1. Что такое абстракция? | Выделение общих характеристик объекта, исключая набор незначительных

2. Что называется конструктором? | метод, имя которого совпадает с именем класса и который вызывается автоматически при создании объекта класса

3. Компилятор языка С++: | переводит текст программы в машинные инструкции

4. Если имеется абстрактный класс А и производный от этого класса класс А1 то какая из записей верна? | A1 a1; A& a=a1;

5. Можно ли перегружать оператор разрешения области видимости – "::" | нет

6. Что произойдет, если определение класса будет находиться в файле в двух местах? | ошибка компиляции

7. Отметьте верное утверждение о статических атрибутах класса: | существуют в единственном экземпляре, независимо от количества объектов

8. Пусть заданы классы

class A {... } A1;

class B : public A { ... } B1;

class C : public A { ... } C1;

Что будет выведено при выполнении оператора throw (C1); если обработка исключительной ситуации записана следующим образом:

catch (B& b) { cout << 1; }

catch (C& c) { cout << 2; }

catch (A& a) { cout << 3; }

catch (...)

{ cout << 4; } | 2

9. Что будет выведено в результате выполнения инструкций:

double x = 12.4;

cout << setw(5) << x << setw(3) << setfill('\*') << "" << endl; | " 12.4\*\*\*"

10. Какая из записей является правильной записью абстрактного класса?; | class A { virtual int f() = 0; };

11. Если в классе операция new переопределена как

void\* operator new(size\_t size, int a);

то какой вызов этой операции правильный? | Foo\* ptr = new (20) Foo;

12. Нужно ли учитывать при перегрузке бинарных операций порядок следования операндов? | необходимо учитывать

13. Конструктор копирования запускается, когда: | функция возвращается по значению

14. Совокупность типов формальных параметров, их порядка и имени функции определяет: | сигнатуру (подпись) функции

15. В программе на языке Си++ обязательно имеется функция | main

16. Какой будет результат выполнения следующего кода?

class A {

public:

int inc(int x) {

return ++x;

};

int inc(short x) {

return x + 2;

};

};

A obj;

int y = 5;

cout << obj.inc(y); | 6

17. Какая возможность языка Си++ помогает предупреждать ошибки | контроль типов при компиляции

18. Укажите какой результат будет у следующего примера? int array[10]; array[1] = 2; array[10] = 3; cout << array[10]; | непредсказуемый результат из-за выхода за границы массива

19. Ключевое слово void обозначает что функция | ничего не возвращает

20. Какая функция класса, не являясь его компонентом, имеет доступ к его защищенным и внутренним компонентам? | дружественная

21. Какой вид преобразования типов используется в следующем выражении?

int a = 0;

float f = 3.4;

a += f; | неявное преобразование типов

22. С помощью какой директивы происходит подключение других модулей программы на C++? | #include

23. Что является минимальной областью видимости имен? | блок

24. Членами класса могут быть | как переменные, так и функции, могут быть объявлены как private, public или protected

25. Что из перечисленного может быть аргументом оператора throw? | объект класса

26. Что произойдет при выводе в файл, открытый с помощью ofstream("filename", ios::out|ios::app|ios::trunc) | вывод будет производиться в конец файла

27. Отметьте правильный заголовок шаблона функции: | template void Sum(T x1, T x2);

28. Какое из следующих утверждений об операторе return является верным? | оператор return завершает выполнение функции

29. Сколько производных классов можно получить из базового класса? | неограниченное количество

30. Для чего предназначены манипуляторы потоков ввода-вывода? | для управления форматом ввода-вывода

31. Если в производном классе переопределена операция new то | базовый класс также будет использовать переопределенную операцию

32. Что будет на экране после выполнения следующей программы?

#include <iostream.h>

#include <fstream.h>

#include <string.h>

int main() {

int i = 1, j = 25;

double a = 25e6;

char s[40];

strcpy(s, "Test");

ofstream outfile("c:\\test.dat");

if (!outfile) { cout << "Ошибка создания файла"; return 1;

}

outfile << i << ' ' << j << ' ' << a << ' ' << s << endl;

outfile.close();

} | будет создан текстовый файл "C:\test.dat" с содержимым "1 25 2.5e+07 Test"

33. Какой будет результат следующего выражения?

template <class T> T

sum(T \*a, T \*b)

{

T f = 5;

return (\*a + \*b) - f;

}

int main() {

int i = 10,

j = 20;

double x = 5.1,

y = 2.2;

cout << sum(&i, &j) << " " << sum(&x, &y); } | 25 2.3

34. Существует ли в С++ готовый набор шаблонов: | да, существует специальная библиотека STL

35. Характеристики объекта - это … | свойства

36. Какими по умолчанию объявляются элементы структуры? | public

37. Для того, чтобы использовать reverse\_iterator, необходимо: | инкрементировать его для сдвига назад по контейнеру

38. Что является результатом компоновки программы? | исполняемый файл или библиотека

39. В каком случае описание класса верно?

1. class A {

public: int x;

int summ(int a){

return x+a;}

};

2. class my\_cl {

public: int f;

int summ(int a){

return x+a;

}

};

3. class B {

public: int F;

void Ch\_F(int x) {

F=x;

return F;

}

}; | в первом

40. Что такое cout? | объект типа iostream (std::ostream)

41. Пусть имеется следующий класс:

class Point

{ int x,y;

public:

Point() {

x=0;

y=0;

}

Как называется метод Point(): | Конструктор по умолчанию

42. Для переопределенного оператора верно: | переопределенный оператор выполняется с тем же приоритетом, что и исходный оператор

43. Если имеется код

class A { public: int a; };

A obj;

как обратиться к переменной a? | obj.a

44. Какой будет результат?

int f(int& x) {

static int a = 0;

if (!a) a = ++x;

return a;

}

int y = 6;

f(y);

cout << f(y) << y; | 77

45. После компиляции программы | ее можно выполнять многократно без перекомпиляции

46. Процедуры и функции, объявление которых включено в описание класса, выполняющие действия над объектами класса, – это … | методы класса

47. Если имеется класс с двумя атрибутами

class Complex {

double real;

double img;

. . . }; какой у него должен быть деструктор? | деструктор не обязателен

48. Что произойдет, если определение функции будет находиться в файле в двух местах? | ошибка компиляции

49. Отметьте верное утверждение о статических методах класса: | не могут объявляться со словом const в конце объявления

50. Если заданы классы

class A {... } A1;

class B : public A { ... } B1;

class C : public A { ... } C1;

а обработка исключительной ситуации записана

catch (B& b) { cout << 1; }

catch (C& c) { cout << 2; }

catch (A& a) { cout << 3; }

catch (...)

{ cout << 4; }

то что будет выведено при выполнении оператора throw (A1); | 3

51. Какой из стандартных классов используется для вывода строк на терминал: | ostream

52. В заданной строке определяется имя класса и имя метода класса: void Student::show (void), а именно | имя класса – Student, метода - show

53. Абстрактный класс – это класс, в котором | есть хотя бы один чисто виртуальный метод

54. Произойдет ли ошибка при использовании следующей конструкции?:

class A { const int f()

{ . . .};

};

g(const& A a)

{

a.f(); . . .

} | да, ошибка компиляции

55. Какие операции поддаются перегрузке? | унарные и бинарные

56. Оператор throw без аргументов | повторно вызывает обрабатываемую исключительную ситуацию

57. Отдельные, четко обозначенные экземпляры некоторого класса, – это … | объекты

58. Отметьте верное утверждение: | шаблон может быть членом класса или шаблона класса

59. Какой класс может использоваться в качестве типа атрибута класса? | произвольный класс

60. Если определена операция вычитания для двух объектов класса A, а операция преобразования к int не определена, что будет вызвано при A a1,a2,a3=5; a3 = a1 – a2; | произойдет ошибка

61. При определении метода запись this-> говорит о том, что: | атрибут принадлежит объекту, получившему сообщение

62. Какой правильный вызов функции базового класса из объекта производного класса, если в производном классе эта функция была замещена? | Base::FunctionName();

63. В чем различие использования следующих выражений #include <...> и #include "..." | различие заключается в методе поиска препроцессором включаемого файла

64. Для чего предназначен оператор namespace? | для заключения в группу объявлений классов, переменных и функций в отдельный контекст со своим именем

65. Какие требования предъявляются к классу исключительных ситуаций? | он может быть произвольным классом

66. Что означает cout << setw(3) ? | ширина поля вывода устанавливается равной 3

67. Какой правильный вариант описания шаблона семейства классов? | template <class T> class Array {. . . };

68. Некоторые особые состояния, в которые может попадать объект, – это … | события

69. Укажите правильный доступ к членам класса:

class my { public: double Z;

int f(int c, int d) {

return c+d;

}

char s;

} T1, T2; | T2.f(4,1);

70. В ассоциативном контейнере: | ключи упорядочены

71. Укажите манипулятор потоков среди перечисленных | flush

72. В каких случаях код будет работать правильно

... vector <int> IntVector;

//объект класса вектор

/\*запись значений в IntVector\*/

int d=IntVector[0]; ... } | если оператор [] переопределен соответствующим образом

73. Существует файл "test.dat" в котором записано "Hello World". Каково будет содержимое файла после выполнения кода:

ofstream outfile("c:\\test.dat",ios::in);

if (!outfile) {

cout << "Ошибка создания файла";

return 1; }

outfile << "!!!";

outfile.close(); | !!!lo World

74. Отметьте истинное высказывание: | переменная объявляется, потом изменяется

75. Можно ли в шаблоне класса определить статический метод? | да

76. Вызовет ли данный код ошибку компиляции?

class Rectangle {

public: int a,b;

int sum();

int square();

~rect();

}; | да, имя деструктора должно совпадать с именем класса

77. Сколько блоков catch может быть после блока try? | минимум один

78. Отметьте правильный вариант описания функции шаблона: | template <class T> void change(T \*p1, T \*p2) { . . . };

79. Некоторая часть окружающего нас мира, которая может быть рассмотрена как единое целое, –это … | объект

80. Для того чтобы вывести символ новой строки, надо: | воспользоваться специальным манипулятором endl

81. Если имеется код

class A { public: int a, b, c; }; A obj;

как обратиться к члену класса c? | obj.c

82. Если в классе A определены методы

A(int x);

operator int();

operator++(int);

то какие из них будут вызваны в следующем выражении ?

A b; static\_cast (b + 1); | operator int(), конструктор А(int x)

83. Можно ли создать объект класса, у которого все атрибуты и методы – статические? | да, можно

84. Для того чтобы выполнить чтение из файла с произвольной позиции, надо использовать объект класса | Ifstream

85. Что содержится в записи минимального по своим возможностям класса | не содержится ничего

86. Компоненты, которые видны во время работы во время работы приложения, с ними напрямую может взаимодействовать пользователь, называются: | визуальными

87. Какое из следующих определений представляет собой правильную запись операции сложения целого числа и объекта: | friend A operator+(int a1, const A& a2);

88. При определении класса-шаблона | он должен быть отмечен ключевым словом template

89. В чем разница между фактическими и формальными параметрами? | формальные параметры определены в заголовке функции, а фактические - значения, с которыми функция вызывается

90. Методика разработки программ, в основе которой лежит понятие объекта как некоторой структуры, описывающей объект реального мира, его поведение, – это … | объектно-ориентированное программирование

91. В каком файле заголовков определён объект cout: | iostream.h

92. Сопоставьте:

1. Конструктор –

2. Деструктор –

3. Дружественная функция –

4. Переопределение операций –

A - вызывается автоматически, как только объект класса уничтожается.

B – имеет доступ к защищенным и собственным компонентам класса, не являясь его компонентом.

C – возможность распространения действия стандартных операций на операнды, для которых эти операции первоначально в языке не предполагались.

D – используется для инициализации объектов класса. | 1-D, 2-A, 3-B, 4-C

93. Что происходит при попытке выполнить оператор return внутри блока catch? | выход из функции

94. Что означает cout << flush ? | произвести вывод и очистку буферов

95. Какой правильный вариант создания экземпляра объекта?

template <class T>

class Matrix {

public : Matrix(int, int);

~Matrix() { }

} | Matrix x(4, 5);

96. Имеется запись:

monster - базовый класс, demon - производный:

// Описываются указатели: monster \*p; demon \*d;

При выполнении какого выражения всегда можно говорить, что потери информации не будет? | p=d

97. Переопределение операции сложения приведет к | ее вызову при выполнении операции сложения с объектом класса

98. Блок try catch | заключает участок кода, в котором может сложиться исключительная ситуация

99. Что произойдёт если операция выделения памяти new завершится неудачно? | выделение памяти под объект не произойдёт, и операция new вернёт нулевой указатель или будет сгенерировано исключение

100. В каком порядке происходит вызов деструкторов при уничтожении объекта производного класса? | вызывается деструктор производного класса, затем деструкторы атрибутов производного класса и, потом, деструктор базового класса

101. Именованные категории, позволяющие группировать сходные объекты, – это … | классы

102. Отметьте истинное высказывание для данного примера:

Item::Item() : taken(false), invNumber(0) { } | происходит инициализация атрибутов класса taken и invNumber

103. Отметьте верное утверждение: | шаблон может быть членом класса или шаблона класса

104. Какое правильное объявление виртуальной функции, которая принимает одно целочисленное значение и возвращает void: | virtual void SomeFunction(int);

105. Об ошибке в конструкторе класса может сигнализировать: | исключительная ситуация

106. Имеется код: char a[8]; cin>>a Вводится текст “Hello World”. Что будет в массиве а? | Hello

107. Что будет выведено в результате выполнения кода:

double x=12.4;

cout<<setw(5)<<x<<setw(3)<<setfill(‘\*’)<<””<<endl; | 12.4\*\*\*

108. Какой из классов используется для вывода строк на экран? | ostream

109. Для того, чтобы выполнить чтение из файла с произвольной позиции, надо использовать объект класса | ifstream

110. Что будет выведено при выполнении оператора throw C, если заданы классы

class A {…};

class B: public A {…};

class C: public A {…};

а обработка исключительной ситуации записана

catch (B&b) {cout<<1;}

catch (C&c) {cout<<2;}

catch (A&a) {cout<<3;}

catch(…) {cout<<4;} ? | 2

111. Если в конструкторе класса

class A {

char \*ptr;

public:

A() {ptr=new char [size]; Init();}

!A() {if (ptr) delete [] ptr;}

};

произойдёт исключительная ситуация, будет ли потеряна память при откате по стеку? | будет, если создавалась автоматическая переменная класса A

112. Что называется наследованием? | механизм, посредством которого производный класс получает элементы родительского и может дополнять либо изменять их свойства и методы

113. Дружественная функция – это | функция, объявленная в классе с атрибутом friend, но не являющаяся членом класса

114. Если записано

class A {public:virtual void f() {cout<<1;}};

class B:public A {public:virtual void f() {cout<<2;}};

то что будет напечатано, если

B b; A &a=b; a.f(); ? | 2

115. Что такое диаграмма взаимодействия в UML? | диаграмма, на которой представлено взаимодействие, состоящее из множества объектов и отношений между ними, включая и сообщения, которыми они обмениваются

116. Дан код:

namespace demo {

int i;

void f1(){};

}

Какая конструкция правильно объявляет доступность только метода f1? | using demo::f1();

117. Какая характеристика текста является значимой в UML-диаграммах? | начертание

118. Что понимается под термином "интерфейс"? | абстрактный класс, содержащий только описание

119. Какой описатель соответствует элементам интерфейса? | public abstract

120. Какие элементы могут входить в состав интерфейса? | индексаторы и события

121. Что обозначает запись class A { virtual f() = 0; };? | запись абстрактного класса

122. Выберите правильную синтаксическую конструкцию наследования: | class имя\_класса : список\_базовых\_классов { список\_элементов класса}

123. Какие функции называются чисто виртуальными? | виртуальные функции, к описанию которых добавлен инициализатор =0

124. Для доступа к элементам объекта используются: | при обращении через имя объекта – точка, при обращении через указатель – операция ->, при обращении через имя объекта – два двоеточия, при обращении через указатель – операция ->, при обращении через имя объекта – двоеточие, при обращении через указатель – операция ->

125. Может ли конструктор быть виртуальным? | не может

126. Каков синтаксис функций (или операторов) преобразования объекта одного типа в объект другого типа? | operator <имя нового типа>();

127. Какая ошибка в таком определении шаблона: template <class T, T def\_val> class My {}; | ошибки нет

128. Если существует несколько обработчиков особой ситуации, какой будет вызван? | сработает первый обработчик, перехвативший особую ситуацию

129. Используются ли в UML трехмерные фигуры? | да, на диаграмме развертывания

130. Что обозначает запись typedef void (\*terminate\_handler)();? | описание типа функции обработки ошибок, использующегося при аварийном завершении программы

131. Что собой представляет исключительная ситуация? | неожиданное (нежелательное) состояние, которое возникает во время выполнения программы

132. Как будет работать такой блок:

try

{

...

}

catch (...) { } | будет перехватывать все особые ситуации

133. Что произойдет, если при возникновении особой ситуации блок try не будет обнаружен в данной функции? | поиск try будет продолжен в вызывающей функции и т.д. по стеку

134. Есть ли ошибка в следующем коде?

try

{ }

catch (a) { }

catch (b) { }

catch (...) { }

catch (c) { }

catch (d){ } | оператор catch(...) должен стоять последним

135. Какой атрибут имеют члены класса по умолчанию | private

136. Cколько параметров может принимать catch | один

137. Какая функция вызывается при аварийном завершении программы? | terminate();

138. Что произойдет, если особая ситуация возникнет в конструкторе объекта? | объект создан не будет

139. Где может встречаться блок try-catch? | в любом месте

140. Укажите поток вывода сообщений об ошибках: | cerr

141. Если происходит ошибка при выполнении функции new, то | функция возвращает ноль

142. Какой из модификаторов видимости изображается в UML с помощью символа # (шарп, диез)? | protected

143. Что такое суперкласс? | более общий класс, конкретным воплощением которого является подкласс

144. Имеются следующие обработчики ошибок:

try {}

catch (a) {код 1}

catch (b) {код 2}

catch (с) {код 3}

catch (d) {код 4}

catch (d) {код 5}

Если возникла особая ситуация типа d, какой обработчик будет вызван? | catch (d) {код 4}

145. Ошибки в языке С++ бывают: | синтаксические

146. Если элементы класса объявлены как private, то кому они будут доступны: | будут недоступны ни наследникам класса, ни внешним функциям

147. Какая функция будет первой вызвана, если особая ситуация запущена и не перехвачена? | terminate()

148. Какая функция будет первой вызвана, если функция запустила особую ситуацию, не указанную в ее описании | unexpected()

149. Какая функция вызывается для нормального завершения программы? | exit()

150. В каком файле определяются операторы ввода и вывода? | iostream.h

151. Что выведет следующий код?

сout.width(2);

cout << '(' << "abcdef" << ')'; | (abcdef)

152. Какой класс содержит средства управления вводом и выводом?: | iostream

153. Какая функция задает точность при выводе вещественного числа? | precision(n);

154. Что будет выведено в результате исполнения следующего кода? cout.precision(3); cout << 12.509; | 12.5

155. Какой манипулятор указывает, в какой системе счисления должны выводиться числа? | dec

156. Какая функция устанавливает текущую позицию для чтения из файла? | seekg();

157. В чём ошибка в следующем определении класса?

class Nameable {

virtual sting getName();

} | всё вышеперечисленное

158. В чем заключается недостаток использования функции printf(...) | отсутствие контроля типов передаваемых параметров

159. Объектом какого класса является оператор cout? | ostream

160. Расставьте шаги проектирования классов в наиболее правильном порядке:

1)Определить классы

2)Определить интерфейсы

3)Определить операции для классов

4)Определить связи между классами

5)Определить зависимость от других классов | 1, 5, 2, 4, 3

161. Что понимается под компонентом? | множество связанных между собой классов

162. Если два класса имеют нечто общее, как лучше поступить? | создать для них общий базовый класс

163. Как называются операторы, изменяющие состояние объекта? | модификаторы

164. Как называются операторы, порождающие объект другого типа? | операции преобразований

165. Как называются операторы, позволяющие организовать доступ ко всем частям объекта в строго определенной последовательности? | итераторы

166. В чем заключается самая типичная перестройка иерархии классов? | выделение общей части двух классов в новый класс

167. Какая бывает иерархия? | классов

168. Как правильно ведёт себя конструктор: | конструктор вызывается автоматически при создании объекта

169. Может ли класс, которому принадлежит другой класс, переопределить виртуальные функции этого класса? | Не может

170. Как называется первый язык объектно-ориентированного программирования? | Simula

171. Какой из паттернов относятся к уровню паттернов объектно-ориентированного проектирования? | Посетитель

172. Какая группа паттернов отвечает за организацию сочленения объектов и классов? | структурные паттерны

173. Каковы особенности паттерна «делегат»? | назначение паттерна - передача ответственности на другой класс за выполнение запроса клиенту

174. Какой из порождающих паттернов использует метод клонирования? | прототип

175. Про какой из поражающих паттернов можно сказать: «паттерн уровня объекта, отделяющий конструирование сложного объекта от его реализации»? | строитель

176. Для какого порождающего паттерна инстанцируемые классы определяются динамически? | прототип

177. Какой из структурных паттернов помогает использовать чужой класс, интерфейс которого несовместим с классом, который его должен использовать? | адаптер

178. Какой из структурных паттернов помогает разорвать связь между интерфейсом и реализацией интерфейса? | мост

179. Почему следует использовать метод вместо прямого доступа к полю структуры? | это позволяет изменить представление данных

180. Какой из структурных паттернов помогает организовать единообразное хранение объектов в древовидных структурах? | компоновщик

181. Какой из структурных паттернов помогает динамически добавлять новые обязанности объекту? | декоратор

182. Какой из структурных паттернов описывает разбиение системы на слои? | фасад

183. Какой из структурных паттернов обеспечивает контроль доступа к объекту путём реализации proxy-объекта? | заместитель

184. Какие возможности даёт использование поведенческий паттерн «команда»? | инкапсулировать обмен данными между клиентом и сервером в обмен объектами

185. Какие возможности даёт использование поведенческий паттерн «итератор»? | обеспечивает доступ к содержимому агрегата без раскрытия его представления

186. Какие возможности даёт использование поведенческий паттерн «посредник»? | организовать взаимодействие большого количества объектов между собой

187. Какие возможности даёт использование поведенческий паттерн «наблюдатель»? | построить широковещательные рассылки

188. Какие особенности кода класса являются признаком необходимости использования поведенческих паттернов «состояние», «стратегия»? | однотипные ветвления кода с похожими условиями

189. Структурный паттерн "Адаптер" востребован в ситуации, когда… | Необходимо организовать использование функций определенного бизнес объекта, недоступного для модификации

190. Какими по умолчанию объявляются методы класса? | private

191. Паттерн "Адаптер" обеспечивает… | Взаимодействие несовместимых интерфейсов, путем предоставления единого устойчивого интерфейса для нескольких компонентов

192. Чаще всего паттерн "Адаптер" применяется, если… | Необходимо создать определенный класс, производный от уже существующего

193. В случаях, когда необходимо управлять доступом к объекту, так чтобы создавать громоздкие компоненты только "по требованию" оптимально использовать паттерн… | Заместитель

194. Когда в системе должна аккумулироваться, преобразовываться и удаляться необходимая информация применяется паттерн… | Информационный эксперт

195. Применение паттерна "Компоновщик" особенно востребовано, когда в информационной системе… | Реализованы и поддерживаются древовидные структуры объектов

196. Применение паттерна "Мост" позволяет… | Получить более устойчивые проектные решения для представления элементов абстракции и реализации, упрощая их возможное последующее изменение

197. Когда необходимо обеспечить поддержку множества мелких объектов, инициализированных и используемых в информационной системе применяют структурный паттерн проектирования… | Приспособленец

198. "Приспособленец"– это… | Экземпляр объекта, который выдает себя за группу самостоятельных экземпляров

199. Поведенческие паттерны проектирования определяют… | Общие закономерности связей между объектами, реализующими данные паттерны

200. В поведенческих паттернах, как и в смежных им структурных паттернах, в качестве инструмента определения поведения для различных классов используется… | Наследование

201. У какой переменой в данном коде самое короткое "время жизни"?

char foo(char my\_ch) {

char ch= my\_ch;

static int flag = 1;

if (flag){

char p;

p=ch;

ch=ch+1;

}

..... return ch;

} | p

202. Сколько параметров можно передать в деструктор? | нельзя передать параметры в деструктор

203. Когда требуется, чтобы сложный составной объект, предоставлял доступ к своим элементам, не раскрывая их внутреннюю структуру, применяется паттерн | Абстрактная фабрика

204. В качестве основного назначения паттерна "Итератор", следует выделить… | Предоставляет способ последовательного доступа ко всем элементам составного объекта, не раскрывая его внутреннего представления

205. Когда необходимо послать объекту запрос, не зная о том, выполнение какой операции запрошено, и кто будет получателем целесообразно применять паттерн… | Команда

206. Для реализации паттерна "Наблюдатель" необходимо… | Определить интерфейс "Подписки". Это интерфейс должен быть спроектирован оптимальным образом, не слишком большим, но и не слишком специализированным

207. Основным недостатком паттерна "Посетитель" выделяют то, что… | Затруднено добавление новых классов к системным "элементам", поскольку требуется объявление новой абстрактной операции в классе "Посетитель"

208. В том случае, когда необходимо обеспечить взаимодействие множества объектов, сформировав при этом слабую связанность и избавив объекты от необходимости явно ссылаться друг на друга применяется паттерн… | Посредник

209. В ситуациях, когда требуется варьировать поведение объекта в зависимости от его внутреннего состояния, используют паттерн проектирования… | Состояние

210. В ситуациях, когда класс содержит ряд схожих алгоритмов, как правило, эти алгоритмы приводят к одному и тому же результату, но могут отличаться по другим параметрам. В подобных ситуациях целесообразно использовать паттерн… | Стратегия

211. Когда необходимо зафиксировать поведение объекта для его последующей реализации применяется паттерн… | Хранитель

212. Какое ключевое слово используется в обработке исключительных ситуаций? | catch

213. В случаях, когда требуется эффективно, компактно, надежно реализовать обработку потока информации с потенциально большим количеством обработчиков используется паттерн проектирования… | Контроллер

214. Когда имеются два разных, но в тоже время очень похожих компонента и требуется внести изменения в оба компонента, избежав при этом вредоносного дублирования кода применяется… | Шаблонный метод

215. В условиях, когда система должна отвечать за обработку большого количества входных системных событий целесообразно использовать паттерн… | Контроллер

216. В основные обязанности группы Порождающих паттернов проектирования входит работа по… | Созданию, изменению, управлению объектами информационной системы

217. При реализации Порождающих паттернов широко используется… | Наследование

218. Паттерн, который описывает решение задачи создания объектов путем их инициализации в классе общего назначения называется… | Абстрактная фабрика

219. Основным недостатком паттерна "Одиночка" является… | В некоторых случаях приводит к созданию не масштабируемого приложения

220. Любая современная информационная система не должна зависеть от того, как в ней создаются, компонуются и представляются объекты. Для того, чтобы практически поддержать данный постулат следует создавать новые объекты с помощью паттерна… | Прототип

221. Если требуется отделить конструирование сложного объекта от его представления, таким образом, чтобы в результате одного и того же конструирования могли получаться различные представления используют паттерн… | Строитель

222. Что является основой для реализации механизма интерфейсов в языках программирования? | полиморфизм

223. Что такое инкапсуляция? | Механизм сокрытия внутреннего устройства объектов для контроля за доступом

224. Что понимается под потоком в языке программирования C++? | Механизм ввода-вывода

225. В условиях, когда необходимо определить интерфейс для создания конкретного объекта, но требуется делегировать системным подклассам решение о том, какой класс нужно инстанцировать, используется паттерн… | Фабричный метод

226. Конструкторы используются для | Инициализации созданного объекта

227. Выберите термин, относящийся к полиморфизму | Динамическое связывание

228. Когда подкласс объявляет метод, который имеет тот же тип аргументов, что и метод, объявленный одним из его суперклассов, это называется | Переопределение метода

229. Процесс, с помощью которого один объект может приобретать свойства другого объекта | Наследование

230. Простые процедуры, которые программисты используют в качестве заполнителей при тестировании системы, называются | Заглушки

231. Характеристики объекта, которые отличают его от всех других типов объектов это: | Абстракция

232. ООП позволяет расширять функциональность классов. Это называется | Масштабируемость

233. Какой принцип ООП здесь лишний: | Агрегация

234. Что такое поле/атрибут класса? | Характеристика объекта

235. Операции, выполняемые оператором присваивания и конструктором копирования: | похожи, за исключением того, что конструктор копирования создает новый объект

236. Отношение, при котором объекты одного типа определенным образом связаны с объектами другого типа, это | Ассоциация

237. Отношение, при котором один объект может быть только частью другого объекта. | Композиция

238. Чем отличается композиция от наследования? | При композиции один класс включается в другой как поле, при наследовании один класс является сущностью другого

239. Какая связность классов лучше для поддерживаемости кода? | Слабая

240. Отношение классов к изменению/расширению поведения. Выберите верное утверждение | Нельзя изменять, можно расширять

241. Выберите определение инкапсуляции | это объединение в единое целое свойств и методов с одновременным скрытием реализации

242. Выберите определение полиморфизма | это свойство ООП, при котором одно и то же имя может вызывать различные действия на этапе выполнения

243. Можно ли создать сущность абстрактного класса? | Нет

244. Может ли класс реализовывать несколько интерфейсов? | Да

245. Какой принцип ООП описывает следующее предложение? Этот принцип является способностью использовать общий интерфейс для нескольких форм (типов данных). | Полиморфизм

246. Следует ли объявлять методы открытыми? | Объявляйте методы открытыми только при условии, что они позволяют использовать основные возможности класса

247. Метод определения объектов, при котором производные объекты наследуют свойства от своих предков | Наследование

248. Выберите верное утверждение | Все данные экземпляра класса хранятся в атрибутах этого экземпляра

249. Расшифровка аббревиатуры ООП | Объектно-ориентированное программирование

250. Определение метода | Действия, которые совершает объект

251. Функции, доступные объекту класса - это | Методы

252. Как называется метод, который можно вызывать без создания экземпляра класса? | Статический метод

253. Как называется метод, который не имеет тела? | Абстрактный метод

254. Что представляет собой шаблон объектов? | Класс

255. Какой из принципов гарантирует внутреннее сокрытие реализации одних компонентов от других компонентов? | Инкапсуляция

256. Возможность при описании класса указывать на его происхождение от другого класса это… | Наследование

257. Выберите правильное объявление производного класса | class MoreDetails: public Details;

258. Что такое конкретный экземпляр класса? | Объект

259. Что такое сокрытие реализации класса и отделение его внутреннего представления от внешнего? | Инкапсуляция

260. Имеются выражения “является” и “имеет“. Что они подразумевают в плане принципов ООП? | “является” - наследование, “имеет” - композиция

261. Использование какого принципа ООП обеспечивается динамическим связыванием объектов? | полиморфизм

262. Какие методы (функции) называются перегруженными? | оба вида методов

263. Какой тип связывания происходит при перегрузке метода (функции), а какой при переопределении? | при перегрузке - раннее, при переопределении – позднее

264. Что такое локальная переменная (свойство, поле)? | переменная (свойство, поле), которая определена внутри тела метода (функции) и существует, пока выполняется метод (функция)

265. Какие требования нужно соблюдать при переопределении метода (функции)? | сигнатура метода (функции) должна быть такая же; возвращаемое значение должно быть такое же

266. Выберите верную цепочку наследования | врач-хирург-нейрохирург

267. Статические функции … | Могут быть вызваны без обращения к конкретному объекту класса

268. Допустим, что базовый и производный классы включают в себя методы с одинаковыми именами. Какой из методов будет вызван объектом производного класса, если не использована операция разрешения имени? | тот, который принадлежит порожденному классу

269. Как расшифровывается аббревиатура UML? | Unified Modeling Language

270. Видна ли переменная, объявленная внутри функции, снаружи этой функции? | Нет

271. Видна ли статическая переменная снаружи класса, в котором объявлена? | Зависит и от модификатора переменной, и от модификатора класса

272. Функции, объявленные внутри класса, и выполняющие действие над объектами класса, - это … | методы класса

273. Если метод есть и у базового, и производного классов, то при его вызове у производного класса… | Будет вызван метод производного класса

274. Какой из модификаторов доступа ограничивает доступ больше всего? | private

275. Какой из модификаторов доступа ограничивает доступ меньше всего? | public

276. При использовании какого подхода UML предоставляет максимум преимуществ? | объектно-ориентированное проектирование

277. Основное назначение абстракции | выделение наиболее важных для решения конкретной задачи свойств предмета

278. Основное назначение полиморфизма | дать возможность работать одинаковым образом с разными объектами

279. Почему следует размещать определение метода внутри класса? | Не рекомендуется, поскольку это разглашает излишние подробности реализации.

280. Словом “агрегация” (включение, композиция) точнее всего описывается отношение между… | Вашей комнатой и мебелью в ней

281. Какие механизмы в ОО языках обычно позволяют обеспечить инкапсуляцию объектов? | Модификаторы доступа

282. Драконы умеют летать (как, например, птицы) и ползать (как, например, ящерицы). С точки зрения ООП, примером чего является данная ситуация (выберите наиболее точный вариант)? | Множественное наследование

283. Выберите наиболее подходящее определение класса | Тип, описывающий характеристики и поведение объекта

284. Какая разница между идентичностью (identity) и равенством (equality) объектов в ООП? | Идентичность означает, что две ссылки указывают на один и тот же объект, а равенство - что они содержат одинаковые данные

285. Почему в некоторых языках программирования отказываются от поддержки множественного наследования (имеется в виду наследование реализации)? | Из-за неоднозначности выбора поведения, в случае если суперклассы некоторого класса содержат методы с одинаковыми сигнатурами

286. High Cohesion (сильное сцепление) - это ОО принцип, наиболее ассоциирующийся с ... | тем, что класс спланирован с единственным и конкретным назначением

287. Иерархическое наследование (hierarchical inheritance): | Содержит один базовый класс и несколько производных классов одного и того же базового класса

288. ООП поддерживает два вида связывания объектов с кодом методов. Соответствующие методы называются: | Статическими и виртуальными

289. Термин “наследование” обозначает, что... | Производные классы содержат поля и методы родительского класса

290. Одно из назначений наследования состоит в том, чтобы | добавлять возможности к существующим классам без их модификации

291. Инкапсуляция, как принцип ООП, это? Выберите верное утверждение | механизм, позволяющий ограничить доступ одних компонентов программы к другим

292. Какой механизм в рамках ООП позволяет переиспользовать код из другого класса? Выберите верный вариант | Наследование

293. Какие основные элементы STL инкапсулируют хранение различных значений и объектов? | контейнеры STL

294. Какие основные элементы STL абстрагируют перемещение по коллекциям объектов? | итераторы STL

295. Какие основные элементы STL решают вопросы обработки данных, размещённых в стандартных контейнерах STL? | обобщённые алгоритмы STL

296. Какой из последовательных контейнеров требует самое большое время для доступа к элементам контейнера? | список

297. Какой из последовательных контейнеров обеспечивает наилучшее время вставки элемента? | список

298. Какое утверждение насчёт ассоциативных контейнеров верно? | ассоциативные контейнеры – это контейнеры переменной длины

299. Какое отличие дека от вектора? | быстрая вставка и удаление в начало контейнера

300. По каким причинам для списка не подходят алгоритмы сортировки, реализованные как стандартные алгоритмы | итератор списка не поддерживает всех необходимых для стандартного алгоритма методов

301. Предположим, что класс Derv является производным класса Base. Оба класса содержат метод func() без аргументов. Отметьте правильное выражение, входящее в метод класса Derv, которое вызывает метод func() базового класса. | Base::func();

302. В чем отличие ассоциативных контейнеров от последовательных контейнеров | хранимые данные должны иметь функцию сравнения

303. Каким образом определяется эквивалентность ключей в ассоциативных контейнерах? | используется только оператор меньше и если один ключ не меньше другого и наоборот, то ключи считаются эквивалентны , используется оператор сравнения "<>"

304. Какому классу итераторов эквивалентны указатели С++? | Итераторы произвольного доступa

305. Разыменование какого итератора ведёт к вставке элемента в контейнер? | итератор вставки

306. В чём основное отличие итератора вставки от других итераторов STL? | разыменование оператора вставки ведёт к вставке элемента в контейнер

307. Выберите правильное утверждение | объект в C++ может иметь сколько угодно предков

308. С каким модификатором доступа должны быть описаны члены класса, чтобы к ним имели доступ только объекты самого класса? | private

309. С каким модификатором доступа должны быть описаны члены класса, чтобы к ним имели доступ только объекты самого класса и объекты класса потомка? | protected

310. С каким модификатором доступа должны быть описаны члены класса, чтобы к ним имели доступ любые объекты? | public

311. Когда происходит динамическое связывание? | на этапе выполнения программы

312. Предположим, что класс Derv является производным от класса Base. Мы определяем объект класса Derv, расположенный в функции main(). Через него мы можем получить доступ к: | Членам класса Derv, объявленным как public

313. Какое из следующих утверждений является примером полиморфизма? | выражение, в котором вызывается виртуальная функция

314. Какое из следующих утверждений истинно? | виртуальные функции должны полностью совпадать по типу и числу параметров

315. Какое из приведенных ниже высказываний о наследовании истинно? | объект производного класса – это также и объект родительского класса

316. Выберите правильное утверждение.

class A {};

class B : public А

{}; | класс В является открытым потомком класса А.

317. Какой уровень доступа к функции "setx" в классе "derived" в приведенном примере?

class basex

{

int x;

public:

void setx(int y) {x = y;}

};

class derived : basex {}; | private

318. Какой уровень доступа имеет переменная "x" из класса "derived" в приведенном примере?

class basex

{

protected:

int x;

};

class derived : public basex {}; | protected

319. Какое из следующих утверждений правильное? | дружественность не наследуется

320. Какие элементы модели могут быть уточнены с помощью диаграмм активностей в UML? | любые элементы модели, имеющие динамическое поведение

321. Что понимается под "жизненным циклом программного обеспечения"? | этапы анализа, проектирования, разработки, сопровождения программного продукта, начиная с его первой версии до его последней версии, завершающей существование продукта

322. Для объектно-ориентированного стиля разработки программного обеспечения наиболее подходит модель: | кластерная

323. Предположим, что существует класс Beta, производный от базового класса Alpha. Выберите правильное объявление конструктора производного класса, принимающего один аргумент и передающего его в конструктор базового класса | Beta(int arg):Alpha(arg)

324. Что справедливо для методологии Agile – гибкой разработки программного обеспечения? | она не предполагает жесткой структуры этапов разработки. Ориентирована на постепенную расширяемость строящейся системы и тесную связь разработчиков и заказчиков, в интересах которых строится система

325. Для какой модели разработки программного обеспечения характерна практика парного программирования? | XP

326. Классы в программе могут соответствовать: | существительным в описаниях вариантов использования;

327. Диаграмма классов в языке UML может содержать | отношения

328. Для моделирования поведения системы в языке UML может использоваться диаграмма: | диаграмма последовательности

329. В языке UML диаграмма состояний применяется для описания поведения таких компонентов системы как: | актёр

330. В языке UML интерфейс – это: | совокупность операций, которые определяют сервис (набор услуг), предоставляемый классом или компонентом

331. В языке UML в каком отношении находятся понятия прецедента и кооперации? | реализации

332. В языке UML какие сообщения на диаграммах последовательностей иногда обозначают пунктирной линией? | ответные сообщения

333. Разновидностью какой диаграммы UML являются диаграммы активностей? | диаграммы состояний

334. Что такое полиморфизм? | возможность использования одних и тех же методов для работы с различными объектами базового и порожденного им классов

335. Укажите правильное описание класса Tire, который является производным классов Wheel и Rubber | class Tire: public Wheel, public Rubber

336. Каким образом объекты соотносятся с деятельностями при изображении траектории объекта в языке UML? | с помощью зависимости

337. Укажите правильную запись первой строки описания класса Beta, который является public-производным класса Alpha | class Beta: public Alpha

338. Абстрактный класс используется, когда: | с его помощью запрещено создавать какие-либо объекты

339. Виртуальные функции позволяют: | использовать один и тот же вызов функции для выполнения методов объектов, принадлежащих разным классам

340. Дружественная функция может быть использована для того, чтобы | разрешать доступ к несвязанному классу

341. Принятие решения о том, какая именно функция будет выполняться по конкретному вызову функции, называется | поздним связыванием

342. Продолжительность жизни переменной, которая является | локальной по отношению к методу, совпадает с продолжительностью жизни функции

343. Пусть указатель р ссылается на объекты базового класса и содержит адрес объекта порожденного класса. Пусть в обоих этих классах имеется невиртуальный метод ding(). Тогда выражение p->ding(); поставит на выполнение версию функции ding() из какого класса: | базового

344. Статическая функция | может быть вызвана с использованием имени класса и имени функции

345. Чистая виртуальная функция – это виртуальная функция, которая | делает свой класс абстрактным

346. Что такое класс? | тип данных, определяемый пользователем и сочетающий в себе данные и функции их обработки, называемые методами

347. Методы класса определяют: | какие операции можно выполнять с объектами данного класса

348. Для чего используют закрытые данные? | для внутренней реализации класса

349. Чем класс отличается от структуры? | все члены класса по умолчанию являются закрытыми

350. Аргументы командной строки: | набираются после названия программы в командной строке

351. Для чего нужно определять методы класса открытыми? | Чтобы иметь возможность доступа к закрытым членам класса

352. Когда необходимо создавать конструктор класса? | если необходимо инициализировать класс значениями, отличными от значений по умолчанию

353. Для чего предназначен шаблонный класс? | работает с разными типами данных

354. Какая функция выполняется во второй строке кода:

string str1;

string str2 = str1; | конструктор копирования объекта str2

355. Какие функции вызываются при выполнении этого кода и в каком порядке?

{

string str1;

string str2;

} | конструктор объекта str1, конструктор объекта str2, деструктор объекта str2, деструктор объекта str1

356. Реальный код шаблонной функции генерируется при: | вызове функции в исходном коде

357. Возможность и способ обращения производного класса к элементам базового определяется | ключами доступа private, public, protected в заголовке объявления производного класса

358. Когда запускается деструктор суперкласса? | после вызова деструктора подкласса

359. Что необходимо сделать в конструкторе класса Cat в условиях приведенной ниже иерархии классов?

class Mammal {

public:

Mammal (const string& species\_name);

};

class Cat: public Mammal

{

public:

Cat();

}; | вызвать конструктор Mammal с аргументом cat, воспользовавшись списком инициализации

360. Куда указывает ссылка последнего узла односвязного списка? | null

361. Для записи данных, содержащих переменные типа float, в объект типа ofstream необходимо использовать | write()

362. Исключение в большинстве случаев возникает из-за: | ошибки выполнения

363. Какое из утверждений об уровнях доступа к классу верное? | Подкласс имеет доступ к защищенным и открытым методам и данным родительского класса

364. Когда следует использовать директиву using namespace? | В начале любого файла cpp при отсутствии конфликтов пространств имен

365. Для чего нужны пространства имён? | Чтобы предотвращать конфликты имен в крупных базах кода

366. Исключения передаются: | из выражения, в котором возникла ошибка, в блок-ловушку

367. Для чего следует использовать шаблоны? | Для использования одного и того же кода для разных типов данных

368. Возможность объектов с одинаковой спецификацией иметь различную реализацию, – это: | полиморфизм

369. Когда необходимо использовать тип параметра шаблона? | В шаблонных функциях, при условии, что тип нельзя определить автоматически, и всегда для классов шаблонов

370. Как компилятор определяет, можно ли применить шаблонный параметр к конкретному шаблону? | Он пытается использовать шаблонный параметр: если тип шаблона поддерживает все требуемые операции, то параметр принимается

371. В чём различие размещения в заголовочном файле класса шаблона и обычного класса? | Все методы класса шаблона должны быть определены в заголовочном файле

372. Когда следует делать функцию шаблонной? | Чтобы применить уже реализованный алгоритм, работающий с одним типом данных, к другому типу данных с похожими свойствами

373. Алгоритм STL — это: | независимая функция для работы с контейнерами

374. В какой строке кода допущена ошибка:

class X {

int a;

int f() const;

int g() { return a++;}

int h() const {return a++;}

}; | В пятой

375. Правильно ли перегружены функции? :

class X {

static void f();

void f() const;

}; | Нет

376. Даны следующие три свойства, которыми обладает функция-член класса:

1) Функция имеет право доступа к закрытой части объявления класса;

2) Функция находится в области видимости класса.

3) Функция должна вызываться для объекта класса (имеется указатель this).

Какими из этих свойств обладает дружественная функция? | Первым

377. Дан следующий код: С помощью какого синтаксиса может быть вызвана функция f?

class X {

friend void f(X&);

}; | X x; f(x)

378. Какой правильный заголовок шаблона? | template <class t1, class t2>;

379. Возможность при описании класса указывать на его происхождение от другого класса, – это: | наследование

380. Вектор является подходящим контейнером, если вы: | имеете индекс и хотите получить быстрый доступ к элементу с этим индексом

381. Отметьте правильный вариант описания шаблона семейства функций: | template<class T> void func(T\* p1, T\* p2) {…}

382. Когда уместно использовать вектор? | Когда вы хотите получить доступ к элементам из большого количества отдельных элементов

383. Как одновременно удалить все элементы из словаря? | Вызвать метод clear

384. Когда следует реализовывать собственные структуры данных? | Когда требуется напрямую работать со структурой данных, например, строить дерево выражений

385. Какая из конструкций корректно объявляет итератор, который можно использовать с вектором vector? | vector::iterator itr;

386. Что из нижеперечисленного получает доступ к ключу элемента key, на который в текущий момент указывает итератор словаря itr? | itr->first

387. Как определить, можно ли использовать итератор? | Сравнить его со значением, возвращенным методом end() контейнера, перебор которого выполняется

388. Что из приведённого является компилятором языка C++? | GNU GCC

389. Шаблон функции - это... | определение функции, в которой типу обрабатываемых данных присвоено условное обозначение

390. Возможность скрыть внутреннее устройство объекта от его пользователей, предоставив через интерфейс доступ только к тем членам объекта, с которыми клиенту разрешается работать напрямую, – это: | инкапсуляция

391. Переопределение операций имеет вид: | имя\_класса, ключевое слово operator, символ операции, в круглых скобках могут быть указаны аргументы

392. Алгоритм copy() возвращает итератор на: | элемент, располагающийся после последнего элемента, в который производилось копирование

393. Виртуальными называются функции | функции базового класса, которые могут быть переопределены в производном классе

394. Что целесообразно определять в public разделе класса? | всё, что относится к интерфейсу класса

395. Программа на языке Си++ начинает выполняться с: | функции main

396. Если записано

class A {

public: void f() {

cout << 1;

}

};

class B : public A {

public: void f() {

cout << 2;

}

};

то что будет напечатано в результате выполнения следующего кода?

B b; A& a=b; a.f(); | 1

397. Каким может быть аргумент деструктора? | аргументов у деструктора не бывает

398. Класс B наследован от класса A. Отметьте верное для класса B. | объект класса B может использоваться как объект базового класса

399. Возможно ли использовать механизм исключительных ситуаций в деструкторах? | можно, но обрабатывать их следует внутри деструктора

400. Какая из записей соответствует обращению к атрибуту m\_arg класса AC в определении метода этого же класса? | m\_arg