МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ «ЛУГАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ ВЛАДИМИРА ДАЛЯ»

Факультет компьютерных систем и информационных технологий

Кафедра информационных и управляющих систем

УТВЕРЖДАЮ
Декан факультета компьютерных систем и информационных технологий
_____ Кочевский А.А.
«19» 04 2023 г.

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ
по учебной дисциплине

«Объектно-ориентированное программирование»

09.03.02 Информационные системы и технологии

«Информационные системы и технологии»

Разработчик:
доцент _____ Стоянченко С.С.

ФОС рассмотрен и одобрен на заседании кафедры информационных и управляющих систем от «18» апреля 2023 г., протокол № 15

Заведующий кафедрой
информационных и управляющих систем _____ Горбунов А.И.

Луганск 2023 г.

Паспорт
фонда оценочных средств по учебной дисциплине
«Объектно-ориентированное программирование»

Перечень компетенций (элементов компетенций), формируемых в результате освоения учебной дисциплины

№ п/п	Код контролируемой компетенции	Формулировка контролируемой компетенции	Контролируемые разделы (темы) учебной дисциплины	Этапы формирования (семестр изучения)
1	ПК-03	Способность осуществлять разработку, отладку, проверку работоспособности и безопасности, расчет экономической эффективности информационных систем и технологий, модификацию программного обеспечения.	Тема 1. Теоретические основы ООП. Тема 2. Базовые средства поддержки концепций ООП в языке С++. Тема 3. Расширенные средства поддержки парадигмы ООП в языке С++. Тема 4. Паттерны проектирования. Порождающие паттерны. Тема 5. Паттерны проектирования. Структурные паттерны. Тема 6. Паттерны проектирования. Паттерны поведения.	3,4

Показатели и критерии оценивания компетенций, описание шкал оценивания

№ п/п	Код контролируемой компетенции	Показатель оценивания (знания, умения, навыки)	Контролируемые разделы (темы) учебной дисциплины (модуля), практики[1]	Наименование оценочного средства[2]
1	**2**	**3**	**4**	**5**
1	ПК-03	ПК-03.1. Знать базовые приемы обработки информации, языки программирования, основные процедуры написания и отладки программ, угрозы безопасности	Тема 1. Тема 2. Тема 3. Тема 4. Тема 5. Тема 6.	Лабораторные работы, курсовая работа, промежуточная аттестация (экзамен)

		информационных систем и способы их предотвращения, методы расчета экономической эффективности информационных систем и технологий. ПК-03.1. Уметь обоснованно выбирать средства языка программирования, необходимые для решения поставленных задач, выявлять угрозы безопасности, проводить расчет экономической эффективности информационных систем и технологий. ПК-03.3. Владеть навыками использования современных интегрированных сред разработки для создания программных продуктов, запуска процедуры резервного копирования, применения методов ценообразования для создаваемых информационных продуктов или услуг.		

Фонды оценочных средств по дисциплине «Объектно-ориентированное программирование»

Курсовая работа:

Пример тем курсовой работы.

1. Разработка объектно-ориентированной модели сюжета сказки «Курочка ряба» на основе паттернов проектирования и ее программная реализация.

2. Разработка объектно-ориентированной модели сюжета сказки «Иван царевич и серый волк» на основе паттернов проектирования и ее программная реализация.

3. Разработка объектно-ориентированной модели сюжета сказки «Василиса прекрасная» на основе паттернов проектирования и ее программная реализация.

Критерии и шкала оценивания по оценочному средству «Курсовая работа»

Шкала оценивания (интервал баллов)	Критерий оценивания
5	Курсовая работа выполнена на высоком уровне (правильные ответы даны на 90-100% вопросов/задач)
4	Курсовая работа выполнена на среднем уровне (правильные ответы даны на 75-89% вопросов/задач)
3	Курсовая работа выполнена на низком уровне (правильные ответы даны на 50-74% вопросов/задач)
2	Курсовая работа выполнена на неудовлетворительном уровне (правильные ответы даны менее чем на 50%)

Лабораторные работы:

Пример тем лабораторных работ:

1. Исследование инструментальных средств поддержки ООП в алгоритмическом языке C++. Создание простых классов. Инкапсуляция. Спецификаторы доступа public и private.
2. Исследование особенностей разработки и использования конструкторов класса.
3. Исследование особенностей использования перегружаемых конструкторов. Список инициализации членов класса.
4. Исследование особенностей деструкторов класса.
5. . Исследование особенностей использования классов в языке C++.

Оценочные средства для оценочного средства «защита лабораторных работ»

Лабораторная работа № 1. Исследование инструментальных средств поддержки ООП в алгоритмическом языке C++. Создание простых классов. Инкапсуляция. Спецификаторы доступа public и private

Контрольные вопросы

1. Что такое структура?
2. Как называются переменные, входящие в состав структуры?
3. Что такое объединение?
4. Каковы особенности хранения объединений в памяти?
5. Что такое класс данных?
6. Что входит в состав класса?
7. Чем классы отличаются от структур и объединений?
8. Классы и структуры – что из них поддерживает наследование?
9. Из каких двух частей состоит описание класса в C++?
10. Для чего необходим и когда вызывается конструктор класса?
11. Для чего необходим и когда вызывается деструктор класса?
12. При помощи чего в классах обеспечивается инкапсуляция (сокрытие) внутренней структуры данных?
13. Опишите назначение модификаторов видимости **`private`**, **`protected`** и **`public`**. Общие черты и различия.
14. Что такое наследование и иерархия?
15. Как и для чего используется служебное слово virtual?
16. Что такое абстрактный класс?

17. Почему конструктор не может быть виртуальным, а деструктор почти всегда является виртуальным?

Лабораторная работа № 2. Исследование особенностей разработки и использования конструкторов класса

Контрольные вопросы

1. Укажите назначение конструкторов
2. Какой тип данных возвращает конструктор?
3. Какое имя имеет функция конструктор?
4. Укажите особенности использования аргументов по умолчанию в конструкторах
5. Приведите пример написания конструктора с использованием ключевого слова this.
6. Какие виды конструкторов Вы знаете?
7. Может ли класс не иметь конструкторов?
8. Что такое конструктор «по умолчанию»?

Лабораторная работа № 3. Исследование особенностей использования перегружаемых конструкторов. Список инициализации членов класса.

Контрольные вопросы

1. Укажите особенности перегрузки конструкторов.
2. Опишите особенности конструктора копирования.
3. Опишите особенности списков инициализации. Приведите примеры.

Лабораторная работа № 4 Исследование особенностей деструкторов класса

Контрольные вопросы

1. Укажите особенности деструкторов.
2. Может ли класс иметь несколько деструкторов?
3. Может ли деструктор иметь параметры.
4. Может ли быть деструктор виртуальным?
5. Приведите пример создания деструктора.

Лабораторная работа № 5. Исследование особенностей использования классов в языке C++.

Контрольные вопросы

1. Укажите типы отношений между классами, которые Вам известны
2. Опишите особенности отношения композиция между классами.
3. Опишите особенности отношения агрегации между классами.
4. Опишите особенности отношения ассоциации между классами.
5. Опишите особенности отношения зависимости между классами.

Лабораторная работа № 6. Исследование особенностей использования статических членов класса

Контрольные вопросы

1. Укажите особенности статических полей в классах языка C++.
2. Как инициализируются статические поля в языке C++.
3. Укажите особенности статических методов в классах языка C++.
4. Напишите пример, демонстрирующий использования статических полей.
5. Напишите пример, демонстрирующий использования статических методов.

Лабораторная работа № 7. Исследование особенностей создания и использования паттерна Singleton

Контрольные вопросы

1. Укажите области использование паттерна Singlton.
2. Укажите особенности реализации паттерна Singlton на языке C++.
3. С какой целью используются статические поля в паттерне Singlton?

Лабораторная работа № 8. Исследование особенностей наследования классов. Спецификатор доступа protected

Контрольные вопросы

1. Укажите области использование паттерна Singlton.
2. Укажите особенности реализации паттерна Singlton на языке С++.
3. С какой целью используются статические поля в паттерне Singlton?

Лабораторная работа № 9. Исследование особенностей реализации полиморфизма в объектной модели языка С++. Виртуальные функции

Контрольные вопросы

1. Дайте характеристику принципу полиморфизма в ООП.
2. Опишите особенности механизма виртуальных функций в языке С++.
3. Приведите пример использования виртуальных функции.
4. Объясните работу разработанной в рамках лабораторной работы программы.

Лабораторная работа № 10. Исследование особенностей построения иерархии классов. Абстрактные классы

Контрольные вопросы

1. Какой класс называют абстрактным?
2. Что такое чистая виртуальная функция?
3. Может ли виртуальная функция быть статической?
4. Наследуются ли статические члены класса

Лабораторная работа № 11. Исследование особенностей разработки и использования классов–оболочек

Контрольные вопросы

1. Какой класс называют классом-оболочкой?
2. Перечислите свойства классов-оболочек?
3. Объясните работу разработанной в рамках лабораторной работы программы.

Лабораторная работа № 12. Исследование особенностей обработки исключительных ситуаций в языке программирования С++.

Контрольные вопросы

1. Как создается защищенный блок кода?
2. Как описывается процедура обработки конкретного исключения?
3. Как генерируется исключение?
4. Какие стандартные классы исключений вы знаете? (перечислить не менее трех)
5. Какой класс является базовым для всех стандартных классов исключений?
6. Как можно ограничить список исключений, которые могут генерироваться в функции?

Лабораторная работа № 13. Исследование особенностей аппарата дружественные функции в языке программирования С++.

Контрольные вопросы

1. Что такое конструктор копирования и для чего он нужен?
2. Что является входным параметром конструктора копирования? Почему входной параметр должен описываться со служебным словом **`const`**
3. Какие функции называются дружественными? Как описываются?
4. Для чего необходимы дружественные функции?

Лабораторная работа № 14. Исследование особенностей аппарата дружественных функции-членов класса

Контрольные вопросы

1. Что такое конструктор копирования и для чего он нужен?
2. Что является входным параметром конструктора копирования? Почему входной параметр должен описываться со служебным словом **`const`**
3. Какие функции называются дружественными? Как описываются?
4. Для чего необходимы дружественные методы-функции?

Лабораторная работа № 15. Исследование особенностей механизма дружественных классов в языке С++.

Контрольные вопросы

1. Что такое конструктор копирования и для чего он нужен?
2. Что является входным параметром конструктора копирования? Почему входной параметр должен описываться со служебным словом **`const`**
3. Какие функции называются дружественными? Как описываются?
4. Для чего необходимы дружественные классы?

Лабораторная работа № 16. Исследование особенностей механизма вложенных классов в языке программирования С++.

Контрольные вопросы

1. Что такое конструктор копирования и для чего он нужен?
2. Что является входным параметром конструктора копирования? Почему входной параметр должен описываться со служебным словом **`const`**
3. Какие функции называются дружественными? Как описываются?
4. Для чего необходимы дружественные классы?

Лабораторная работа № 17. Исследование особенностей обобщенного программирования в языке С++. Шаблоны методов класса

Контрольные вопросы

1. Опишите назначение обобщенного программирования.
2. Опишите структуру описания шаблона функции.
3. Укажите ограничения на использование переменных типов в шаблонных функциях.
4. Дайте характеристику конкретизации функции.

Лабораторная работа № 18. Исследование особенностей шаблонных классов в языке программирования С++.

Контрольные вопросы

1. Опишите назначение обобщенного программирования.
2. Опишите структуру описания шаблона класса.
3. Укажите ограничения на использование переменных типов в шаблонных классах.
4. Дайте характеристику конкретизации класса.

Лабораторная работа № 19. Исследование особенностей перегрузки операторов в языке программирования С++

Контрольные вопросы

1. В чем заключается перегрузка операторов?
2. Какие операторы нельзя перегружать?
3. Можно ли создать свой собственный оператор?
4. В каких случаях перегрузка оператора может осуществляться только при помощи дружественной функции? Приведите примеры.
5. Почему при перегрузке операторов ввода из потока и вывода в поток необходимо возвращать ссылку на поток?

Лабораторная работа № 20. Исследование особенностей механизма определения пространства имен в языке программирования С++. Его связь с именами классов

Контрольные вопросы

1. Дайте характеристику аппарата пространства имен в языке программирования С++.
2. Опишите особенности описания имен в теле класса.
3. Опишите особенности программного обеспечения, разработанного в рамках данной лабораторной работы.

Критерии и шкала оценивания по оценочному средству «Лабораторная работа»

Шкала оценивания (интервал баллов)[2]	Критерий оценивания
5	Лабораторная работа выполнена самостоятельно на высоком уровне и в полном объеме, отчет оформлен в соответствии с

	требованиями, сделаны правильные выводы по проведенным экспериментам.
4	Лабораторная работа выполнена самостоятельно на среднем уровне и в полном объеме, отчет оформлен с незначительными отклонениями от требований, допущены незначительные неточности в выводах по проведенным экспериментам
3	Лабораторная работа выполнена на низком уровне и не полностью, отчет оформлен с отклонениями от требований, выводы по экспериментам сделаны не в полном объеме.
2	Лабораторная работа не выполнена, отчет не оформлен, или представленный отчет не соответствует варианту задания.

4.

Оценочные средства для промежуточной аттестации (экзамен)

1. Что определяет класс? Чем обличается класс от объекта?
2. Можно ли объявлять массив объектов? А массив классов?
3. Разрешается ли объявлять указатель на объект? А указатель на класс?
4. Допускается ли передавать объекты в качестве параметров, и какими способами? А возвращать как результат?
5. Как называется использование объекта одного класса в качестве поля другого класса?
6. Является ли структура классом? Чем класс отличается от структуры?
7. Какие ключевые слова в C++ обозначают класс?
8. Объясните принцип инкапсуляции.
9. Что такое композиция?
10. Для чего используются ключевые слова public и private?
11. Можно ли использовать ключевые слова public и private в структуре?
12. Существуют ли ограничения на использование public и private в классе? А в структуре?
13. Обязательно ли делать поля класса приватными?
14. Что такое метод? Как вызывается метод?
15. Может ли метод быть приватный?
16. Как определить метод непосредственно внутри класса? А вне класса? Чем эти определения отличаются?
17. Можно в методах присваивать параметрам значения по умолчанию?
18. Что обозначается ключевым словом this?
19. Зачем нужны константные методы? Чем отличается определение константного метода от обычного?
20. Может ли константный метод вызываться для объектов-переменных? А обычный метод — для объектов-констант?
21. Объясните принцип полиморфизма.
22. Сколько места в памяти занимает объект класса? Как это узнать?
23. Каков размер «пустого» объекта?
24. Влияют ли методы на размер объекта?
25. Одинаков ли размер класса и аналогичной структуры?
26. Какие операции нельзя перегружать? Как вы думаете, почему?
27. Можно ли перегружать операции для встроенных типов данных?
28. Можно ли при перегрузке изменить приоритет операции?
29. Можно ли определить новую операцию?

30. Перечислите особенности перегрузки операций как методов класса. Чем отличается перегрузка внешним образом от перегрузки как метода класса?

31. Какой результат должны возвращать операции с присваиванием?

32. Как различаются перегруженная префиксная и постфиксная операции инкремента и декремента?

33. Что означает выражение *this? В каких случаях оно используется?

34. Какие операции не рекомендуется перегружать как методы класса? Почему?

35. Какие операции разрешается перегружать только как методы класса?

36. Дайте определение дружественной функции. Как объявляется дружественная функция? А как определяется?

37. Дайте определение конструктора. Каково назначение конструктора? Перечислите отличия конструктора от метода.

38. Сколько конструкторов может быть в классе? Допускается ли перегрузка конструкторов? Какие виды конструкторов создаются по умолчанию?

39. Может ли конструктор быть приватным? Какие последствия влечет за собой объявление конструктора приватным?

40. Приведите несколько случаев, когда конструктор вызывается неявно.

41. Как проинициализировать динамическую переменную?

42. Как объявить константу в классе? Можно ли объявить дробную константу?

43. Каким образом разрешается инициализировать константные поля в классе?

44. В каком порядке инициализируются поля в классе? Совпадает ли этот порядок с порядком перечисления инициализаторов в списке инициализации конструктора?

45. Какие конструкции C++ разрешается использовать в списке инициализации качестве инициализирующих выражений?

46. Какой вид конструктора фактически является конструктором преобразования типов?

47. Для чего нужны функции преобразования? Как объявить такую функцию в классе?

48. Как запретить неявное преобразование типа, выполняемое конструктором инициализации?

51. Влияет ли наличие целочисленных констант-полей на размер класса?

52. Разрешается ли объявлять массив в качестве поля класса. Как присвоить элементам массива начальные значения?

53. Сколько операндов имеет операция индексирования []? Какой вид результата должна возвращать эта операция?

54. Для чего нужны статические поля в классе? Как они определяются?

55. Как объявить в классе и проинициализировать статический константный массив?

56. Что такое выравнивание и от чего оно зависит? Влияет ли выравнивание на размер класса?

57. Дайте определение контейнера.

58. Какие виды встроенных контейнеров в C++ вы знаете?

59. Какие виды доступа к элементам контейнера вам известны?

60. Чем отличается прямой доступ от ассоциативного?

61. Перечислите операции, которые обычно реализуются для последовательного доступа к элементам контейнера.

62. Дайте определение итератора.

63. Можно ли реализовать последовательный доступ без итератора? В чем преимущества реализации последовательного доступа с помощью итератора?

64. Что играет роль итератора для массивов C++?

65. Что такое деструктор? Может ли деструктор иметь параметры?

66. Почему для классов-контейнеров деструктор надо писать явным образом?

67. Допускается ли перегрузка деструкторов?

68. Что такое «глубокое копирование» и когда в нем возникает необходимость?

69. Какое копирование осуществляет стандартный конструктор копирования?

70. Чем отличается копирование от присваивания?

71. Объясните, почему в операции присваивания требуется проверка присваивания самому себе?

72. Можно ли в качестве операции индексирования использовать операцию вызова функции ()? В чем ее преимущества перед операцией []?

73. Почему необходимо писать два определения операции индексирования? Чем они отличаются?

74. Дайте определение вложенного класса.

75. Можно ли класс-итератор реализовать как внешний класс? А как вложенный? В чем отличия этих методов реализации?

76. Может ли объемлющий класс иметь неограниченный доступ к элементам вложенного класса? А вложенный класс — к элементам объемлющего?

77. Ограничена ли глубина вложенности классов?

78. Можно ли определить вложенный класс внешним образом? Зачем это может понадобиться?

79. Каким образом вложенный класс может использовать методы объемлющего класса? А объемлющий — методы вложенного?

80. Что такое «запредельный» элемент, какую роль он играет в контейнерах?

81. Объясните, по каким причинам трудно написать универсальный контейнер, элементы которого могут иметь произвольный тип.

82. Назовите ключевые слова C++, которые используются для обработки исключений.

83. Исключение — это:

1) событие;

2) ситуация;

3) объект;

4) ошибка в программе;

5) прерывание;

84. Каким образом исключение генерируется?

85. Каковы функции контролируемого блока?

86. Что обозначается ключевым словом catch?

1) контролируемый блок;

2) блок обработки исключения;

3) секция-ловушка;

4) генератор исключения;

5) обработчик прерывания;

87. Какого типа может быть исключение?

88. Сколько параметров разрешается писать в заголовке секции-ловушки?

89. Какими способами разрешается передавать исключение в блок обработки?

90. Объясните, каким образом преодолеть ограничение на передачу единственного параметра в блок обработки.

91. Почему нельзя выполнять преобразования типов исключений при передаче в секцию-ловушку?

92. Напишите конструкцию, которая позволяет перехватить любое исключение.

93. Могут ли контролируемые блоки быть вложенными?

94. Зачем нужен «контролируемый блок-функция» и чем он отличается от обычного контролируемого блока?

95. Перечислите возможные способы выхода из блока обработки.

96. Каким образом исключение «передать дальше»?

97. Сколько секций-ловушек должно быть задано в контролируемом блоке?

98. Что такое «спецификация исключений»?

99. Что происходит, если функция нарушает спецификацию исключений?

100. Учитывается ли спецификация исключений при перегрузке функций?

101. Что такое «иерархия исключений»?

102. Существуют ли стандартные исключения? Назовите два-три типа стандартных исключений.

103. Поясните «взаимоотношение» исключений и деструкторов.

104. Объясните, зачем может понадобиться подмена стандартных функций завершения.

105. Какие виды нестандартных исключений вы знаете?

106. В чем отличие механизма структурной обработки исключений Windows от стандартного механизма?

107. Какие две роли выполняет наследование?

108. Какие виды наследования возможны в C++?

109. Чем отличается модификатор доступа protected от модификаторов private и public?

110. Чем открытое наследование отличается от закрытого и защищенного?

111. Какие функции не наследуются?

112. Сформулируйте правила написания конструкторов в производном классе.

113. Каков порядок вызова конструкторов? А деструкторов?

114. Можно ли в производном классе объявлять новые поля? А методы?

115. Если имя нового поля совпадает с именем унаследованного, то каким образом разрешить конфликт имен?

116. Что происходит, если имя метода-наследника совпадает с именем базового метода?

117. Сформулируйте принцип подстановки.

118. Когда выполняется понижающее приведение типов?

119. Объясните, что такое «срезка» или «расщепление».

120. Объясните, зачем нужны виртуальные функции.

121. Что такое связывание?

122. Чем «раннее» связывание отличается от «позднего»?

123. Какие два вида полиморфизма реализованы в C++?

124. Дайте определение полиморфного класса.

125. Может ли виртуальная функция быть дружественной функцией класса?

126. Наследуются ли виртуальные функции?

127. Каковы особенности вызова виртуальных функций в конструкторах и деструкторах?

128. Можно ли сделать виртуальной перегруженную операцию, например, сложение?

129. Может ли конструктор быть виртуальным? А деструктор?

130. Как виртуальные функции влияют на размер класса?

131. Как объявляется «чистая» виртуальная функция?

132. Дайте определение абстрактного класса.

133. Наследуются ли чистые виртуальные функции?

134. Можно ли объявить деструктор чисто виртуальным?

135. Чем отличается чистый виртуальный деструктор от чистой виртуальной функции?

136. Зачем требуется определение чистого виртуального деструктора?

137. Наследуется ли определение чистой виртуальной функции?

138. Приведите классификацию целей наследования.

139. Объясните разницу наследования интерфейса от наследования реализации.

140. Назовите причины, требующие разделения программ на части.

141. Дайте определение термина «единица трансляции»?

142. Чем отличается файл с исходным текстом от единицы трансляции?

143. Существуют ли в C++ конструкции, позволяющие идентифицировать отдельный модуль?

144. Какие способы сборки программы вы можете назвать?

145. Что такое «объектный модуль»? Программа, которая «собирает» объектные модули в программу, называется _____ ?

146. В чем заключается отличие аргумента «файл» от <файл> в директиве #include?

147. Что такое ODR?

148. Объясните, что такое «страж» включения и зачем он нужен.

149. Является ли интерфейс класса его определением?

150. Сколько определений класса может быть в единице трансляции?

151. Сколько определений класса может быть в многофайловой программе?

152. Чем отличаются стандартные заголовки <string>, <string.h> и <cstring>?

153. Объясните суть идиомы Pimpl.

154. Что такое делегирование и как его можно использовать для повышения степени инкапсуляции?

155. Каким образом глобальную переменную, определенную в одной единице трансляции, сделать доступной в другой единице трансляции? А константу?

156. Можно ли использовать слово extern при объявлении функций?

157. Как локализовать объявление функции в файле?

158. Чем отличается «внешнее» связывание от «внутреннего» связывания?

159. Что такое «спецификации компоновки»?

160. Какие объекты обладают внутренним связыванием по умолчанию?

161. Какие области видимости имен вы знаете?

162. Для чего используются пространства имен?

163. Чем отличаются именованные и неименованные пространства имен?

164. Могут ли пространства имен быть вложенными?

165. Для чего применяются алиасы пространства имен?

166. Как сделать члены пространства имен доступными в нескольких (в пределе — во всех) файлах программного проекта?

167. Объясните разницу между статической и динамической инициализацией.

168. В чем состоит проблема инициализации глобальных статических переменных?

169. Какие элементы класса можно объявлять статическими?

170. Можно ли объявить в классе статическую константу? А константный статический массив?

171. А какие статические поля можно инициализировать непосредственно в классе?

172. Как определяются статические поля? В какой момент работы программы выполняется инициализация статических полей?

173. Сколько места в классе занимают статические поля ?

174. Чем отличается статический метод от обычного?

175. Какие методы класса не могут быть статическими?

176. Какие применения статических полей вы можете привести? А каким образом применяются статические методы?

177. Приведите структуру и принцип действия паттерна Singleton.

178. Для чего предназначены шаблоны?

179. Какие виды шаблонов в C++ вы знаете?

180. Объясните термин «инстанцирование шаблона».

181. В чем разница между определением и объявлением шаблона?

182. Объясните назначение ключевого слова typename.

183. Какие виды параметров разрешается задавать в шаблоне класса? А в шаблоне функции?

184. Можно ли параметрам шаблона присваивать значения по умолчанию?

185. Может ли параметром шаблона быть другой шаблон? Каковы особенности объявления параметра-шаблона?

186. Что такое специализация шаблона? Объясните разницу между полной и частичной специализацией.

187. Разрешается ли специализировать шаблон функции?

188. Может ли класс-шаблон быть вложенным в другой класс-шаблон? А в обычный класс?

189. Можно ли объявить в классе шаблонный метод? А шаблонный конструктор?

190. Можно ли перегружать функцию-шаблон?

2. Практические вопросы:

1. Составить концептуальное и объектно-ориентированное описание предметной области задачи моделирования ремонта квартиры. В зависимости от состояния квартиры требуется циклевка пола, побелка потолка, оклейка стен обоями и т.п.
2. Составить объектно-ориентированное описание предметной области задачи моделирования мушкетерского поединка. Исход зависит от мастерства и состояния дуэлянтов, а также от окружающих условий и степени важности причины дуэли.
3. Определить временную переменную, поместить в нее массив чисел, вводимый с помощью суфлера. Присоединить к нему массив чисел, считанных из файла. Используя обобщенные сообщения итерации, вывести в окно системной информации и в файл массив проверки четности элементов полученного массива. Вывести в то же окно класс элементов нового массива.
4. Определить глобальную переменную, поместить в нее пустой массив заданного размера. Заполнить массив строками, считанными из файла, затем дополнить его с помощью суфлера. Используя обобщенные сообщения итерации, вывести в окно системной информации и в файл в столбик каждый элемент массива, взятый в обратном порядке (в верхнем регистре). Вывести туда же массив размеров заданных строк.
5. Определить глобальную переменную, поместить в нее пустой массив заданного размера. Заполнить массив строками, вводимыми с помощью суфлера. Используя обобщенные сообщения итерации, вывести в окно системной информации в столбик каждый элемент массива, взятый в обратном порядке (в верхнем регистре). Вывести туда же массив размеров заданных строк.
6. Составить объектно-ориентированное описание предметной области задачи моделирования сражения гнома с эльфом с использованием волшебных шаров и колец, дающих невидимость и летучесть.
7. Составить концептуальное и объектно-ориентированное описание предметной области задачи подбора и покупки в магазине монитора для домашнего компьютера. Выбор производится в зависимости от типа компьютера, стоимости, а также потребностей и возможностей покупателя.

Критерии и шкала оценивания по оценочному средству промежуточная аттестация (экзамен)

Шкала оценивания	Характеристика знания предмета и ответов
Отлично (5)	Студент глубоко и в полном объеме владеет программным материалом. Грамотно, исчерпывающе и логично его излагает в устной или письменной форме. При этом знает рекомендованную литературу, проявляет творческий подход в ответах на вопросы и правильно обосновывает принятые

	решения, хорошо владеет умениями и навыками при выполнении практических задач.
Хорошо (4)	Студент знает программный материал, грамотно и по сути излагает его в устной или письменной форме, допуская незначительные неточности в утверждениях, трактовках, определениях и категориях или незначительное количество ошибок. При этом владеет необходимыми умениями и навыками при выполнении практических задач.
Удовлетворительно (3)	Студент знает только основной программный материал, допускает неточности, недостаточно четкие формулировки, непоследовательность в ответах, излагаемых в устной или письменной форме. При этом недостаточно владеет умениями и навыками при выполнении практических задач. Допускает до 30% ошибок в излагаемых ответах.
Неудовлетворительно (2)	Студент не знает значительной части программного материала. При этом допускает принципиальные ошибки в доказательствах, в трактовке понятий и категорий, проявляет низкую культуру знаний, не владеет основными умениями и навыками при выполнении практических задач. Студент отказывается от ответов на дополнительные вопросы.

Оценочные средства для итоговой аттестации (экзамен)

1. Какие параметры функции-шаблона выводятся автоматически?
2. Может ли шаблон класса быть наследником обычного класса? А обычный класс от шаблона?
3. Объясните, что такое класс свойств (класс трактовок).
4. Каким образом можно использовать возможность наследования обычного класса от шаблона?
5. Может ли шаблонный конструктор быть конструктором по умолчанию?
6. Для чего применяются директивы явного инстанцирования?
7. Объясните, в чем состоят проблемы, возникающие при разделении шаблонного класса на интерфейс и реализацию?
8. Что такое «модель явного инстанцирования» и как она работает?
9. Может ли шаблонный класс иметь «друзей»?
10. Какие проблемы возникают при объявлении дружественной функции для класса-шаблона?
11. Разрешается ли определять в классе-шаблоне статические поля? А статические методы?
12. Что такое «инициализация нулем»?
13. Что является единицей памяти в C++? Какие требования к размеру единицы памяти прописаны в стандарте C++?
14. В каких единицах выдает результат операция sizeof? Какие типы данных имеют размер 1?
15. Какие три вида памяти входят в модель памяти C++?
16. Сколько видов динамической памяти обеспечивает C++?
17. Какие функции для работы с динамической памятью достались C++ по наследству от C? В какую библиотеку они включены?

18. Какие функции выделяют память, и с помощью каких функций память освобождается?
19. Какое важное отличие имеет функция calloc() от функции malloc()?
20. Какие действия выполняют функции выделения памяти, если память не может быть выделена?
21. Зависит ли объем выделенной памяти от типа указателя? Влияет ли выравнивание на объем выделяемой динамической памяти?
22. Можно ли с помощью функции realloc() уменьшить объем выделенной памяти?
23. Что произойдет, если функции free() передать в качестве аргумента нулевой указатель?
24. В чем главное отличие объектно-ориентированного механизма new/delete от механизма malloc()/free()?
25. Сколько существует форм new/delete? В чем их отличие?
26. Какие типы являются POD-типами? Чем отличается работа механизма new/delete с POD-объектами и nonPOD-объектами?
27. Какие функции выполняет обработчик new?
28. Можно ли реализовать собственный обработчик new и «прицепить» его к механизму new/delete?
29. В чем главное отличие объединения от других видов классов C++?
30. Может ли объединение участвовать в иерархии наследования?
31. Разрешается ли определять для объединения конструкторы и деструктор? А виртуальные функции?
32. В чем похожи и чем отличаются объединение и размещающий new?
33. Объясните, почему при использовании размещающего new нужно явным образом вызывать деструктор?
34. Зачем нужны интеллектуальные указатели?
35. Что такое «стратегия владения»? Сколько стратегий владения вы знаете?
36. Какой интеллектуальный указатель реализован в стандартной библиотеке STL, и какую стратегию владения он реализует?
37. Объясните, в чем преимущества и недостатки интеллектуальных указателей со счетчиком ссылок.
38. Разрешается ли перегружать new и delete и какими способами?
39. Опишите схему функции, перегружающей глобальную функцию new.
40. Отличается ли реализация перегруженной функции new[]() для массивов от реализации «обычной» функции new()?
41. Как вы думаете, почему функции new/delete, перегружаемые для класса, являются статическими?
42. Зачем при перегрузке new/delete для класса нужно проверять размер запрашиваемой памяти?
43. Объясните, чем определяется «динамичность» контейнеров?
44. Что такое «стратегия распределения памяти», и какие стратегии выделения памяти вы знаете?
45. Рассмотрите следующую стратегию распределения памяти: память выделяется для нескольких элементов блоками фиксированной длины, но блоки связываются в список. Для какого вида контейнера можно использовать такую стратегию?
46. Какие операции можно перегрузить для доступа к элементам двумерного массива?
47. В чем заключаются сложности использования операции индексирования [] для доступа к элементам двумерного массива?
48. Каковы способы реализации операций с контейнерами?
49. Какую конструкцию можно назвать «обобщенный алгоритм»?

50. Каким образом объявить указатель на метод?
51. Объясните разницу между указателем на функцию и указателем на метод.
52. Каким образом получить адрес метода?
53. Можно ли указателю на функцию присваивать адрес метода?
54. Какие операции определены в C++ для косвенного вызова метода через указатель?
55. Что такое «функтор»? Приведите пример функционального класса.
56. Какими способами функтор вызывается?
57. Можно ли использовать наследование при разработке функторов?
58. Разрешается ли операцию вызова функции () определять как виртуальный метод? А как статический?
59. В чем преимущества функторов перед указателями на функции?
60. Объясните, зачем нужны адаптеры функторов? Какие виды адаптеров вы знаете?
61. Как используются классы свойств при разработке функторов?
62. Объясните, что такое «композиция» и приведите примеры?
63. Объясните, чем отличается множественное наследование от простого?
64. Приведите структуру и принцип действия паттерна Adapter.
65. Сформулируйте основную проблему множественного наследования.
66. Выполняется ли принцип подстановки при открытом множественном наследовании?
67. Что такое виртуальное наследование? Каковы его преимущества и недостатки по сравнению с обычным наследованием?
68. Может ли виртуальное наследование быть одиночным?
69. Влияет ли виртуальное наследование на размер класса?
70. Объясните, каким образом с помощью виртуального наследования можно вообще запретить наследование.
71. Какие средства C++ составляют RTTI?
72. Объясните разницу между повышающим, понижающим и перекрестным приведением.
73. Какими свойствами должен обладать класс, чтобы с ним работал механизм RTTI?
74. В чем приведение указателей отличается от приведения ссылок?
75. Какие исключения связаны с механизмом RTTI?
76. Что такое «поток» — дайте определение.
77. Как классифицируются потоки, реализованые в библиотеках ввода/вывода C++?
78. Что такое буферизация и зачем она нужна?
79. Какие библиотеки ввода/вывода реализованы в C++ и чем они отличаются?
80. Перечислите стандартные потоки и объясните их назначение.
81. Зачем нужен процесс форматирования и когда он выполняется?
82. Что такое «форматная строка», и в каких функциях она используется?
83. Объясните назначение элементов спецификатора формата.
84. Сколько спецификаторов формата может быть в форматной строке?
85. Какой из элементов спецификатора формата не является умалчиваемым?
86. Перечислите несколько известных вам обозначений типов в спецификаторе формата, и укажите их назначение.
87. Сколько модификаторов типа вы знаете, и какую роль модификатор типа играет в спецификаторе формата?
88. С помощью какого флага можно выровнять выводимое значение влево? А каким образом вывести ведущие нули?
89. Какое действие оказывают на выводимую строку ширина, точнойть и флаги в спецификаторе формата?

90. Для чего в спецификаторе формата может использоваться символ звездочка («*»)? Чем отличается действие этого символа при воде и при выводе?
91. Каковы особенности ввода строк?
92. Каким образом ограничить набор вводимых символов при вводе?
93. Что является главной проблемой при использовании форматного ввода/вывода из библиотеки <cstdio>?
94. Объясните, для чего нужны строковые потоки. Почему строковые потоки —всегда форматируемые?
95. С помощью каких функций выполняется работа со строковыми потоками?
96. Можно ли использовать тип string (и каким образом) со строковыми потоками?
97. Объясните, в чем заключается различие между текстовым и двоичным файлом.
98. Объясните, что означает «открыть» файл и «закрыть» файл?
99. Каким образом внешний файл связывается с потоком?
100. Можно ли один и тот же поток связать с разными файлами? А один и тот же файл с разными потоками?
101. Перечислите режимы открытия файла. Чем отличается режим “r” от режима “a”?
102. Какую роль в режиме открытия играет знак плюс («+»)?
103. В каких случаях необходимо следить за ситуацией «конец файла»? Каким способом это делается?
104. Можно ли текстовый файл открыть как двоичный? А двоичный — как текстовый?
105. Какие функции ввода/вывода используются для обмена с текстовыми файлами?
106. Перечислите функции ввода/вывода для работы с двоичными файлами.
107. Какие функции реализованы в библиотеке <cstdio> для обеспечения прямого доступа к записям двоичного файла? Можно ли их использовать для работы с текстовыми файлами?
108. Объясните назначение функции fseek().
109. Чем отличается функция ftell() от функции fgetpos()?
110. Объясните, что означает «перенаправление» потока? Какие потоки можно перенаправлять и куда?
111. Каким образом перенаправление ввода можно использовать для ввода строк с пробелами?
112. В чем преимущества объектно-ориентированной библиотеки по сравнению с процедурной?
113. В каких состояниях может находиться поток? Каким образом отслеживается состояние «конец потока»?
114. Какие объектно-ориентированные потоки связаны со стандартными потоками?
115. Чем отличаются объектно-ориентированные строковые потоки от процедурных строковых потоков?
116. Каким образом строковые потоки можно использовать для ограничения ширины поля ввода? А можно ли с той же целью использовать строковые потоки <cstdio>?
117. Сравните средства форматирования объектно-ориентированной и процедурной библиотеки.
118. Каким образом ввести строку типа string с пробелами?
119. Каково назначение флагов форматирования? Какие средства реализованы в библиотеке для работы с флагами форматирования?
120. Что такое «манипулятор»? В чем преимущества манипуляторов перед флагами форматирования?
121. Как связываются файлы с потоками в объектно-ориентированной библиотеке?
122. Можно ли файлы, записанные функциями библиотеки <cstdio>, прочитать объектно-ориентированными средствами? А наоборот?

123. Перечислите режимы открытия объектно-ориентированных файловых потоков. каким образом комбинируются режимы открытия файловых потоков?
124. Обязательно ли закрывать файл, связанный с объектно-ориентированным файловым потоком? А открывать?
125. Каким образом открыть файловый поток для чтения и записи одновременно?
126. Как открыть файловый поток для дозаписи?
127. Можно ли вывести значение переменной в двоичном виде и как это сделать?
128. Разрешается ли наследовать от классов библиотеки ввода/вывода?
129. Каким образом можно перенаправить объектно-ориентированный поток?
130. Как используется буфер потока для копирования потока?
131. Какими операциями выполняется форматированный ввод/вывод в файловые потоки? А неформатированный?
132. Реализованы ли в объектно-ориентированной библиотеке средства прямого доступа к файловым потокам? Сравните их с аналогичными средствами библиотеки <cstdio>.
133. С какими объектно-ориентированными потоками разрешается, и с какими не разрешается использовать средства прямого доступа?
134. Покажите, каким образом можно выполнить перегрузку операций ввода/вывода для нового типа данных.
135. Как выполняется обработка ошибок ввода/вывода в объектно-ориентированной библиотеке?
136. Какое стандартное исключение генерируется при ошибках ввода/вывода? Обязательно ли оно генерируется?
137. Чем стандартные широкие потоки отличаются от узких?
138. Что такое — «локаль», и каково ее назначение?
139. Как установить русский шрифт при выводе в консольное окно?
140. Чем отличается ли ввод/вывод широких файловых потоков от узких?
141. Перечислите все последовательные контейнеры стандартной библиотеки. Чем они отличаются друг от друга?
142. Перечислите адаптеры последовательных контейнеров и дайте их подробную характеристику.
143. Почему для адаптеров-очередей нельзя использовать вектор в качестве базового?
144. Чем простая очередь queue отличается от приоритетной очереди priority_queue?
145. Каким требованиям должны удовлетворять элементы контейнера?
146. Могут ли быть указатели элементами контейнера? А итераторы?
147. Почему нельзя использовать в качестве элементов контейнера стандартный интеллектуальный указатель auto_ptr?
148. Зачем в контейнере list реализованы собственные методы сортировки поиска и слияния? Можно ли пользоваться соответствующими стандартными алгоритмами при обработке списка?
149. Перечислите типовые виды конструкторов, с помощью которых можно создавать последовательный контейнер.
150. Можно ли инициализировать контейнер элементами встроенного массива? А элементами другого контейнера? Какими способами это можно сделать?
151. Почему конструктор инициализации, параметрами которого являются итераторы, сделан шаблонным во всех контейнерах?
152. Какие методы реализованы в контейнере-векторе для доступа к элементам?
153. Отличается ли функция at() доступа по индексу от перегруженной операции индексирования и чем?

154. Перечислите методы контейнера deque, относящиеся к определению размеров контейнера.
155. Чем метод size() отличается от метода capacity()? А в чем отличие этих методов от метода max_size()?
156. Перечислите методы контейнера list, предназначенные для вставки удаления и замены элементов. Отличаются ли эти методы от соответствующих методов вектора и дека?
157. Каким образом выполняются операции сравнения контейнеров?
158. Разрешается ли изменять элемент ассоциативного контейнера, доступный в данный момент по итератору?
159. Какие контейнеры называются ассоциативными и почему?
160. Чем контейнер map отличается от контейнера multimap?
161. Объясните, почему в ассоциативных контейнерах нельзя изменять элемент, доступный в данный момент по итератору.
162. По каким причинам в контейнере-множестве не реализованы типовые операции объединения, пересечения, разности и другие?
163. Как используется структура-пара в ассоциативных контейнерах?
164. Объясните, что такое «критерий сортировки», и каким требованиям он должен удовлетворять? Какой критерий сортировки принят по умолчанию?
165. Какими преимуществами обладает функция make_pair() по сравнению с конструктором pair()?
166. Почему в контейнерах-отображениях операция индексирования перегружена, а в контейнерах-множествах — нет?
167. Какие гарантии безопасности обеспечивают контейнеры стандартной библиотеки?
168. Что такое «транзакционная гарантия безопасности» и чем она отличается от базовой?
169. На какие 4 класса по надежности можно разделить все операции с контейнерами?
170. Что такое «распределитель памяти» и зачем он нужен?
171. Чем отличается битовый вектор bitset от битового вектора vector<bool>?
172. Дайте определение итератора.
173. Что такое «начальный» итератор и «конечный» итератор? Какие методы, связанные с итераторами, обязательно включает каждый контейнер?
174. Чем константный итератор отличается от неконстантного?
175. Объясните, что такое «недействительный» итератор. В каких случаях итераторы становятся недействительными?
176. Какие категории итераторов вы знаете? Какие операции обязательно реализуются для всех категорий итераторов?
177. К какому виду итераторов можно отнести встроенный указатель и почему?
178. Какие вспомогательные функции для итераторов вы знаете? В каких случаях оправдано их применение?
179. Какие адаптеры итераторов реализованы в библиотеке?
180. Объясните, почему итераторы реализованы как вложенные классы в контейнерах.
181. Чем отличаются итераторы вставки от обычных итераторов?
182. Каким образом используются потоковые итераторы?
183. Какие стандартные функторы реализованы в библиотеке STL? Каково их основное назначение?
184. Для чего нужны адаптеры функторов bind1st() и bind2nd()?
185. Как применяются адаптеры-отрицатели?
186. Почему алгоритмы remove() не удаляют элементы из контейнеров? Как реально удалить элементы из контейнера?

187. Чем отличается стабильная сортировка от обычной?
188. Какую функцию выполняет алгоритмы unique()?
189. Могут ли стандартные алгоритмы работать со строками?
190. Нужно ли сортировать ассоциативные контейнеры?
191. Можно ли алгоритмы для работы с множествами применять для последовательных контейнеров? При каких условиях?
192. Какие алгоритмы предназначены для заполнения контейнера значениями? С какими контейнерами они могут работать?
193. Каким образом заполнить с помощью алгоритма generate() последовательный контейнер, не имеющий ни одного элемента?
194. Перечислите алгоритмы, предназначенные для операций с каждым элементом контейнера.
195. Можно ли с помощью алгоритма for_each() изменить элементы контейнера?

Типовой экзаменационный билет

ФГБОУ ВО «ЛУГАНСКИЙ НАЦИОНАЛЬНЫЙ УНИВЕРСИТЕТ ИМЕНИ ВЛАДИМИРА ДАЛЯ»

Кафедра Информационные и управляющие системы

Экзаменационный контроль **Семестр** 4

Дисциплина «Объектно-ориентированное программирование»

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 1

1. Модификаторы типа. Какую роль модификатор типа играет в спецификаторе формата?
2. Объясните, что такое «критерий сортировки», и каким требованиям он должен удовлетворять? Какой критерий сортировки принят по умолчанию?
3. Определить временную переменную, поместить в нее массив чисел, вводимый с клавиатуры. Присоединить к нему массив чисел, считанных из файла. Используя обобщенные сообщения итерации, вывести в окно системной информации и в файл массив проверки четности элементов полученного массива. Вывести в то же окно класс элементов нового массива.

Утверждено на заседании кафедры _____

Протокол № ___

Зав. кафедрой _____ Преподаватель _____

Критерии и шкала оценивания по оценочному средству итоговая аттестация (экзамен)

Шкала оценивания	Характеристика знания предмета и ответов
Отлично (5)	Студент глубоко и в полном объеме владеет программным материалом. Грамотно, исчерпывающе и логично его излагает в устной или письменной форме. При этом знает рекомендованную литературу, проявляет творческий подход в ответах на вопросы и правильно обосновывает принятые решения, хорошо владеет умениями и навыками при выполнении практических задач.

Хорошо (4)	Студент знает программный материал, грамотно и по сути излагает его в устной или письменной форме, допуская незначительные неточности в утверждениях, трактовках, определениях и категориях или незначительное количество ошибок. При этом владеет необходимыми умениями и навыками при выполнении практических задач.
удовлетворительно (3)	Студент знает только основной программный материал, допускает неточности, недостаточно четкие формулировки, непоследовательность в ответах, излагаемых в устной или письменной форме. При этом недостаточно владеет умениями и навыками при выполнении практических задач. Допускает до 30% ошибок в излагаемых ответах.
Неудовлетворительно (2)	Студент не знает значительной части программного материала. При этом допускает принципиальные ошибки в доказательствах, в трактовке понятий и категорий, проявляет низкую культуру знаний, не владеет основными умениями и навыками при выполнении практических задач. Студент отказывается от ответов на дополнительные вопросы.

3. Методические материалы, определяющие процедуры оценивания знаний, умений, навыков

Требования к выполнению контрольных заданий определены Методическими указаниями для лабораторных работ по дисциплине «Объектно-ориентированное программирование», (электронное издание) (для студентов по специальности 09.03.02 Информационные системы и технологии).

Контрольные сроки защиты лабораторных работ определены графиком учебного процесса, в частности, лабораторные работы 1-5 должны быть защищены в первой половине семестра, 6-10 до начала экзаменационной сессии.

Требования к выполнению курсовой работы определены методическими указаниями к курсовому проектированию по дисциплине «Объектно-ориентированное программирование», направление подготовки 09.03.02 Информационные системы и технологии (для студентов дневного и заочного обучения).

Контрольный срок защиты курсовой работы – до начала экзаменационной сессии.

Итоговая аттестация проводится в виде экзамена по билетам, содержащим 2 теоретических вопроса и одно практическое задание.

4. Лист изменений и дополнений

№ п/п	Виды дополнений и изменений	Дата и номер протокола заседания кафедры (кафедр[1]), на котором были рассмотрены и одобрены изменения и дополнения	Подпись (с расшифровкой) заведующего кафедрой (заведующих кафедрами)

Примечание:

[1] **-** для ФОС по государственной итоговой аттестации указываются реквизиты протоколов заседания кафедр и подписи заведующих кафедрами, деканов/директоров, совместно реализующих ОП.

Экспертное заключение

Представленный фонд оценочных средств (далее – ФОС) по дисциплине «Объектно-ориентированное программирование» соответствует требованиям ФГОС ВО.

Предлагаемые формы и средства текущего и промежуточного контроля адекватны целям и задачам реализации основной профессиональной образовательной программы по направлению подготовки 09.03.02 Информационные системы и технологии.

Оценочные средства для текущего контроля успеваемости, промежуточной аттестации по итогам освоения дисциплины представлены в полном объеме.

Виды оценочных средств, включенные в представленный фонд, отвечают основным принципам формирования ФОС.

Разработанный и представленный для экспертизы фонд оценочных средств рекомендуется к использованию в процессе подготовки обучающихся по указанному направлению.

Председатель учебно-методической комиссии факультета компьютерных систем и информационных технологий  Ветрова Н. Н.