МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
Институт кибербезопасности и цифровых технологий

Региональный партнёр
ФГБОУ ВО
«Дагестанский государственный технический университет»

УТВЕРЖДАЮ
И.о. ректора ФГБОУ ВО
Н.Л. Баламирзоев
«__» 09 2023 г.

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ

ПО УЧЕБНОЙ ДИСЦИПЛИНЕ

Б1.В.03 Технологии программирования

Направление подготовки 09.03.01 «Информатика и вычислительная техника»

Направленность (профиль подготовки): «Прикладной искусственный интеллект»

Квалификация выпускника бакалавр

Форма обучения очная

Махачкала 2023

ПАСПОРТ

фонда оценочных средств

по дисциплине Б1.В.03 Технологии программирования

(наименование дисциплины)

Результаты обучения по дисциплине Б1.В.03 Технологии программирования

1. :

Коды компе-тенции	Наименование компетенции	Индикатор достижения компетенции	Структурные элементы компетенции (в результате освоения дисциплины обучающийся должен знать, уметь, владеть)	Другая дисциплина (дисциплины)/практика, участвующая в формировании компетенции
ПК-1	Способен проектировать интеллектуальное программное обеспечение для решения практических задач	ПК-1.1. Осуществляет проектирование компонентов программного обеспечения с элементами искусственного интеллекта	Знать теоретические основы и современные информационные технологии проектирования и разработки программного обеспечения. Уметь использовать основные принципы объектно-ориентированного программирования при разработке программ сложной структуры. Владеть средствами проектирования ПО основе методов объектно-ориентированного программирования.	Программирование Технологии разработки интернет-ресурсов Интерфейсы программирования приложений Компьютерная графика и 3D моделирование Объектно-ориентированное программирование Учебная (ознакомительная) практика, Учебная (эксплуатационная) практика, Производственная (технологическая) практика,
		ПК-1.2. Создаёт варианты реализации компонент ПО на основе анализа предъявляемых требований	Знать теоретические основы и современные информационные технологии анализа требования к ПО, стандартные библиотеки программных модулей, используемые при	

			разработке программного обеспечения. Уметь формировать требования к программным проектам, использовать существующие типовые решения и шаблоны проектирования программного обеспечения. Владеть средствами формализации требования к программным проектам и инструментами разработки компонент ПО.	Производственная (эксплуатационная) практика, Производственная (проектно-технологическая) практика
		ПК-1.3. Понимает принципы организации взаимодействия элементов ПО в рамках единой архитектуры.	Знать принципы построения архитектуры программного обеспечения и виды архитектуры программного обеспечения. Уметь применять методы и средства проектирования программного обеспечения и программных интерфейсов. Владеть навыками организации взаимодействия между отдельными подсистемами в разрабатываемом комплексе ПО.	

2. Программа оценивания контролируемой компетенции:

№ п/п	Контролируемые разделы (темы) дисциплины	Код контролируемой компетенции / индикатора	Наименование оценочного средства

1	Классы в языке C++.	ПК-1 / ПК-1.2	Защита лабораторных работ 1-3. Защита курсовой работы Вопросы экзамена.
2	Функции класса в языке C++.	ПК-1 / ПК-1.2	Защита лабораторных работ 1-3. Защита курсовой работы Вопросы экзамена.
3	Наследование классов в языке C++.	ПК-1 / ПК-1.2	Защита лабораторной работы 4. Защита курсовой работы Вопросы экзамена.
4	Перегрузка операторов и функций в языке C++.	ПК-1 / ПК-1.2	Защита лабораторной работы 5. Защита курсовой работы Вопросы экзамена.
5	Полиморфизм в языке C++.	ПК-1 / ПК-1.2	Защита лабораторных работ 6-8. Вопросы экзамена.
6	Пространство имен в языке C++.	ПК-1 / ПК-1.2	Защита лабораторной работы 11. Защита курсовой работы Вопросы экзамена.
7	Шаблоны в языке C++.	ПК-1 / ПК-1.1	Защита лабораторной работы 9. Вопросы экзамена.
8	Библиотека STL.	ПК-1 / ПК-1.1	Защита лабораторной работы 10. Вопросы экзамена.
9	Исключения в языке C++.	ПК-1 / ПК-1.1	Защита лабораторной работы 11. Вопросы экзамена.
10	Типы значений в языке C#.	ПК-1 / ПК-1.2	Защита лабораторных работ 12-13. Защита курсовой работы. Вопросы экзамена.
11	Ссылочные типы в языке C#.	ПК-1 / ПК-1.2	Защита лабораторных работ 12-13. Защита курсовой работы. Вопросы экзамена.
12	Перегрузка операторов в языке C#.	ПК-1 / ПК-1.1	Защита лабораторной работы 14. Вопросы экзамена.
13	Переопределение функций в языке C#.	ПК-1 / ПК-1.2	Защита лабораторных работ 15-17. Вопросы экзамена.
Форма промежуточной аттестации в 5 семестре *– (экзамен,курс. работа)* Форма промежуточной аттестации в 6 семестре *– (экзамен)*			

Вопросы и задания для экзамена

по дисциплине Б1.В.03 Технологии программирования

1. Технология программирования. Понятие технологии программирования. Методология процедурно-ориентированного программирования. Абстракция и декомпозиция.
2. Технология программирования. Методология объектно-ориентированного программирования. Основные принципы ООП: наследование, инкапсуляция, полиморфизм.
3. Технология программирования. Методология объектно-ориентированного анализа и проектирования.
4. Технология программирования. Методология системного анализа и системного моделирования.
5. Методы проектирования программ. Метод нисходящего проектирования программ. Метод восходящего проектирования программ. Метод смешанного проектирования. Метод объектно-ориентированного проектирования.
6. C++. Понятие класса. Объявление класса. Объявление экземпляров класса. Спецификаторы доступа. Указатель this.
7. C++. Конструкторы. Деструктор. Создание и удаление объекта класса. Операторы new и delete. Указание размещения.
8. C++. Функции класса. Объявление функции. Реализация функции. Вызов функции. Встраиваемые функции.
9. C++. Функции. Передача параметров. Возвращаемое значение.
10. C++. Функции. Перегрузка имен функций. Перегрузка и возвращаемые типы. Перегрузка и область видимости.
11. C++. Функции. Перегрузка имен функций. Явное разрешение неоднозначности. Разрешение в случае нескольких параметров. Перегрузка функций класса.
12. C++. Функции. Значения параметров по умолчанию. Неопределенное число параметров.
13. C++. Наследование. Производные классы. Конструкторы и деструкторы базовых и производных классов. Порядок создания и удаления классов и членов классов.
14. C++. Наследование. Спецификаторы доступа для базовых классов. Доступ к базовым классам. Доступ к членам базового класса. Стандартные преобразования типов.
15. C++. Наследование. Множественное наследование. Множественное вхождение базового класса.
16. C++. Перегрузка операторов. Перегрузка операторов new и delete.
17. C++. Полиморфизм. Поля типа.
18. C++. Полиморфизм. Виртуальные функции. Таблица виртуальных функций.
19. C++. Полиморфизм. Чистая виртуальная функция. Абстрактные классы. Виртуальные и перегруженные функции.
20. C++. Вложенные классы. Локальные классы.
21. C++. Пространство имен. Using-объявление и using-директива.

22. C++. Шаблоны. Параметризованные функции. Шаблоны функций с несколькими параметрами. Перегрузка шаблонов функций. Взаимозаменяемость параметризованных функций.
23. C++. Шаблоны. Параметризованные классы. Специализация шаблонов класса. Частичная специализация шаблонов класса.
24. STL. Контейнеры. Типы контейнеров. Вектор (vector), список (list), дек (deque). Множество (set), словарь (map), стек (stack).
25. STL. Итераторы. Типы итераторов. Алгоритмы.
26. STL. Аллокаторы. Функциональные объекты. Строки.
27. C++. Исключения. Спецификация исключений. Повторная генерация исключений. Передача параметров в блок catch.
28. C#. Общие сведения. Платформа .NET Framework. Типы данных. Типы значений: простые типы.
29. C#. Структуры. Конструкторы и деструктор. Объявление экземпляра структуры. Производные структуры.
30. C#. Ссылочные типы: общие сведения. Интерфейсы. Наследование интерфейсов.
31. C#. Классы. Спецификаторы доступа. Конструкторы и деструктор. Производные классы. Множественное наследование.
32. C#. Классы. Виртуальные функции. Абстрактные классы и абстрактные функции.
33. C#. Классы. Запрещение наследования. Вызов конструктора базового класса. Доступ к членам базового класса.
34. C#. Делегаты. Групповая адресация. Управление объектами.
35. C#. Объекты. Строки. Пустой тип. Неявный тип. Тип объекта (Type). Типы, допускающие значение NULL.
36. C#. Массивы. Инициализация массива. Ступенчатый массив. Свойство Length.
37. C#. Перегрузка операторов.
38. C#. Переопределение функций.
39. C#. Свойства и Индексаторы.
40. Введение. Понятие технологии программирования. Предмет и задачи дисциплины.
41. Методология изучения технологии программирования
42. Перечень вопросов, относящихся к технологии разработки программного обеспечения
43. Связь с другими дисциплинами.
44. Программное обеспечение.
45. Понятие программного изделия (ПИ).
46. Классификация и функции ПИ.
47. Состав программного обеспечения
48. Жизненный цикл программного обеспечения.
49. Абстрагирование.
50. Теории классификации.
51. Методы выявления классов и объектов.
52. Формат объявления переменной. Свойства переменной. Синтаксические и семан-

тические ошибки. Область действия идентификатора.*

53. Кодирование объектно-ориентированных систем.

54. Современные технологии объектно-ориентированного программирования. Стратегии и методы тестирования. Прямое и обратное тестирование.*

Задания:

1. Задан класс C++. Необходимо добавить конструктор по умолчанию.
2. Задан класс C++. Необходимо добавить копирующий конструктор.
3. Задан класс C++. Необходимо добавить два или более конструкторов с параметрами.
4. Задан класс C++. Необходимо добавить встраиваемую функцию.
5. Задан класс C++. Необходимо добавить функцию с передачей параметров по значению.
6. Задан класс C++. Необходимо добавить функцию с передачей параметров по указателю.
7. Задан класс C++. Необходимо добавить функцию с передачей параметров по ссылке.
8. Задан класс. C++ Необходимо выполнить перегрузку одной любой функции.
9. Задан класс C++. Необходимо добавить функцию с несколькими параметрами. Часть параметров должна иметь значения по умолчани.
10. Задан класс C++. Необходимо добавить функцию с неопределенным числом параметров.
11. Задан класс C++. Необходимо создать производный класс.
12. Задан базовый и производный классы C++. В производном классе продемонстрировать выбор конструктора базового класса.
13. Задан базовый и производный классы C++. Добавить производный класс, продемонстрировав множественное вхождение базового класса.
14. Задан класс C++. Выполнить перегрузку оператора ==.
15. Задан базовый и производный классы C++. Добавить виртуальную функцию.
16. Задан класс C++. Переместить класс внутрь пространства имен Test.
17. Задан класс C++. Реализовать параметризованную функцию сравнения на равенство. Проверить на типах int и классе.
18. Продемонстрировать использование STL контейнера словарь (map).
19. Задан класс C#. Необходимо добавить копирующий конструктор.
20. Задан класс C#. Необходимо добавить два или более конструкторов с параметрами.
21. Задан класс C#. Необходимо разработать интерфейс и пронаследовать от него имеющийся класс.
22. Продемонстрировать использование делегатов в C# на примере структуры и класса.
23. Задан класс C#. Добавить в него одно свойство и один индексатор.

Описание показателей и критериев оценивания с указанием шкалы оценивания для очной и других форм обучения (с применением балльно-рейтинговой системы):

Оцениваются следующие показатели: понимание вопросов, правильность, полнота и логическое изложение ответов.

Оценка по дисциплине складывается из текущего рейтинга и рейтинга промежуточной аттестации.

Экзаменационный рейтинг промежуточной аттестации определяется следующим образом:

Ответы на 1, 2 вопрос – до 15 баллов, ответ на дополнительный вопрос – до 10 баллов.

Оценивание ответов на любой из теоретических вопросов:

13-15 баллов выставляется, если студент демонстрирует полное понимание вопросов, правильность ответов, полное и логически последовательное изложение материала.

11-12 баллов выставляется, если студент демонстрирует: значительное понимание вопросов, правильность, но недостаточную полноту ответов на заданные теоретические вопросы; допущение неточности ответа;

9-10 баллов выставляется, если студент демонстрирует: понимание вопросов, по существу излагает материал, но не усвоил его деталей, есть погрешности в ответах; допускает неточности, недостаточно правильные формулировки, нарушения логической последовательности в изложении материала;

Менее 9 баллов выставляется, если студент демонстрирует: непонимание вопросов; студент не знает значительной части материала, не ответил на дополнительные вопросы или отказался от ответов на вопросы и задания.

Оценивание ответов на дополнительный вопрос:

9-10 баллов выставляется, если студент демонстрирует полное понимание вопросов, правильность ответов, полное и логически последовательное изложение материала.

7-8 баллов выставляется, если студент демонстрирует: значительное понимание вопросов, правильность, но недостаточную полноту ответов на заданные теоретические вопросы; допущение неточности ответа;

6-7 баллов выставляется, если студент демонстрирует: понимание вопросов, по существу излагает материал, но не усвоил его деталей, есть погрешности в ответах; допускает неточности, недостаточно правильные формулировки, нарушения логической последовательности в изложении материала;

Менее 6 баллов выставляется, если студент демонстрирует: непонимание вопросов; студент не знает значительной части материала, не ответил на дополнительные вопросы или отказался от ответов на вопросы и задания.

Минимальный балл экзаменационного рейтинга в соответствии с положением о рейтинге равен 24.

В итоге по курсу, суммируя итоги текущего рейтинга и рейтинга промежуточной аттестации:

- оценка «отлично» выставляется обучающемуся, если он набрал 87-100 баллов;
- оценка «хорошо» выставляется обучающемуся, если он набрал 73-86 баллов;
- оценка «удовлетворительно» выставляется обучающемуся, если он набрал 60-72 балла;
- оценка «неудовлетворительно» выставляется обучающемуся, если он набрал менее 60 баллов;

Перечень тем для курсовых работ

по дисциплине Б1.В.03 Технологии программирования

(наименование дисциплины)

1. Редактор графических изображений.
2. Текстовый редактор с поддержкой различных кодировок.
3. Проигрыватель аудио файлов.
4. Проигрыватель видео файлов.
5. Файловый менеджер.
6. Программа управления обоими рабочего стола.
7. Электронный органайзер.
8. Программа просмотра текстовой информации в формате HTML.
9. Чат.
10. Программа сбора информации о компьютере.
11. Редактор реестра.
12. Программа управления файловыми ассоциациями.
13. Электронная записная книжка.
14. Программа построения графиков функций.
15. Словарь.
16. База данных по музыкальным произведениям.
17. Программа учета складского товара.

Описание показателей и критериев оценивания с указанием шкалы оценивания для очной и других форм обучения (с применением балльно-рейтинговой системы):

Оценка за курсовую работу складывается из текущего рейтинга и рейтинга за защиту курсовой работы.

Текущий контроль осуществляется поэтапно.

Текущий рейтинг определяется: правильно подобранным материалом, знанием теоретических основ соответствующих разделов курсовой работы, умением применить их на практике и обосновать используемое решение. На каждом этапе текущего контроля обучающийся может набрать:

18-30 баллов, если студент подобрал и освоил теоретический материал, соответствующий разрабатываемой теме, выполнил необходимые этапы при разработке курсовой работы в соответствии с требованиями задания, владеет теоретическим материалом, связывает его с программной реализацией.

0-18 баллов выставляется, если студент недостаточно проработал этапы курсовой работы в соответствии с требованиями задания, недостаточно правильно и полно владеет теоретическим материалом, не связывает его с программной реализацией.

Рейтинг за защиту курсовой работы определяется следующим образом.

Оцениваются следующие показатели: соответствие требованиям задания, соответствие требованиям оформления отчета, правильность схем алгоритмов, расчетов и работы разработанной программы во время демонстрации, корректность и обоснованность выводов, самостоятельность выполненной работы.

Максимальная оценка при защите курсовой работы – 40 баллов; минимальная – 24 балла. Минимальный балл, свидетельствующий об успешной защите курсовой работы – 24.

32-40 баллов выставляется, если выполнены все требования, предъявляемые к выполнению курсовой работы, выполнены требования по оформления отчета, приведены правильные схемы алгоритмов, расчетов и разработанная программа работает корректно, предоставлены выводы, работа выполнена самостоятельно, даны полные и правильные ответы на поставленные вопросы.

24-31 балла выставляется, если требования, предъявляемые к выполнению курсовой работы, выполнены с существенными отклонениями и даны неполные и неточные ответы на поставленные вопросы.

0-23 балла выставляется, если не выполнены требования, предъявляемые к выполнению курсовой работы, студент не отвечает на поставленные вопросы.

Оценка за курсовую работу оценивается следующим образом:

- оценка «отлично» выставляется обучающемуся, если он набрал 87-100 баллов;
- оценка «хорошо» выставляется обучающемуся, если он набрал 73-86 баллов;
- оценка «удовлетворительно» выставляется обучающемуся, если он набрал 60-72 балла;
- оценка «неудовлетворительно» выставляется обучающемуся, если он набрал менее 60 баллов;

Вопросы для защиты лабораторных работ и контроль выполнения лабораторных работ

по дисциплине Б1.В.03 Технологии программирования

(наименование дисциплины)

Лабораторная работа 1.

Проверка хода выполнения лабораторной работы.

Вопросы для защиты лабораторной работы:

1. Что такое класс в языке C++?
2. Как определяется класс?
3. Что такое спецификатор доступа?
4. Поясните назначение конструкторов и деструкторов.
5. Поясните особенности конструкторов с параметрами и способы передачи параметров в конструкторы.
6. Какие существуют способы создания объекта класса и чем они отличаются?

Лабораторная работа 2.

Проверка хода выполнения лабораторной работы.

Вопросы для защиты лабораторной работы:

1. Чем отличается класс от объекта класса (экземпляра класса)?
2. Как создаются объекты класса?
3. Как создать объект в динамической памяти? Как его уничтожить?
4. Что делают операторы new и delete?
5. Опишите жизненный цикл объекта.

Лабораторная работа 3.

Проверка хода выполнения лабораторной работы.

Вопросы для защиты лабораторной работы:

1. Что такое копирующий конструктор? В чем его назначение.
2. Какие существую способы передачи параметров в C++?
3. Что такое значение параметров по умолчанию?
4. Сколько можно задавать значений параметров по умолчанию?

Лабораторная работа 4.

Проверка хода выполнения лабораторной работы.

Вопросы для защиты лабораторной работы:

1. Что такое наследование?
2. Как описывается производный класс?

3. Как определяется видимость членов производных классов?
4. В чем заключается особенность вызова конструкторов при наследовании?
5. В чем заключается множественное наследование?

Лабораторная работа 5.

Проверка хода выполнения лабораторной работы.

Вопросы для защиты лабораторной работы:

1. Что такое перегрузка операторов?
2. С какой целью выполняется перегрузка операторов?
3. Как переопределить оператор в языке C++?
4. Какие оператор можно перегружать в языке C++?
5. Как определяется приоритет перегружаемого оператора?
6. Как называется механизм, когда для одного оператора выполнено несколько реализаций с разным набором параметров?

Лабораторная работа 6.

Проверка хода выполнения лабораторной работы.

Вопросы для защиты лабораторной работы:

1. Что такое полиморфизм и как он связан с виртуальными функциями?
2. Что такое виртуальная функция?
3. Что такое абстрактный класс?
4. Когда включается механизм виртуализации?
5. Каким образом осуществляется выбор требуемого варианта реализации виртуальной функции?
6. Что такое таблица виртуальных функций?

Лабораторная работа 7.

Проверка хода выполнения лабораторной работы.

Вопросы для защиты лабораторной работы:

1. Что такое вложенный класс?
2. Как задать видимость вложенного класса?
3. Каково назначение виртуального деструктора?

Лабораторная работа 8.

Проверка хода выполнения лабораторной работы.

Вопросы для защиты лабораторной работы:

1. Что такое сериализация и восстановление (десериализация)?

Лабораторная работа 9.

Проверка хода выполнения лабораторной работы.

Вопросы для защиты лабораторной работы:

1. Каково назначение шаблонов?
2. Каким образом описывается тип данных, для которого реализуется шаблон?
3. Допускается ли описание нескольких типов данных для одной параметризованной функции или одного параметризованного класса?
4. В чем заключается отличие в реализации между перегрузкой операторов и параметризованными функциями (на примере сравнения двух объектов)?

Лабораторная работа 10.

Проверка хода выполнения лабораторной работы.

Вопросы для защиты лабораторной работы:

1. Каково назначение библиотеки STL?
2. Какие компоненты включает библиотека STL?
3. Какие типы контейнеров включает библиотека STL?
4. Что такое итератор?

Лабораторная работа 11.

Проверка хода выполнения лабораторной работы.

Вопросы для защиты лабораторной работы:

1. Что такое исключение?
2. Что такое пространство имен и как оно описывается?
3. Как генерируется исключение?
4. Как перехватывается исключение?
5. Зачем необходимо использовать пространство имен?

Лабораторная работа 12.

Проверка хода выполнения лабораторной работы.

Вопросы для защиты лабораторной работы:

1. Перечислите основные особенности синтаксиса языка C#?
2. Перечислите основные особенности семантики языка C#?
3. Как определяется класс в C#?
4. Как задается наследование в C#?

Лабораторная работа 13.

Проверка хода выполнения лабораторной работы.

Вопросы для защиты лабораторной работы:

1. Что такое интерфейс в C#?

2. Что такое виртуальная функция в C#?
3. Чем отличается абстрактный класс от виртуального?
4. Какие спецификаторы доступа используются в интерфейсе?

Лабораторная работа 14.

Проверка хода выполнения лабораторной работы.

Вопросы для защиты лабораторной работы:

1. Как выполняется перегрузка операторов в C#?
2. Какие операторы нельзя перегружать в C#?
3. Какие особенности перегрузки операторов в C#?

Лабораторная работа 15.

Проверка хода выполнения лабораторной работы.

Вопросы для защиты лабораторной работы:

1. Что такое делегат C#?
2. Что такое группирование?
3. Что такое разгруппирование?
4. Как вызываются делегаты?
5. Какой результат получается после вызова делегата?

Лабораторная работа 16.

Проверка хода выполнения лабораторной работы.

Вопросы для защиты лабораторной работы:

1. Что такое коллекция?
2. Какие стандартные коллекции в C#?
3. Что такое файловый поток?
4. Как работают файловые потоки в C#?

Лабораторная работа 17.

Проверка хода выполнения лабораторной работы.

Вопросы для защиты лабораторной работы:

1. Что такое маршалинг?
2. Для чего используется маршалинг в C#?

Описание показателей и критериев оценивания с указанием шкалы оценивания для очной и других форм обучения (с применением балльно-рейтинговой системы)

Оцениваются следующие показатели: знание теоретических основ лабораторной работы, умение применить их на практике, обосновать используемое решение, выполнение в установленные сроки. В рамках защиты по каждой лабораторной работе задается несколько вопросов.

Для лабораторных работ 1-11

6 баллов выставляется, если студент выполнил работы в установленный срок, правильно и полно отвечает на вопросы по каждой лабораторной работе, объясняет их на примерах, связывает с программной реализацией.

4-5 баллов выставляется, если студент отвечает на вопросы недостаточно полно или с неточностями, или не отвечает на часть заданных вопросов, не может объяснить их на примере, есть недочеты в лабораторной работе.

0-3 *балла выставляется, если* студент не отвечает на вопросы, не может объяснить их на примере, лабораторная работа выполнена некорректно.

Для лабораторных работ 12-13

3 балла выставляется, если студент выполнил работы в установленный срок, правильно и полно отвечает на вопросы по каждой лабораторной работе, объясняет их на примерах, связывает с программной реализацией.

2 балла выставляется, если студент отвечает на вопросы недостаточно полно или с неточностями, или не отвечает на часть заданных вопросов, не может объяснить их на примере, есть недочеты в лабораторной работе.

0-1 *балл выставляется, если* студент не отвечает на вопросы, не может объяснить их на примере, лабораторная работа выполнена некорректно.

Для лабораторных работ 14-17

6 баллов выставляется, если студент выполнил работы в установленный срок, правильно и полно отвечает на вопросы по каждой лабораторной работе, объясняет их на примерах, связывает с программной реализацией.

4-5 баллов выставляется, если студент отвечает на вопросы недостаточно полно или с неточностями, или не отвечает на часть заданных вопросов, не может объяснить их на примере, есть недочеты в лабораторной работе.

0-3 *балла выставляется, если* студент не отвечает на вопросы, не может объяснить их на примере, лабораторная работа выполнена некорректно.

Темы групповых и/или индивидуальных творческих заданий/проектов**

по дисциплине Б1.В.03 Технологии программирования

(наименование дисциплины)

Разработать, отладить программу, демонстрирующую использование указанного шаблона проектирования.

Индивидуальные творческие задания/проекты к лабораторной работе 1:

Разработать консольное приложение, в котором должен быть реализован простой класс в соответствии с вариантом задания.

1. Класс «Автомобиль» (легковой/грузовой, мощность, объем двигателя, цена и т.д.).
2. Класс «Компьютер» (процессор, объем оперативной памяти, цена и т.д.).
3. Класс «Телефон» (производитель, модель, размер экрана и т.д.).
4. Класс «Книга» (тема, автор, издательство, год выпуска и т.д.).
5. Класс «Журнал» (название, тема, тираж и т.д.).
6. Класс «Студент» (ФИО, ВУЗ, факультет, кафедра, форма обучения и т.д.).
7. Класс «Сотрудник» (ФИО, предприятие, должность, зарплата, стаж и т.д.).
8. Класс «Музыкальное произведение» (название, автор, исполнитель, жанр и т.д.)
9. Класс «Фильм» (название, режиссер, студия, жанр и т.д.)
10. Класс «Населенный пункт» (название, количество жителей, географические координаты, виды общественного транспорта и т.д.).
11. Класс «Предприятие» (название, адрес, направление производства, количество сотрудников и т.д.).
12. Класс «Магазин» (название, адрес, торговая площадь, количество сотрудников и т.д.).
13. Класс «Кинотеатр» (название, адрес, количество залов, репертуар и т.д.).
14. Класс «Телевизор» (производитель, размер экрана, поддерживаемые форматы, цена и т.д.).
15. Класс «Компьютерная игра» (название, жанр, производитель, цена и т.д.).

Индивидуальные творческие задания/проекты к лабораторной работе 2:

На базе лабораторной работы №1 доработать класс с использованием операторов new и delete.

- В классе все строковые переменные, которые ранее были объявлены как массив символов (char str1[100] или wchat_t str2[200]), необходимо заменить на

указатели (char* str1 или wchat_t* str2). Во всех функциях, которые обрабатывают строковые переменные, сделать изменения касающиеся размещения/удаления памяти под строки.

- В программе все объекты класса заменить на указатели, в том числе и в массиве (из массива объектов сделать массив указателей).
- Выполнить размещение всех объектов класса с использованием оператора **new**, в том числе выполнить размещение объектов в массиве.
- Добавить в программу удаление всех размещенных объектов класса с использованием оператора **delete**, в том числе выполнить удаление объектов из массива.

Индивидуальные творческие задания/проекты к лабораторной работе 3:

- В классе реализовать копирующий конструктор (через указатель или ссылку).
- В классе для одного из конструкторов с параметрами задать значения двух или более параметров по умолчанию.
- В программу добавить создание нового объекта класса с вызовом копирующего конструктора. В качестве исходного объекта можно использовать любой ранее созданный объект. Для проверки правильности работы следует выполнить следующее:
  - Вызвать функцию Output для исходного и вновь созданного объекта.
  - Удалить исходный объект и вызвать функцию Output для сохранившегося нового объекта.
  - Удалить новый объект.
- В программу добавить несколько вариантов создания нового объекта класса с вызовом конструктора, у которого заданы параметры по умолчанию. Должны быть варианты:
  - Все параметры, которые объявлены как параметры по умолчанию, не заданы.
  - Все параметры, которые объявлены как параметры по умолчанию, явно заданы.
  - Часть параметров, которые объявлены как параметры по умолчанию, явно заданы, а часть не заданы.

Индивидуальные творческие задания/проекты к лабораторной работе 4:

Необходимо разработать не менее 3 производных классов. Каждый производный класс должен расширять базовый. В каждый производный класс необходимо добавить переменные и функции, уточняющие базовый класс.

Структура классов должна быть следующей:

- **Base** : **Derived1**
- **Base** : **Derived2**
- **Derived2** : **Derived3**

В качестве базового класс **Base** берется класс, разработанный в лабораторной работе №3. Производные классы **Derived1** и **Derived2** должны быть пронаследованы от базового класса. Класс **Derived3** должен быть пронаследован от производного класса **Derived2**.

Во всех производных классах **Derived1**, **Derived2** и **Derived3** должны быть реализованы два метода: Input и Output. Методы должны использовать вызовы из

родительского класса. Т.е. в производном классе в функциях первоначально должны быть вызваны функции базового класса (для **Derived1** и **Derived2** функции из класса **Base**, для **Derived3** функции из класса **Derived2**), а затем выполнен ввод и вывод данных текущего класса.

Индивидуальные творческие задания/проекты к лабораторной работе 5:

Лабораторное задание выполняется на базе лабораторной работы №4. В производный класс **Derived3** добавляется реализация операторов в соответствии с вариантом задания.

Задание включает реализацию 2 операторов:

- Оператор, который возвращает логическое значение (истина или ложь).
- Оператор, который выполняет изменение данных класса.

Каждый оператор должен быть реализован для 3 типов параметров:

- Параметр целочисленного типа **int**.
- Параметр строкового типа **char*** (или **wchar_t***).
- Параметр ссылки на исходный класс **Derived3&** (или **const Derived3&**).

Каждый оператор должен быть реализован и внутри и вне описания класса. Т.е. как минимум одна реализация (для одного типа параметров) должна быть внутри описания класса и как минимум одна реализация должна быть вне описания класса.

В основной функции программы должен присутствовать вызов каждого оператора для всех типов параметров.

Индивидуальные творческие задания/проекты к лабораторной работе 6:

Реализовать абстрактный класс **Interface**. В классе объявить чистые виртуальные функции: Input и Output.

Пронаследовать класс **Base** от абстрактного класса **Interface**. В результате структура классов должна быть следующей:

- **Base** : **Interface**
- **Base** : **Derived1**
- **Base** : **Derived2**
- **Derived2** : **Derived3**

В классах **Derived1**, **Derived2** и **Derived3** реализовать виртуальные функции Input и Output.

Выполнить перегрузку операторов **new** и **delete**. В операторах реализовать учет выделяемой/освобождаемой динамической памяти. Для этого объявить в программе глобальную переменную, значение которой в операторе **new** должно увеличиваться на величину выделяемой памяти, а в операторе **delete** уменьшаться на размер освобождаемой памяти. Функции для работы с памятью:

- **HeapAlloc** – выделение памяти.
- **HeapFree** – освобождение памяти.
- **HeapSize** – получение размера выделенной памяти.

Индивидуальные творческие задания/проекты к лабораторной работе 7:

В работе требуется разработать класс **<Структура>**, реализующий функционал в соответствии с вариантом задания.

- Класс в соответствии с заданием должен формировать динамическую структуру данных из объектов **Derived1**, **Derived2** и **Derived3**. Для этого в классе должен быть реализован вложенный класс **<Элемент структуры>**.
- Класс **<Элемент структуры>** является служебным классом, который описывает одну единицу данных, входящую в структуру данных. Класс должен включать в себя весь необходимый набор переменных для организации структуры данных (указатель на следующий/предыдущий элемент данных и т.п.). Одним из полей класса **<Элемент структуры>** должна быть переменная типа **Interface***. Данная переменная должна использоваться для размещения адреса объекта классов **Derived1**, **Derived2** и **Derived3**.
- В классе **<Структура>** должны быть реализованы:
  - Функции, указанные в варианте задания.
  - Функции обхода всех элементов структуры данных: GetFirst и GetNext. Функция GetFirst должна возвращать указатель типа **Interface*** на первый элемент данных. Функция GetNext должна возвращать указатель типа **Interface*** на следующий элемент данных. При этом для линейных структур данных выход за границу структуры должен вызывать ошибку (функция GetFirst и/или GetNext должны возвращать NULL). Для циклических структур данных выход за границу структуры должен вызывать переход к первому/последнему элементу.
- В классе **<Структура>** в деструкторе должно выполняться удаление всех элементов структуры.
- В основной программе должен быть реализован цикл с возможными операциями:
  - добавление элемента в структуру;
  - удаление элемента из структуры;
  - вывод структуры на экран.
- Добавление элемента в структуру должно предлагать пользователю выбрать тип создаваемого объекта: **Derived1**, **Derived2** или **Derived3**. Для размещенного объекта должна быть вызвана функция Input.
- Вывод структуры на экран должен быть выполнен на базе функций GetFirst и GetNext, реализованных в классе структуры. С использованием данных функции необходимо перебирать все элементы данных в структуре и поочередно печатать их на экран. Печать каждого элемента должна выполняться через вызов функции Output. Печать структуры должна быть реализована в 2-х режимах:
  - целиком вся структура данных;
  - отдельно по элементам, чтобы была продемонстрирована ситуация выхода за границу структуры.

Индивидуальные творческие задания/проекты к лабораторной работе 8:

В класс **Interface** добавить две чистые виртуальные функции: Serialize и Deserialize.

В классах **Derived1**, **Derived2** и **Derived3** реализовать виртуальные функции Serialize и Deserialize.

- Serialize – функция сериализация данных. Функция должна сохранять все свои данные в буфер, указанный в параметрах функции.
- Deserialize – функция восстановления данных. Функция должна восстанавливать данные объекта из буфера, указанного в параметрах функции.

В класс **<Структура>** добавить два метода:

- Serialize – функция сериализация структуры данных. Функция должна сохранить все элементы структуры в файл. В своей работе функция должна вызывать соответствующую функцию Serialize для всех объектов.
- Deserialize – функция восстановления структуры данных. Функция должна восстанавливать все элементы структуры из файла. Во время работы функция должна вызывать соответствующую функцию Serialize для всех элементов. В процессе восстановления данных все элементы должны заново размещаться в динамической памяти. Т.е. все объекты классов **Derived1**, **Derived2** и **Derived3** должны заново создаваться, после чего размещенные объекты добавляются в структуру.
- В основной программе необходимо разместить объект класса **<Структура>** в динамической памяти.
- Выполнить добавление нескольких различных элементов в структуру (**Derived1**, **Derived2** и **Derived3**).
- Вывести на экран содержимое всей структуры.
- Сериализовать всю структуру в файл.
- Удалить объект класса **<Структура>** из динамической памяти. При этом все элементы структуры также должны быть удалены.
- Разместить новый объект класса **<Структура>** в динамической памяти.
- Восстановить сохраненную структуру данных из файла.
- Вывести на экран содержимое всей структуры. Повторная печать должна полностью повторить содержимое первой печати, доказав таким образом что данные восстановлены корректно.

Индивидуальные творческие задания/проекты к лабораторной работе 9:

Лабораторная работа выполняется на базе работ №1 и №7. Из работы №1 берется базовый класс, из работы №7 берется динамическая структура.

В работе требуется реализовать две параметризованные функции и один (или более) параметризованный класс.

- Шаблонные функции должны выполнять сравнение двух объектов класса из лабораторной работы №1.
  - Первая функция должна сравнивать объекты на больше (>).
  - Вторая функция должна сравнивать объекты на меньше или равно (<=).
- Шаблонный класс должен реализовывать функционал динамической структуры данных. Вариант структуры берется из лабораторной работы №7.
- Класс должен поддерживать формирование динамической структуры для фундаментальных (int, short, void* и т.п.) и производных типов данных (классы, структуры). В качестве производного типа данных должен использоваться класс из лабораторной работы №1.
- Класс из лабораторной работы №1 должен быть дополнен всеми необходимыми функциями (конструкторами) и операторами для обеспечения работы шаблонных функций и шаблонного класса.
- В программе должны несколько раз вызываться шаблонные функции для сравнения различных объектов класса.
- В программе должны быть реализованы две динамические структуры:
  - Первая, для объектов фундаментального типа (int, short, void* и т.п.).

- Вторая, для объектов производного типа. Класс берется из лабораторной работы №1.

- Для всех динамических структур должны быть реализованы циклы с операциями:
  - добавление элемента в структуру;
  - удаление элемента из структуры;
  - вывод структуры на экран.

Индивидуальные творческие задания/проекты к лабораторной работе 10:

Для указанного в лабораторной работе №1 базового класса реализовать структуру данных в соответствии с лабораторным заданием.

- Класс из лабораторной работы №1 должен быть дополнен всеми необходимыми функциями (конструкторами) и операторами для обеспечения работы STL контейнеров и функций.
- В классе все строковые данные представить с использованием классов string (или wstring).
- Программа должна реализовать контейнер объектов класса в соответствии с вариантом задания.
- Для работы с контейнером должны быть реализованы циклы с операциями:
  - добавление элемента в контейнер;
  - удаление элемента из контейнера;
  - вывод контейнера на экран.

Индивидуальные творческие задания/проекты к лабораторной работе 11:

Требуется разработать набор классов описания исключительной ситуации. Для этого необходимо реализовать базовый класс **CMyException** и два производных класса (**CMyException1** и **CMyException2**), детализирующих исключительную ситуацию.

Требуется разработать три функции: **Func1**, **Func2** и **Func3**, которые сгенерируют исключения в зависимости от заданного условия (условие определяется вариантом задания).

Функция **Func1** должна открывать файл с указанным именем и возвращать файловый идентификатор. Функция может генерировать два исключения (**CMyException** и **CMyException1**). Исключения генерируются при проверке имени файла.

Функция **Func2** должна читать содержимое файла в буфер. Функция может генерировать одно исключение (**CMyException2**). Исключение генерируется при проверке прочитанных данных.

Функция **Func3** должна закрывать файловый идентификатор. Функция может генерировать одно исключение (**CMyException**). Исключение генерируется при проверке файлового указателя.

Требуется разработать функцию **FuncHandle**, реализующую обработку исключительных ситуаций. В данной функции внутри блока **try** должны поочередно вызываться функции **Func1**, **Func2** и **Func3**. В функции должны быть реализованы несколько блоков **catch**, для каждого возможного типа исключения. Внутри блока необходимо **catch** вывести информацию по ошибке (функция, описание и т.д.).

В основной функции необходимо несколько раз вызвать функцию **FuncHandle** задав такие условия, чтобы поочередно были сгенерированы все возможные исключения. В том числе должен быть такой вызов **FuncHandle**, который не генерирует исключения.

- Классы должны содержать сведения о функции, в которой сгенерировано исключение (текстовое имя).
- Классы должны содержать общее описание ошибки (текстовая строка).
- Производные классы должны содержать дополнительные индивидуальные данные по ошибке.

Все классы и функции (кроме функции **main**) должны быть реализованы внутри собственного пространства имен.

Индивидуальные творческие задания/проекты к лабораторной работе 12:

Разработать консольное приложение, в котором должен быть реализован набор классов в соответствии с вариантом задания.

Должны быть разработаны базовый класс (**Base**) не менее 3 производных классов (**Derived1**, **Derived2** и **Derived3**). Каждый производный класс должен расширять базовый. В каждый производный класс необходимо добавить переменные и функции, уточняющие базовый класс.

Структура классов должна быть следующей:

- **Base** : **Derived1**
- **Base** : **Derived2**
- **Derived2** : **Derived3**

Производные классы **Derived1** и **Derived2** должны быть пронаследованы от базового класса (**Base**). Класс **Derived3** должен быть пронаследован от производного класса **Derived2**.

- Класс должен содержать конструктор без параметров. В конструкторе должны инициализироваться переменные класса значением по умолчанию.
- Класс должен содержать два или более конструкторов с параметрами. В конструкторах должны инициализироваться переменные класса значениями, которые переданы в параметрах конструктора. Если для какой-либо переменной класса в параметрах конструктора не предусмотрен параметр, то переменная должна инициализироваться значением по умолчанию.
- Класс должен содержать деструктор. В деструкторе все переменные класса должны быть затерты (присвоены недействительные значения).
- Класс должен содержать три или более методов класса (функций). Методы должны быть реализованы в трех секциях: public, protected и private. В каждом методе должно быть обращение к одной или более переменной класса.
- Класс должен содержать три или более (в соответствии с вариантом задания) переменных класса. Переменные класса должны быть реализованы в трех секциях: public, protected и private.
- Класс должен содержать одну или более переменную со строковыми данными.
- В классе должны быть реализованы два метода: Input и Output. Метод Input должен выполнять ввод данных класса с клавиатуры. Метод Output должен печатать на экране данные класса. Методы должны располагаться в секции public. В производных классах (**Derived1**, **Derived2** и **Derived3**) методы должны использовать вызовы из родительского класса. Т.е. в производном классе в функциях первоначально должны быть вызваны функции базового класса (для **Derived1** и **Derived2** функции из класса **Base**, для **Derived3** функции из класса **Derived2**), а затем выполнен ввод и вывод данных текущего класса.

Индивидуальные творческие задания/проекты к лабораторной работе 13:

Необходимо реализовать один интерфейс **Interface1** и один абстрактный класс **Abstruct1**. **Interface1** должен включить описание функции Input. **Abstruct1** должен включить описание функции Output.

Расширить наследование класса **Base**, добавив к нему **Interface1** и **Abstruct1**. В каждом классе **Derived1**, **Derived2** и **Derived3**, реализовать функции Input и Output.

Индивидуальные творческие задания/проекты к лабораторной работе 14:

В производный класс **Derived3** добавляется реализация операторов в соответствии с вариантом задания.

Задание включает реализацию 2 операторов:

- Оператор, который возвращает логическое значение (истина или ложь).
- Оператор, который выполняет изменение данных класса.

Каждый оператор должен быть реализован для 3 типов параметров:

- Параметр целочисленного типа **int**.
- Параметр строкового типа **string**.
- Параметр ссылки на исходный класс **Derived3**.

В основной функции программы должен присутствовать вызов каждого оператора для всех типов параметров.

Индивидуальные творческие задания/проекты к лабораторной работе 15:

Требуется объявить 2 делегата для функций Input и Output.

- В основной функции программы необходимо создать 3-5 объектов класса **Derived1**.
- Создать 2 объекта делегата для функций Input и Output. Присвоить им адреса функций 1-го созданного объекта класса **Derived1**. Вызвать функции с использованием делегатов.
- Выполнить группирование делегатов. Для ранее созданных объектов делегатов добавить адреса функций оставшихся объектов класса **Derived1** (со 2 по 3-5). Вызвать функции с использованием делегатов.
- Выполнить разгруппирование делегатов. Удалить из объектов делегатов все адреса функций. Вызвать функции с использованием делегатов (убедиться, что ни одна функция не вызвана).

Индивидуальные творческие задания/проекты к лабораторной работе 16:

В работе требуется реализовать коллекцию элементов в соответствии с вариантом задания. В качестве элементов коллекции должен использоваться класс из лабораторной работы №1.

Необходимо сформировать коллекцию элементов, сохранить ее в файл, а затем восстановить коллекцию из файла.

- В класс **Derived1,** по необходимости, можно добавить функции для чтения/записи данных в файл.

Коллекции:

- System.Collections.ArrayList
- System.Collections.Hashtable
- System.Collections.Queue

- System.Collections.SortedList
- System.Collections.Stack
- System.Collections.Generic.SortedList
- System.Collections.Generic.SortedSet
- System.Collections.Generic.SortedDictionary
- System.Collections.Generic.LinkedList
- System.Collections.Generic.Dictionary
- System.Collections.Generic.HashSet
- System.Collections.Generic.List
- System.Collections.Generic.Queue
- System.Collections.Generic.Stack

Индивидуальные творческие задания/проекты к лабораторной работе 17:

В работе требуется реализовать программу, состоящую из DLL библиотеки и консольного приложения. Программа должна выполнять задание, указанное в лабораторной работе №1.

Консольное приложение должно реализовать набор классов, приведенный в лабораторной работе №1. Во всех функция, выполняющих ввод/вывод данных, вместо стандартных функций должны использоваться вызовы из DLL библиотеки.

DLL библиотека:

- Библиотека должны быть реализована на языке C/C++.
- В библиотеке должны быть реализованы функции ввода/вывода данных.

Консольное приложение:

- Приложение должно быть реализовано на языке C#.
- Для ввода/вывода данных приложение должно использовать вызовы DLL библиотеки.

Требования к программе:

- Для базового класса (**Base**) ввод и вывод данных должен выполняться индивидуально для каждого поля. Т.е. в DLL библиотеке должны быть реализованы функции печати и сканирования отдельно для чисел, строк и т.д. В приложении должен быть сделан импорт функций из DLL библиотеки.
- Для всех 3 производных классов (**Derived1**, **Derived2** и **Derived3**) ввод и вывод данных должен выполняться с использованием структуры. Т.е. в DLL библиотеке должны быть реализованы 3 структуры (по одной для каждого класса), которые должны использоваться для ввода/вывода данных. Для каждой структуры необходимо реализовать по одной функции ввода и вывода данных. В приложении должен быть сделан маршалинг для 3 структур и импорт функций из DLL библиотеки.

Описание показателей и критериев оценивания с указанием шкалы оценивания для очной и других форм обучения (с применением балльно-рейтинговой системы и /или без ее использования):

Оцениваются следующие показатели: соответствие требованиям задания, соответствие требованиям оформления отчета, правильность схем алгоритмов, расчетов и работы разработанной программы во время демонстрации, корректность и обоснованность выводов, самостоятельность выполненной работы.

6 баллов выставляется, если студент выполнил работы в установленный срок, правильно и полно отвечает на вопросы по каждой лабораторной работе, объясняет их на примерах, связывает с программной реализацией.

4-5 баллов выставляется, если студент отвечает на вопросы недостаточно полно или с неточностями, или не отвечает на часть заданных вопросов, не может объяснить их на примере, есть недочеты в лабораторной работе.

0-3 *балла выставляется, если* студент не отвечает на вопросы, не может объяснить их на примере, лабораторная работа выполнена некорректно.

Оформление сведений о дополнениях и изменениях, внесенных в ФОС дисциплины

Сведения о дополнениях и изменениях, внесенных в ФОС дисциплины

Учебный год	Решение кафедры (№ протокола, дата)	Внесенные в ФОС дополнения и изменения	Подпись заведующего кафедрой