**四川大学期末考试试题（闭卷）**

**（2022——2023学年第 2 学期） A卷**

课程号：304024030 课序号：08 课程名称：高级语言程序设计-Ⅱ 任课教师：赵启军 成绩：

适用专业年级：计算机学院2022级 学生人数：12 印题份数： 学号： 姓名：

|  |
| --- |
| **考 生 承 诺**  我已认真阅读并知晓《四川大学考场规则》和《四川大学本科学生考试违纪作弊处分规定（修订）》，郑重承诺：  1、已按要求将考试禁止携带的文具用品或与考试有关的物品放置在指定地点；  2、不带手机进入考场；  3、考试期间遵守以上两项规定，若有违规行为，同意按照有关条款接受处理。  **考生签名：** |
| **Ⅰ. 单选题 (20%)**  1. 以下哪项不是面向对象编程的特性？（ ）  A.继承  B.封装  C.多态  D.抽象  2. 以下哪个C++中的STL容器没有下标运算符[]？  A. vector  B. map  C. unordered\_map  D. set  3. 以下关于C++中“auto”的描述中哪一项是正确的？( )  A.“auto”只能用于声明具有已知类型的变量。  B.“auto”只能用于函数返回类型。  C.“auto”允许编译器根据变量的初始值设定项表达式推导变量的类型。  D.“auto”已弃用，不应使用。  4.以下哪个关键字不是表示C++类中成员变量和成员函数的可见性的( ) ？ |

第 1 页，共 11 页

试卷编号：

|  |
| --- |
| A. Public  B. Protected  C. Private  D. Static  5. C++中的“new”和“malloc”有什么区别？（ ）  A.“new”用于为对象分配内存，而“malloc”用于为变量分配内存。  B.“new”用于初始化对象，而“malloc”用于在不初始化的情况下分配内存。  C.“new”和“malloc”是可互换的，可以互换使用。  D.“new”是C++中的关键字，而“malloc”是标准库中的函数。  6. C++中try-catch的目的是什么？( )  A.处理程序执行过程中可能发生的错误  B.发生错误时终止程序  C.将时间信息打印到控制台  D.以上均无  7.如果没有明确定义，编译器将不会创建哪个函数？（ ）  A.构造函数 B.析构函数  C.拷贝构造函数 D.内联函数  8. 声明操作符重载的关键字是？（ ）  A. void  B. int  C. operator  D. double  9.C++中virtual关键字的用途是什么？（ ）  A.它指定函数是纯虚的  B.它指定一个函数可以被一个子类覆盖  C.它指定函数不能被子类重写  D.它指定函数是静态的 |

第 2 页，共 11 页

|  |
| --- |
| 10. 关键字const在C++中的用途是什么？（ ）  A.它指定变量在初始化后不能修改  B.它指定函数不能修改类的任何成员变量  C.它指定函数不能修改传递给它的任何参数  D.以上所有内容  **Ⅱ. 填空题 (10%)**  1.在现代C++中，\_\_\_\_\_关键字用于表示空指针。  2.C++中的\_\_\_\_\_函数是一个非成员函数，可以访问类的私有成员。  3.在C++中，\_\_\_\_\_是一种智能指针，它只能有一个所有者，并且不能被复制。（从shared\_ptr, weak\_ptr和unique\_ptr中选择）  4.在C++中，\_\_\_\_\_是一个与所属类同名的函数，在创建对象时会自动调用。  5. C++中，使用\_\_\_\_\_\_关键字可以将函数定义为模板函数。  **Ⅲ. 程序阅读题 (40%)**  写出以下代码的输出（假设所有所需的头文件已经被声明了）   1. 写出程序输出：   class A {  int i;  public:  A() : i(0) {}  ~A() { cout << get(); }  void set(int i) { this->i = i; }  int get() { return i; }  };  int main(){  A\* p = new A[2];  delete[] p;  return 0; } |

第 3 页，共 11 页

|  |
| --- |
| 1. 写出程序输出：   class A {  int i;  public:  A() : i(0) {}  virtual void set(int i) { this->i = i; }  virtual int get() { return i; }  };    class B : public A {  int i;  public:  B() : i(10) {}  virtual void set(int i) { this->i = i; }  };    int main()  {  B b;  A\* p = &b;  p->set(30);  cout << p->get();  return 0;  }   1. 写出程序输出：   class A {  int i;  public:  virtual void set(int ii) { i = ii; }  virtual int get() { return i; }  };    class B : public A {  int i;  public:  virtual void set(int ii) { i = ii; }  virtual int get() { return i; }  };    int main(){ |

第 4 页，共 11 页

|  |
| --- |
| B a;  B b;  a.set(10);  b.set(20);  A& p = a;  p.set(30);  p = b;  p.set(40);  cout << a.get();  return 0;  }   1. 写出程序输出：     class Base{  public:  int Bar(char x)  {  return (int)(x);  }  virtual int Bar(int x)  {  return (2\*x);  }  };    class Derived : public Base{  public:  int Bar(char x)  {  return (int)(-x);  }  virtual int Bar(int x)  {  return (x/2);  }  };    int main()  {  Derived obj;  Base\* pObj = &obj; |

第 5 页，共 11 页

|  |
| --- |
| cout<<pObj->Bar((char)100)<<endl;  cout<<pObj->Bar(100)<<endl;    return 0;  }   1. 写出程序输出：   class B0 {  public:  virtual void display() {  cout << "B0::display0" << endl;  }  };  class B1 :public B0 {  public:  void display() { cout << "B1::display0" << endl; }  };  class D1 : public B1 {  public:  void display() {  cout << "D1::display0" << endl;  }  };  void fun(B0 ptr) {  ptr.display();  }  int main() {  B0 b0;  B1 b1;  D1 d1;  fun(b0);  fun(b1);  fun(d1);  }   1. 写出程序输出： |

第 6 页，共 11 页

|  |
| --- |
| class MD {  protected:  float miles;  public:  void setDist(float d){miles=d;}  virtual float getDist(){return miles;}  float square(){return getDist()\*getDist();}  };    class FeetDist: public MD {  protected:  float feet;  public:  void setDist(float);  float getDist(){return feet;}  float getMiles(){return miles;}  };    void FeetDist::setDist(float ft) {  feet=ft;  MD::setDist(feet/2);  }    int main() {  FeetDist feet;  feet.setDist(8);  cout<<feet.getDist()<<","<<feet.getMiles()<<","<<feet.square()<<endl;  return 0;  }   1. 写出程序输出：   template <typename T>  class FF{  T a1,a2,a3;  public:  FF(T b1, T b2, T b3):a1(b1),a2(b2),a3(b3)  {}  T Sum() const  {  return a1 + a2 + a3;  }  }; |

第 7 页，共 11 页

|  |
| --- |
| int main()  {  FF<int> x(2,3,4),y(-2,-3,-4);  cout << x.Sum() << "\t" << y.Sum() << endl;  }  Please correct the following programs（point out the errors and correct them. Please state the reasons if necessary (5%)   1. 写出程序输出：   class A {  virtual void f() { cout << "lala"; }  public:  void f(int a, int b=10) {cout << a+b; }  void f(int a, int b=10) const {cout << a-b; }  };    class B : public A {  public:  void f() { cout << "lili"; }  };    int main()  {  B a;  A \* p = &a;  p->f();  return 0;  } |

第 8 页，共 11 页

|  |
| --- |
| **Ⅳ.程序填空题 (20%)**  class Employee  {  public:  Employee( const char \* const, const char \* const );  ~Employee();  const char 【1】\_\_\_\_\_\_\_\_\_\_ const;  const char 【2】\_\_\_\_\_\_\_\_\_\_ const;  static int getCount();  private:  char \*firstName;  char \*lastName;  【3】\_\_\_\_\_\_\_\_\_\_ count;  };  int Employee::count = 0;  int Employee::getCount()  {  return count;  }  Employee::Employee( const char\* const first, const char\* const last )  {  firstName = 【4】\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_;  strcpy( firstName, first );  lastName = 【5】\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_;  strcpy( lastName, last );  count++;  cout << "Employee constructor for " << firstName  << ' ' << lastName << " called." << endl;  }  Employee::~Employee()  {  cout << "~Employee() called for " << firstName  << ' ' << lastName << endl;  delete [] firstName; |

第 9 页，共 11 页

|  |
| --- |
| delete [] lastName;  count--;  }  const char\* Employee::getFirstName() const  {  return firstName;  }  const char\* Employee::getLastName() const  {  return lastName;  }  int main()  {  cout << "Number of employees: "  << 【6】\_\_\_\_\_\_\_\_\_\_ << endl;  Employee\* e1Ptr = 【7】\_\_\_\_\_\_\_\_\_\_;  Employee\* e2Ptr = 【8】\_\_\_\_\_\_\_\_\_\_;  cout << "Number of employees: "  << e1Ptr->getCount();  cout << "\n\nEmployee 1: "  << e1Ptr->getFirstName() << " " << e1Ptr->getLastName()  << "\nEmployee 2: "  << e2Ptr->getFirstName() << " " << e2Ptr->getLastName() << "\n\n";  【9】\_\_\_\_\_\_\_\_\_\_;  e1Ptr = 0;  【10】\_\_\_\_\_\_\_\_\_\_;  e2Ptr = 0;  cout << "Number of employees: "  << Employee::getCount() << endl;  return 0;  }  上述程序的输出:  Number of employees: 0  Employee constructor for Susan Baker called.  Employee constructor for Robert Jones called. |

第 10 页，共 11 页

|  |
| --- |
| Number of employees: 2  Employee 1: Susan Baker  Employee 2: Robert Jones  ~Employee() called for Susan Baker  ~Employee() called for Robert Jones  Number of employees: 0  **Ⅴ. Lambda函数设计 (10%)**  1.编写一个lambda函数，计算以下vector中有多少数字大于常数LIMIT。  int countIf() {  int LIMIT = 5;  // START OF STUDENT CODE  auto lambda = [](){};  // END OF STUDENT CODE  std::vector<int> nums = {3, 5, 10, 6, 8, 9};  return std::count\_if(nums.begin(), nums.end(), lambda);  }  2.写一个lambda函数来找到第一个是奇数并且不能被3整除的数字。  int find() {  // START OF STUDENT CODE  auto lambda = [](){};  // END OF STUDENT CODE  std::vector<int> nums = {3, 9, 18, 12, 5, 7};  std::vector<int>::iterator it = std::find(nums.begin(), nums.end(), lambda);  return \*it;  } |

第 11 页，共 11 页