           void show(){
39                   cout << array << endl;
40           }
41   };
42
43   int main(){
44           String str1, str2;
45           str1 += str2;
46           str1.show();
47   }
```

4. Concept explanation (20points, 5 each)

Please explain the following concepts with concrete examples.

1) The meaning of polymorphism.

Polymorphism allows a client to treat different objects in the same way even if they were created from different classes and exhibit different behaviors. You can achieve polymorphism in C++ by function overloading and operator overloading.

2) The meaning of "this" pointer of a class.

- The this pointer is a pointer accessible only within the member functions of a class, struct, or union type. It points to the object for which the member function is called. Static member functions do not have a this pointer.

3) The difference between shallow copy and deep copy

- Shallow copy is simple member wise copy
- Deep copy will allocate new space to be pointed by a pointer member so as to avoid that the pointers of different object point to the same space.

4) The similarity and difference between the string and char array.

- Similarity: both store a sequence of characters.
- Difference: string is a class, which encapsulate various functions to ease the operation of characters sequence. Char array is just a continuous memory space, no predefined functions are available.

5) The meaning of abstract class.

- When you define only function prototype in a base class without implementation and do the complete implementation in derived class. This base class is called abstract class and client won't able to instantiate an object using this base class.

5. Program design (20points)

Please design a matrix class, named Matrix. The elements are stored in two dimensional arrays of size N×N, and each element is of the type int.

You need to provide functions to realize the following functionalities:

- Operator "=", which copies the elements of the matrix at the right to the one at the left;
- Operator "+",to do addition of two matrix;
- Operator "×",to do multiplication of two matrix;
- Output all the elements in the matrix.

An example main function using the Matrix class:

```
int main()
{
    Matrix    m(10), m1(10), m2(10); // 10 is the size of the Matrix
```

```
        for(int i=0; i<10; i++)
            for(int j=0;j<10;j++){
                int x;    cin>>x;
                m.append(i, j, x);    //value assignment
            }
                // …
            m1=m; m2=m;
            m=m1+m2;
            m=m1*m2;
    }
```

```
 class Matrix
{
public:
    Matrix(){}

    Matrix(int **A,int N):a(A),n(N){}

    void display() const;
    Matrix    operator+( const Matrix &m2)const;
    Matrix    operator=( const Matrix &m2)const;
     Matrix    operator*( const Matrix &m2)const;

private:
    int **a,n;
};
void Matrix::display ()const
    {
        for(int k=0;k<n;k++)
        {
            for(int w=0;w<n;w++)
                cout<<setw(3)<<a[k][w];
            cout<<endl;
        }
        cout<<endl;
    }
Matrix Matrix::operator+(const    Matrix &m2)const
{
    int **arr;
    arr=new int *[n];
    for(int k=0;k<n;k++)
            arr[k]=new int[n];
    for(int i=0;i<n;i++)
        for(int j=0;j<n;j++)
```

```
                arr[i][j]=a[i][j]+m2.a[i][j];
        return Matrix (arr,n);
    }
    Matrix    Matrix::operator=(const    Matrix &m2)const
    {
        int **arr;
        arr=new int *[n];
        for(int k=0;k<n;k++)
                arr[k]=new int[n];
        for(int i=0;i<n;i++)
             for(int j=0;j<n;j++)
                  arr[i][j]= m2.a[i][j];
        return Matrix(arr,n);
    }

    Matrix Matrix::operator*(const    Matrix &m2)const
    {    int **arr;
        arr=new int *[n];
        for(int k=0;k<n;k++)
                arr[k]=new int[n];
        for(int i=0;i<n;i++)
             for(int j=0;j<n;j++)
                  arr[i][j]=a[i][j]*m2.a[i][j];
        return Matrix(arr,n);
    }
```