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

Паспорт экзамена

по дисциплине «Программирование», 3 семестр

Методика оценки

Экзамен проводится в устной форме по билетам. Билет состоит из 2 вопросов и формируется по следующему правилу:
первый вопрос выбирается из диапазона вопросов 1–22;
второй вопрос выбирается из диапазона вопросов 23–44.

Таким образом, проверяются результаты обучения по дисциплине, соотнесённые с установленными в программе индикаторами достижения компетенций ОПК-2.1, ОПК-2.2, ОПК-7.1, ОПК-7.2 и ОПК-7.3.

На экзамене преподаватель вправе задавать студенту уточняющие и дополнительные вопросы из общего перечня (п. 4).

Форма экзаменационного билета

НОВОСИБИРСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ
Факультет АВТФ

Билет № _____
к экзамену по дисциплине «Программирование»

---

1. Вопрос 1
2. Вопрос 2.

Утверждаю: зав. кафедрой ____ _____ должность, ФИО
(подпись)
(дата)

Критерии оценки результатов обучения, соотнесенных с уровнями освоения индикаторов достижения компетенций

Ответ на экзаменационный билет засчитывается на **продвинутом** уровне, если студент проводит сравнительный комплексный анализ материала, выявляет проблемы, предлагает механизмы их решения, представляет количественные характеристики определенных процессов, приводит конкретные примеры, не допускает ошибок. Совокупность результатов обучения по дисциплине и соотнесенных с ними индикаторов достижения компетенций не содержит пробелов. Установленные в программе компетенции сформированы в полном объеме. Оценка составляет *от 35 до 40 баллов.*

Ответ на экзаменационный билет засчитывается на **базовом** уровне, если студент при ответе на вопросы формулирует основные понятия, дает характеристику процессов, явлений, проводит анализ причин, условий, способен представить количественные и качественные характеристики процессов, не допускает существенных ошибок. Совокупность результатов обучения по дисциплине и соотнесенных с ними индикаторов достижения компетенций содержит несущественные пробелы. Установленные в программе компетенции сформированы на базовом уровне. Оценка составляет *от 27 до 34 баллов.*

Ответ на экзаменационный билет засчитывается на **пороговом** уровне, если студент при ответе на вопросы дает определение основных понятий, может показать причинно-следственные связи явлений, допускает непринципиальные ошибки. Совокупность результатов обучения по дисциплине и соотнесенных с ними индикаторов достижения компетенций содержит пробелы. Установленные в программе компетенции сформированы на пороговом уровне. Оценка составляет *от 20 до 26 баллов.*

Ответ на экзаменационный билет считается **неудовлетворительным** (ниже порогового уровня), если студент при ответе на вопросы не дает определений основных понятий, не способен показать причинно-следственные связи явлений, допускает принципиальные ошибки. Совокупность результатов обучения по дисциплине и соотнесенных с ними индикаторов достижения компетенций содержит существенные пробелы. Установленные в программе компетенции не сформированы. Оценка составляет *менее* 20 *баллов.*

Шкала оценки

Экзамен считается сданным, если сумма баллов по всем заданиям билета составляет от 20 до 40 баллов включительно. Сумма менее 20 баллов признается неудовлетворительным результатом промежуточной аттестации по дисциплине.

В общей оценке по дисциплине экзаменационные баллы учитываются в соответствии с правилами балльно-рейтинговой системы, установленными в НГТУ.

Вопросы к экзамену **по дисциплине «**Программирование**»**

1. Класс и объект в C++: состояние, поведение, инкапсуляция и отличие класса от структуры.
2. Секции public, protected и private. Проектирование интерфейса класса и защита инвариантов.
3. Методы класса, указатель this, константные методы и константные объекты.
4. Конструкторы класса: назначение, виды и правила вызова. Списки инициализации членов.
5. Порядок конструирования базовых классов и полей. Инициализация константных полей и ссылок.

6. Деструктор, время жизни объекта и порядок разрушения. Управление ресурсами по принципу RAII.
7. Конструктор копирования и копирующий оператор присваивания. Поверхностное и глубокое копирование.
8. Перегрузка операций: ограничения, выбор свободной функции или метода, реализация операций для пользовательского контейнера.
9. Наследование: базовый и производный классы, виды наследования и доступ к членам.
10. Полиморфизм, виртуальные функции и абстрактные классы. Виртуальный деструктор и срезка объектов.
11. Умные указатели unique_ptr, shared_ptr и weak_ptr: владение, время жизни и предотвращение циклических ссылок.
12. Класс string: конструирование, присваивание, доступ к символам и управление строковыми данными.
13. Операции string: объединение, извлечение подстроки, вставка, удаление, поиск и сравнение.
14. Преобразования строк и чисел. Проверка входных данных и обработка неудачного преобразования.
15. Итераторы: назначение, категории, базовые операции и представление диапазона.
16. Арифметика итераторов и её ограничения. Различия работы с последовательными и ассоциативными контейнерами.
17. Передача итераторов в функции и стандартные алгоритмы. Предикаты, границы диапазона и инвалидирование итераторов.
18. Шаблоны классов: параметры, создание экземпляров и организация кода обобщённого класса.
19. Реализация шаблонного аналога vector: размер и вместимость, выделение памяти, добавление и удаление элементов.
20. Копирование и перемещение объектов. Lvalue, rvalue и назначение std::move.
21. Конструктор перемещения и перемещающий оператор присваивания. Корректное состояние перемещённого объекта.
22. Правило трёх, пяти и нуля при проектировании класса, управляющего ресурсом.
23. Move-итераторы и перемещение в стандартной библиотеке. Работа с контейнерами некопируемых типов.
24. Некопируемые типы: запрет копирования, передача владения и практические случаи применения.
25. Многопоточность: области применения, преимущества, накладные расходы и ограничения ускорения.
26. Процесс и поток. Конкурентность и параллелизм, разделяемая память и изоляция.
27. Создание и завершение потоков. Передача аргументов, ожидание завершения и обеспечение времени жизни данных.
28. Состояние гонки и гонка данных. Воспроизводимость, обнаружение и предотвращение ошибок совместного доступа.
29. Взаимное исключение: mutex, критическая секция, блокировки и безопасное освобождение блокировки.
30. Семафор и барьер: назначение, различия и примеры координации потоков.

31. SpinLock, SpinWait и Monitor. Сравнение активного ожидания и блокировки; доступность примитивов в разных средах программирования.
32. Примитивы Semaphore и SemaphoreSlim: ограничения, различия и условия выбора в среде, где они предоставляются.
33. Асинхронные вычисления с async и future: получение результата, ожидание и передача исключений.
34. Взаимная блокировка потоков: необходимые условия возникновения и способы предотвращения.
35. Измерение времени выполнения: монотонные часы и таймер StopWatch; воспроизводимость и сопоставимость замеров.
36. Статические и динамические библиотеки: создание, компоновка, загрузка и механизм использования.
37. Проектирование и проверка интерфейса библиотеки. Совместимость типов, владение ресурсами и обработка ошибок.
38. Паттерн «Строитель»: назначение, участники, этапы построения объекта и примеры применения.
39. Паттерн «Фабрика»: отделение создания объектов от использования, реализация и выбор конкретного типа.
40. Принципы SOLID: содержание, применение к иерархии классов и ограничения формального следования принципам.
41. Паттерн Singleton: назначение, ограничения, управление временем жизни и безопасная инициализация в многопоточной программе.
42. Безопасность дружественных функций и сериализации/десериализации при проектировании классов и библиотек.
43. Обоснование выбора языка, библиотек и интегрированной среды разработки, в том числе отечественных средств, для объектно-ориентированной программы.
44. Тестирование и отладка классов, контейнеров и многопоточных программ. Проверка граничных случаев, владения ресурсами и соблюдения норм информационной безопасности.