OOP-C++练习题

剩余时间:84天 提前结束考试

A. 单选题 20 | 程序填空题 12 | 函数题 6 | 编程题 15

5-1 阅读下面的程序，完成其中复制构造函数的代码。

```
#include <iostream>
using namespace std;
class CAT
{     public:
           CAT();
           CAT(const CAT&);
          ~CAT();
          int GetAge() const { return *itsAge; }
          void SetAge(int age){ *itsAge=age; }
      protected:
          int* itsAge;
};
CAT::CAT()
{    itsAge=new int;
     *itsAge =5;
}
CAT::CAT(const CAT& c)
{
[          ] (5分);
[          ] (5分);
}
CAT::~CAT()
{     delete itsAge;   }
```

作者 余春艳
单位 福州大学
时间限制 400 ms
内存限制 64 MB

5-2 Run the following program, the output is: B::f()

```
#include <iostream>
using namespace std;
class A{
public:
     [          ] (1分){ cout<<"A::f()\n"; }
 };
class B:public A{
 public:
          void f() {cout<<"B::f()\n"; }
 };
int main()
{
   B b;
   A &p [          ] (1分);
   [          ] (1分)f();
   return 0;
}
```

作者 hulanqing
单位 浙江大学
时间限制 400 ms
内存限制 64 MB

5-3 Run the following program, Enter: 1, the output is:
S1 == S2 HfLLO HFLLO

作者 hulanqing
单位 浙江大学
时间限制 400 ms
内存限制 64 MB

保存

```
#include <iostream>
using namespace std;
class ERROR{};
class STRING
{
   char *m_pStr;
   int m_len;
public:
   STRING(char *str=NULL){
      if (str != NULL) {
             m_len = strlen(str);
             m_pStr = [          ] (1分);
             strcpy([          ] (1分));
      }
      else {
             m_len = 0;
             m_pStr = NULL;
      }
   }
[          ] (1分) operator=(char *str)
{
   [          ] (1分) m_pStr ;
   m_len = strlen(str)+1;
   m_pStr = new char[m_len];
   strcpy([          ] (1分));
   return [          ] (1分);
}

bool operator==(STRING str) [          ] (1分)
{
   return ([          ] (1分)(m_pStr, str.m_pStr)== 0);
}
char operator [] (int i) [          ] (1分)
{
   if (i<m_len && i>=0) return m_pStr[i];
   throw [          ] (1分);
}
char& operator[](int i) [          ] (1分)
{
        if (i<m_len && i>=0) return m_pStr[i];
        ERROR e;
        [          ] (1分);
}
   [          ] (1分) ostream& operator<<(ostream& out ,STRING s);
};
ostream& operator<<(ostream& out ,STRING s)
{
   out << s.m_pStr;
   return out;
}

int main()
{
   STRING s1,s2("HeLLO");
   int i;
   cin >> i;
   s1 = s2;
   if (s1 == s2) cout << "S1 == S2\n";
   s1[1] = s1[1] + 1;
   cout << s1 << endl;;

   [          ] (1分){
       if(s1[i]>='a' && s1[i]<='z') s1[i] =  s1[i] - 32;
           cout << s1 << endl;
   }
   [          ] (1分)(   ERROR& e)
```

保存

```
    return 0;
}
```

5-4

作者 hulanqing
单位 浙江大学
时间限制 400 ms
内存限制 64 MB

```
#include <iostream>
using namespace std;
class IndexError{};
template [          ] (2分)
class ARRAY
{
        size_t m_size;
        T *m_ptr;
public:
        ARRAY(size_t size) : m_size(size)
        {
                m_ptr = new T[size];
                memset(m_ptr, 0, size*sizeof(int));
        }
        ~ARRAY()
        {
                delete[] m_ptr;
        }
        T& at(int index);
};

template <typename T>
[          ] (2分)::at(int index)
{
        if(index<0||[          ] (2分))
        {
                [          ] (2分) IndexError();
        }
        return m_ptr[index];
}

int main()
{
        ARRAY<int> a(50);
        int i;
        cin >> i;
        [          ] (2分)
    {
                for(int j=0;j<i;j++)
                        a.at(i) = j;
        }
        catch(IndexError e)
        {
                return 0;
        }
        return 0;
}
```

5-5 Run the following program, Enter: 1, the output is: 55 34 21 13 8 5 3 2 1 1

作者 hulanqing
单位 浙江大学
时间限制 400 ms
内存限制 64 MB

保存

```
#include <iostream>
using namespace std;

enum ERROR{UnderFlow,OverFlow};
template<typename T>
class StackTemplate {
        enum { ssize = 100 };
        T stack[ssize];
        int top;
public:
        StackTemplate() : top(0) {}
        void push(const T& i) {
                if (top >= ssize) [          ] (1分);
                stack[top++] = i;
        }
        T pop() {
                if ([          ] (1分)) throw UnderFlow;
                return [          ] (1分);
        }
        int size() const
        { return top; }
};
int fibonacci(int n);

int main() {
        [          ] (1分) {
                [          ] (1分) is;
                for(int i = 0; i < 20; i++)
                        is.push(fibonacci(i));
                for(int k = 0; k < 20; k++)
                        cout << is.pop() << "\t";
        }
        catch( ERROR e ) {
                switch([          ] (1分))
                {
                case OverFlow:
                        exit;
                case UnderFlow:
                        exit;
                }
        }
        catch(...)
        {
                exit;
        }
        return 0;
}

int fibonacci(int n)
{
        [          ] (1分) int sz = 100;
        int i;
        static int f[sz];
        if (n >= sz) [          ] (1分);
           f[0] = f[1] = 1;
        for(i = 0; i < sz; i++)
                if(f[i] == 0) break;
        while(i <= n) {
                [          ] (1分) = f[i-1] + f[i-2];
                i++;
        }
        return [          ] (1分);
}
```

5-6 下面的程序定义了Base1类、Base2类和Derived类。
Base1是一个抽象类，其类体中声明了纯虚函数Show。

作者 **李廷元**

保存

Derived类以公有继承方式继承Base1类，以私有继承方式继承Base2类。在Derived类的构造函数的成员初始化列表中调用Base类的构造函数。
请在空白地方填写适当代码，以完成Base1、Base2和Derived类的功能。
此程序的正确输出结果应为：
I’ m a derived class.

内存限制 64 MB

```
#include <iostream>
#include <cstring>
using namespace std;

class Base1 {
public:
  //下列语句需要声明纯虚函数Show
  [          ] (3分);
};

class Base2 {
protected:
  char * _p;
  Base2(const char *s)
  {
    _p = new char[strlen(s) + 1];
    //下列语句将形参指向的字符串常量复制到该类的字符数组中
    [          ] (3分);
  }
  ~Base2() { delete [] _p; }
};

//Derived类公有继承Base1，私有继承Base2类
class Derived : [          ] (3分) {
public:
  //以下构造函数调用Base2类构造函数
  Derived(const char *s) : [          ] (3分)
  { }
  void Show()
  { cout << _p << endl; }
};

int main()
{
  Base1 *pb = new Derived("I'm a derived class.");
  pb->Show();
  delete pb;
  return 0;
}
```

5-7 将空白的地方填写完整，使程序完成指定的功能。

作者 **范鹏程**
单位 内蒙古师范大学
时间限制 400 ms
内存限制 64 MB

```
#include <iostream>
using namespace std;
class Student
 {public:
    [          ] (5分)//利用参数初始化表进行数据初始化
    void display();
  private:
    int num;
    float score;
 };

void Student::display()
 {cout<<num<<" "<<score<<endl;}

int main()
{
  Student stud[5]={
  Student(101,78.5),Student(102,85.5),Student(103,98.5),
  Student(104,100.0),Student(105,95.5)}; //定义对象数组
 [          ] (5分)//定义对象指针指向对象数组
 for([          ] (5分))//显示第1、3、5名学生信息
  p->display();
 return 0;
 }
}
```

保存

单位	内蒙古师范大学
时间限制	400 ms
内存限制	64 MB

```
#include<iostream>
using namespace std;
class R{
    int len,w;
public:
    R(int len,int w);
    int getArea();
};
R::R(int len,int w){
    [ ] (2分)
    [ ] (2分)
}
int R::getArea(){
    return (this->len)*(this->w);
}
int main(){
    R r1(2,5),r2(3,6);
    cout<<"First Area is "<<[ ] (2分)<<endl;
    cout<<"Second Area is "<<[ ] (2分)<<endl;
    return 0;
}
```

输出数据如下：

```
First Area is 10
Second Area is 18
```

5-9 填写程序中的空白，完成指定的功能

作者	**范鹏程**
单位	内蒙古师范大学
时间限制	400 ms
内存限制	64 MB

```
#include<iostream>
using namespace std;
class Point{
    double x,y;
    [ ] (2分)//定义静态变量
public:
    Point(double a=0,double b=0):x(a),y(b){
        [ ] (2分)
    }
    ~Point(){
        [ ] (2分)
    }
    void show(){
        cout<<"the number of Point is "<<[ ] (2分)<<endl;
    }
};
[ ] (2分)
int main(){
    Point p1;
    Point *p=new Point(1,2);
    p->show();
    delete p;
    p1.show();
    return 0;
}
```

程序输出如下：

```
the number of Point is 2
the number of Point is 1
```

5-10 已知平面上的一点由其横纵坐标来标识。本题要求按照已给代码和注释完成一个基本的“点”类的定义（坐标均取整型数值）。并通过主函数中的点类对象完成一些简单操作，分析程序运行结果，将答案写在对应的空格中。

作者	**GONG**
单位	哈尔滨华德学院
时间限制	400 ms

保存

```
#include <                                (1分)>
using namespace std;

class Point
{
                                (1分)//访问权限设置，私有权限
        int x;//横坐标
        int y;//纵坐标
                                (1分)//访问权限设置，公有权限

        //以下为构造函数，用参数a,b分别为横纵坐标进行初始化
                                (2分)(int a,int b)
        {
                                (1分);
                                (1分);
        }

        //以下为拷贝构造函数，借用对象a_point完成初始化
        Point(                  (2分)a_point)
        {
                x=a_point.x;
                y=a_point.y;
        }

        //以下为析构函数
                                (2分)
        {
                cout<<"Deconstructed Point";
                print();
        }

        //以下为输出点的信息的函数，要求在一行中输出点的坐标信息，形如：(横坐标,纵坐标)
        void print()
        {
                cout<<                  (2分)<<endl;
        }
};


int main()
{
        Point b_point(0,0);
        b_point.print();
        int a,b;
                                (2分)//从标准输入流中提取数值给a,b
        Point c_point(a,b);
        c_point.print();
                                (1分)//主函数的返回语句
}
/*设输入为10 10，则本程序的运行结果为：
                                (1分)
                                (1分)
                                (1分)
                                (1分)
*/
```

5-11 根据所定义的基类，定义派生类，请填空完成程序的功能

作者	**范鹏程**
单位	内蒙古师范大学
时间限制	400 ms
内存限制	64 MB

保存

```
#include <iostream>
#include <string>
using namespace std;
class Student                                 //声明基类
 {public:                                     //公用部分
   Student(int n,string nam)                 //基类构造函数
    {num=n;
     name=nam;
    }
   void display()                             //输出基类数据成员
    {cout<<"num:"<<num<<endl<<"name:"<<name<<endl;}
  protected:                                  //保护部分
    int num;
    string name;
};

class Student1: public Student                //用public继承方式声明派生类student
 {public:
```

(2分)

```
                //派生类构造函数
    {age=a;                                   //在此处只对派生类新增的数据成员初始化
     addr=ad;
    }
   void show( )
    {cout<<"This student is:"<<endl;
     display();                               //输出num和name
     cout<<"age: "<<age<<endl;
     cout<<"address: "<<addr<<endl<<endl;
    }

   void show_monitor()                        //输出子对象的数据成员
    {cout<<endl<<"Class monitor is:"<<endl;
     monitor.display();                       //调用基类成员函数
    }
   private:                                   //派生类的私有数据
    Student monitor;                          //定义子对象(班长)
    int age;
    string addr;
  };

int main( )
 {Student1 stud1(10010,"Wang-li",10001,"Li-sun",19,"115 Beijing Road,Shanghai");
  stud1.show( );                              //输出第一个学生的数据
  stud1.show_monitor();                       //输出子对象的数据
  return 0;
 }
```

程序输出如下

```
This student is:
num:10010
name:Wang-li
age: 19
address: 115 Beijing Road,Shanghai


Class monitor is:
num:10001
name:Li-sun
```

5-12 下面程序定义栈类模板StackTemplate，创建栈对象存储斐波那契数列的前10项数值，并以后进先出的方式取出元素并输出，输出结果为：55 34 21 13 8 5 3 2 1 1。其中void push(const T& i)函数为添加元素、T pop()函数为取出栈顶元素，int fibonacci(int n)函数为计算斐波那契数列的第n项值。在计算斐波那契数列值、添加元素和取出元素的过程中要进行上溢（OverFlow）或者下溢（UnderFlow）的异常处理。请补充空白处的代码（每空1分）。

作者	**黄万丽**
单位	曲阜师范大学
时间限制	400 ms
内存限制	64 MB

保存

```
#include <iostream>
using namespace std;

enum ERROR{UnderFlow,OverFlow};

template<typename T>
class StackTemplate {
        enum { ssize = 100 };
        T stack[ssize];
        int top;
   public:
        StackTemplate() : top(0) {}
        void push(const T& i) {
           if (top >= ssize)
                [          ] (1分);
           stack[top++] = i;
        }
        T pop() {
           if ([          ] (1分))
                throw UnderFlow;
           return [          ] (1分);
        }
        int size() const{
            return top;
        }
};
int fibonacci(int n);
int main() {
   [          ] (1分) {
     [          ] (1分)  is;
    for(int i = 0; i < 10; i++)
       is.push(fibonacci(i));
    for(int k = 0; k < 10; k++)
       cout << is.pop() << "\t";
   }
   catch( ERROR e ) {
      switch([          ] (1分) )  {
        case OverFlow:  exit;
        case UnderFlow: exit;
      }
    }
    catch(...) {
    }
   return 0;
 }

int fibonacci(int n) {
     [          ] (1分)  int sz = 100;
    int i;
    static int f[sz];
    if (n >= sz)
         [          ] (1分);
    f[0] = f[1] = 1;
    for(i = 0; i < sz; i++)
        if(f[i] == 0) break;
    while(i <= n) {
         [          ] (1分) = f[i-1] + f[i-2];
        i++;
    }
    return [          ] (1分);
}
```

保存