Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Новосибирский государственный технический университет»
Кафедра автоматизированных систем управления
Кафедра автоматики
Кафедра вычислительной техники
Кафедра защиты информации

Паспорт
курсовой работы

по дисциплине «Программирование», 3 семестр

Методика оценки.

Выполнение курсовой работы (далее – КР) является обязательным видом самостоятельной работы студента по дисциплине, предусмотренным учебным планом.

Основной целью выполнения КР является формирование компетенций и соотнесенных с ними индикаторов по дисциплине «Программирование» в 3 семестре посредством закрепления, углубления и обобщения знаний, полученных студентами за время теоретического обучения и прохождения практик, а также выработка навыков самостоятельного применения знаний и навыков для творческого решения конкретных задач. Выполнение курсовой работы должно способствовать подготовке их к решению более сложной задачи - выполнению выпускной квалификационной работы.

Задачами выполнения курсовой работы являются: разработка алгоритмов и объектной модели профессиональной задачи; обоснованный выбор языка, структур данных и инструментальных средств; реализация классов, библиотек и при необходимости многопоточных вычислений; тестирование, отладка, оценка трудоёмкости и соблюдение норм информационной безопасности. Работа направлена на формирование индикаторов ОПК-2.1, ОПК-2.2, ОПК-7.1, ОПК-7.2 и ОПК-7.3.

Темы КР соответствуют профилю подготовки, формируются преподавателями в начале семестра и утверждаются решением кафедры. Количество тем и индивидуальных наборов исходных данных обеспечивает каждого обучающегося индивидуальным заданием.

Выполнение студентами КР начинается с ознакомления с примерной тематикой. Закрепление тем КР за студентами и назначение научных руководителей производится распоряжением заведующего кафедрой и/или утверждается решением кафедры.

Курсовая работа выполняется индивидуально.

Структура курсовой работы:

Титульный лист.
Введение: актуальность, цель, задачи, требования и исходные данные.
Теоретическая часть: анализ предметной области, методов, архитектуры, алгоритмов и программных средств.
Практическая часть: объектная модель, реализация, тестирование, отладка и оценка эффективности.
Выводы или заключение.
Список литературы.
Приложения (при необходимости).

Требования к оформлению:

Объем КР до 50 страниц машинописного текста формата А4. Шрифт Times New Roman, 12. Нумерация страниц сквозная, в нижней части листа по центру арабскими цифрами. КР должна быть отредактирована, не содержать орфографических, синтаксических и стилевых ошибок.

Законченная курсовая работа предоставляется для проверки в электронном виде в срок, установленный преподавателем. Преподаватель оценивает качество КР с учетом теоретического и практического содержания, достижения ее целей и задач.

Курсовая работа проверяется руководителем работы, который дает письменное заключение по работе — рецензию.

Если при выполнении КР были допущены ошибки, то работа возвращается студенту для исправления выявленных недочетов и затем вновь предоставляется руководителю для проверки. При положительном результате оценивания студент распечатывает работу, передает на кафедру и защищает до сессии в назначенное преподавателем время.

Защита КР проходит публично перед группой студентов.

По результатам защиты студенту выставляется оценка в соответствии с критериями, приведенными в п. 2 настоящего Паспорта.

Уровни сформированности компетенций и критерии оценки

В соответствии с балльно-рейтинговой системой НГТУ курсовая работа по дисциплине «Программирование» в 3 семестре имеет максимальную оценку 100 баллов.

Курсовая работа выполнена на продвинутом уровне, если:

- она выполнена в полном соответствии с заданием, отличается глубиной проработки всех разделов содержательной части, текстовая часть оформлена с соблюдением установленных правил;
- руководитель характеризует деятельность студента положительно (в частности, отмечает его инициативу, самостоятельность, систематичность работы на всех этапах выполнения работы);
- в докладе исчерпывающе, последовательно, четко и логически правильно изложена суть работы и ее основные результаты;
- студент свободно владеет теоретическим материалом, безошибочно применяет его при решении задач, сформулированных в задании;
- в докладе суть работы и ее основные результаты представлены исчерпывающе, последовательно, четко и логически правильно; на все вопросы студент дал обстоятельные и аргументированные ответы, убедительно защищал свою точку зрения;
- компетенции и соотнесенные с ними индикаторы, закрепленные за дисциплиной, по которой выполняется курсовая работа, сформированы в полном объеме.

Оценка за выполнение КР составляет от 87 до 100 баллов.

Курсовая работа выполнена на базовом уровне, если:

- соответствует заданию, отличается глубиной проработки всех разделов содержательной части, текстовая часть оформлена с соблюдением установленных правил;
- руководитель характеризует деятельность студента положительно, но с незначительными замечаниями;
- в докладе правильно изложена суть работы и ее основные результаты;
- студент достаточно твердо усвоил теоретический материал и может самостоятельно его применять;
- в докладе суть работы и ее основные результаты представлены полно; на все вопросы студент дал ответы, но их полнота и аргументированность недостаточны;
- компетенции и соотнесенные с ними индикаторы, закрепленные за дисциплиной, по которой выполняется курсовая работа, сформированы с небольшими пробелами и соответствуют базовому уровню.

Оценка за выполнение КР составляет от 73 до 86 баллов.

Курсовая работа выполнена на пороговом уровне, если:
выполнена в основном правильно, но без необходимой проработки некоторых разделов;
в докладе упущены некоторые принципиальные моменты содержательной части работы;
в докладе представлены суть работы и ее основные результаты; ответы на вопросы вызвали существенные затруднения;
компетенции и соотнесенные с ними индикаторы, закрепленные за дисциплиной, по которой выполняется курсовая работа, сформированы с пробелами и соответствуют пороговому уровню.
Оценка за выполнение КР составляет от 50 до 72 баллов.

Курсовая работа считается невыполненной, если студентом не проработаны важные разделы исследования, допущены принципиальные ошибки, не исправленные после замечаний руководителя КР. Студент не допущен к защите курсовой работы. Компетенции и соотнесенные с ними индикаторы, закрепленные за дисциплиной, по которой выполняется курсовая работа, не сформированы.
Оценка составляет менее 50 баллов.

Шкала оценки.

Курсовая работа по дисциплине считается успешно выполненной, если сумма полученных баллов составляет от 50 до 100 баллов включительно.

Примерный перечень тем курсовой работы

1. Объектно-ориентированный каталог программных компонентов. Спроектировать иерархию классов, поиск и хранение записей, интерфейс библиотеки и сериализацию; применить паттерн «Фабрика» и проверить инварианты.
2. Библиотека геометрических фигур. Реализовать полиморфную иерархию, операции вычисления характеристик, умные указатели и сохранение данных; проверить корректное разрушение объектов.
3. Шаблонный динамический контейнер. Реализовать аналог vector с итераторами, копированием и перемещением; исследовать сложность, исключительные ситуации и работу с некопируемыми типами.
4. Библиотека обработки строк. Разработать классы для поиска, сравнения, разбиения и преобразования строк; предоставить статическую и динамическую сборки и набор тестов.
5. Многопоточный анализ текстовых файлов. Разделить обработку на задания, собрать частоты и статистики с async/future; проверить отсутствие гонок и сравнить время с последовательной версией.
6. Потокобезопасная очередь заданий. Реализовать контейнер и модель производителей и потребителей; обосновать выбор примитивов синхронизации и проверить корректное завершение потоков.
7. Объектно-ориентированный обработчик синтетических журналов. Построить классы чтения, фильтрации и представления записей, фабрику обработчиков и параллельную обработку файлов.
8. Библиотека матричных вычислений. Реализовать класс матрицы, перегруженные операции, копирование и перемещение; сравнить последовательное и параллельное выполнение вычислений.
9. Конструктор структурированных отчётов. Применить паттерн «Строитель», выделить интерфейсы источников и представления данных, проверить расширение без изменения существующего клиентского кода.

10. Система учёта оборудования. Спроектировать иерархию объектов, операции поиска и выдачи, хранение в стандартных контейнерах и сериализацию; обеспечить инкапсуляцию и тестирование ошибок.
11. Симулятор обслуживания очереди. Представить участников и события классами, реализовать создание через фабрику и запуск независимых экспериментов с async/future; проверить воспроизводимость.
12. Менеджер конфигурации приложения. Исследовать применимость Singleton, безопасную инициализацию, валидацию параметров и многопоточный доступ; сравнить с явной передачей зависимости.
13. Библиотека преобразования форматов данных. Создать единый интерфейс сериализаторов JSON, XML и бинарного формата, фабрику реализаций и проверки недоверенных входных данных.
14. Система обработки графических примитивов. Реализовать классы объектов, полиморфные операции и создание сложных объектов через «Строитель»; использовать умные указатели и контейнеры.
15. Многопоточный поиск в коллекции записей. Разработать классы коллекции и стратегии поиска, разделить диапазоны итераторов, объединить результаты и исследовать границы полезного параллелизма.
16. Сравнение стратегий синхронизации. Реализовать учебную модель общего счётчика и очереди с согласованными примитивами; проверить корректность, взаимные блокировки и результаты замеров времени.
17. Библиотека владения ресурсами. Реализовать RAII-обёртки для файлов и буферов, некопируемые и перемещаемые типы; проверить время жизни, исключения и интеграцию с контейнерами.
18. Обобщённая система преобразования последовательностей. Применить шаблоны классов, итераторы и move-итераторы; продемонстрировать передачу владения и обработку элементов разных типов.
19. Модульная система экспорта данных. Создать стабильный интерфейс, статическую и динамическую библиотеки с несколькими форматами; проверить совместимость, обработку ошибок и загрузку компонентов.
20. Рефакторинг учебной объектной модели по SOLID. Выбрать предметную область, выявить нарушения принципов, переработать классы с фабрикой или строителем и доказать сохранение поведения тестами.

Примерный перечень вопросов к защите курсовой работы

21. Чем обусловлены выбор темы, актуальность и практическое назначение разработанной программы?
22. Как сформулированы требования, ограничения, входные и выходные данные вашей работы?
23. Какие принципы и программные средства, в том числе отечественного производства, рассматривались и почему выбраны используемые инструменты?
24. Как выполнена декомпозиция программы и распределена ответственность между классами и модулями?
25. Какие инварианты классов поддерживаются и как выбранные уровни доступа обеспечивают инкапсуляцию?
26. Каков порядок конструирования и разрушения объектов в вашей программе?
27. Где применяются наследование, композиция и полиморфизм? Почему выбраны именно они?

28. Как обеспечено корректное уничтожение объектов через интерфейс базового класса?
29. Какие операции перегружены и почему их семантика соответствует назначению типа?
30. Как реализованы копирование, присваивание и самоприсваивание объектов, владеющих ресурсами?
31. Где используется перемещение и в каком состоянии остаётся перемещённый объект?
32. Почему некоторые типы сделаны некопируемыми и как передаётся владение ими?
33. Как выбранные умные указатели отражают владение и исключают утечки или циклические ссылки?
34. Как реализованы шаблоны, итераторы и диапазоны? Какие действия могут сделать итератор недействительным?
35. Каковы размер, вместимость и амортизированная сложность операций вашего контейнера?
36. Какие операции string и преобразования данных применены? Как проверяются ошибки ввода?
37. Какие паттерны использованы, какие проблемы они решают и какие альтернативы были рассмотрены?
38. Какие принципы SOLID применены к вашей объектной модели и как проверяется возможность её расширения?
39. Если применён Singleton, как обеспечены безопасная инициализация и корректное время жизни? Чем его можно заменить?
40. Как организованы процессы и потоки, передача данных и завершение параллельной работы?
41. Где потенциально возникают гонки данных и какими средствами они предотвращены?
42. Почему выбраны используемые примитивы синхронизации? Как предотвращается взаимная блокировка?
43. Как применены async/future и как передаются результаты и исключения между потоками?
44. Как измерено время выполнения и обеспечена сопоставимость последовательной и параллельной версий?
45. Как создаются и подключаются статическая и динамическая библиотеки? Что входит в их публичный интерфейс?
46. Как устроен формат сериализации и какие проверки выполняются при чтении повреждённых или недоверенных данных?
47. Когда допустимы дружественные функции и как их применение влияет на безопасность и инкапсуляцию?
48. Какие тесты подтверждают корректность алгоритмов, граничных случаев и обработки ошибок?
49. Какие возможности интегрированной среды использовались для отладки и поиска ошибок памяти или многопоточности?
50. Каковы ограничения полученного результата, возможности его применения и направления дальнейшего улучшения?