Министерство науки и высшего образования Российской Федерации
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ (НИ ТГУ)

Физический факультет

УТВЕРЖДЕНО:
Декан физического факультета
С.Н. Филимонов

Оценочные материалы по дисциплине

Языки программирования высокого уровня

по направлению подготовки

03.03.02 Физика

Направленность (профиль) подготовки:
«Фундаментальная и прикладная физика»

Форма обучения
Очная

Квалификация
Бакалавр

Год приема
2025

СОГЛАСОВАНО:
Руководитель ОП
С.Н. Филимонов

Председатель УМК
О.М. Сюсина

Томск – 2025

1. Компетенции и индикаторы их достижения, проверяемые данными оценочными материалами

Целью освоения дисциплины является формирование следующих компетенций:

– ОПК-3 – Способен понимать принципы работы современных информационных технологий и использовать их для решения задач профессиональной деятельности;

– ПК-1 – Способен проводить научные исследования в выбранной области с использованием современных экспериментальных и теоретических методов, а также информационных технологий.

Результатами освоения дисциплины являются следующие индикаторы достижения компетенций:

– ИОПК-3.1. – Владеет навыками работы с компьютером и компьютерными сетями с целью получения, хранения, обработки и анализа научной информации;

– ИОПК-3.2. – Соблюдает основные требования информационной безопасности при решении задач профессиональной деятельности данных;

– ИПК-1.2. – Владеет практическими навыками использования современных методов исследования в выбранной области.

2. Оценочные материалы текущего контроля и критерии оценивания

Элементы текущего контроля:

– индивидуальные практические задания;

– групповые домашние задания.

Для оценки усвоения синтаксических основ языка С, освоения основных методов и алгоритмов программирования, студентам предлагаются практические задания (ИОПК-3.1, ИПК-1.2).

Выполнение практических заданий предполагает не только создание выдающего корректные результаты кода, но и обоснование выбранного алгоритма, оценку его эффективности, проверку корректности входных данных.

Примеры практических индивидуальных заданий по темам курса.

Тема 1. Базовые средства и операторы языка С.

1. Перераспределить значения переменных x и y так, чтобы в x оказалось большее из этих значений, а в y – меньшее.
2. Верно ли высказывание, что ладья, расположенная на поле (x1, y1) шахматной доски, «бьет» поле (x2, y2) (x1, y1, x2, y2 – целые от 1 до 8). Для справки: ладья ходит по вертикали и по горизонтали.
3. Верно ли высказывание, что сумма двух первых цифр заданного 4-хзначного числа равна сумме двух его последних цифр.
4. Значения переменных a, b и c поменять местами так, чтобы оказалось $a \geq b \geq c$.
5. Найти K – количество различных делителей числа Z.
6. Найти знакочередующуюся сумму цифр заданного натурального числа. Например, 123 –> 1-2+3=2.
7. Вычислить $S = \sum_{k=1}^{n} k^3 \sum_{t=1}^{m} (k-t)^2$.
8. Даны действительные числа a, b. Получить u = min(a, b), v = min (ab, a + b), w = min (u , v^2). (написать функцию нахождения минимума 2-х чисел).

Тема 2. Массивы.

1. Выяснить, имеется ли в массиве A два идущих подряд элемента одного знака (положительных или отрицательных).

2. Найти в массиве самую длинную возрастающую последовательность, расположенную после максимального элемента. Вывести на экран номера ее первого и последнего элементов.

3. Дана целочисленная матрица размера m×n. Определить, упорядочены ли по убыванию элементы k-ой строки.

4. Сортировка строк матрицы по возрастанию первых элементов.

5. В заданной строке заменить все латинские буквы на 'A', цифры на '0', остальные знаки на '?'.

6. Напечатать все слова данного предложения, не содержащие буквы 'a'.

Тема 3. Работа с файлами средствами языка С.

Создать файл из 20 случайных положительных чисел, записанных строго по возрастанию. Выбрать из него во второй файл все четные числа. Вывести оба файла на консоль.

Тема 4. Структуры. Список и стек.

1. Определить структуру, содержащую информацию о студенте:

struct STUD {char name[40]; int group, mark[5];};

Задать массив STUD kurs[10], инициализировав его данными из файла (файл с данными data.txt создайте сами). Вывести список студентов, имеющих только «2».

2. Сформируйте однонаправленный список (с фиктивной головой) из n целых случайных чисел диапазона от 0 до 20 (число n вводится с клавиатуры, тип списка указан в задаче). Добавьте в список заданный элемент после первого такого же элемента. В случае, если элемент в списке не присутствует, добавьте его в хвост.

Результаты выполнения практических заданий по темам 1-4 оцениваются «отлично», «хорошо» или «удовлетворительно».

Оценка «отлично» выставляется, если программа выдает верный ответ, студент выбрал оптимальный алгоритм решения, проверил корректность входных данных, код программы оптимален.

Оценка «хорошо» выставляется, если программа выдает верный ответ, студент выбрал оптимальный алгоритм решения, но не проверил корректность входных данных или код программы не оптимален.

Оценка «удовлетворительно» выставляется, если программа выдает верный ответ, но студент выбрал не оптимальный алгоритм решения, не проверил корректность входных данных.

Для каждого задания устанавливается срок выполнения. Задания, сданные позже установленного срока без уважительной причины, оцениваются на балл ниже реальной оценки.

Для оценки усвоения основ C++ и приобретения навыка объектно-ориентированного программирования, студентам предлагается реализовать следующие классы. (ИОПК-3.1, ИОПК-3.2, ИПК-1.2).

1. Простой класс: дробь, круг, прямоугольник, прямоугольный треугольник, свободный вектор, прямая на плоскости, плоскость в пространстве, точка, отрезок, время, угол (индивидуальное практическое задание).
2. Класс «Массив».
3. Класс-шаблон «Массив».
4. Агрегированный класс «Двунаправленный список»
5. Класс «Множество», наследник класса «Двунаправленный список»

Результаты реализации классов оцениваются «отлично», «хорошо» или «удовлетворительно».

Оценка «отлично» выставляется, если студент реализовал все методы класса; учел все исключения; допускается реализация отдельных методов не самым оптимальным образом.

Оценка «хорошо» выставляется, если студент реализовал 90% методов или реализованы все методы, но для большей части методов использованы не самые эффективные алгоритмы. Возможна потеря некоторых исключений.

Оценка «удовлетворительно» выставляется, если студент реализовал не менее 70% методов класса или реализованы все методы, но для большей части реализация построена на неэффективных алгоритмах, либо не учитывает появление исключений;

Для каждого задания устанавливается срок выполнения. Задания, сданные позже установленного срока без уважительной причины, оцениваются на балл ниже реальной оценки.

3. Оценочные материалы итогового контроля (промежуточной аттестации) и критерии оценивания

Зачет в первом семестре проводится в письменной форме по билетам. Билет содержит теоретический вопрос и практическое задание. Продолжительность зачета 1 час.

Ответы на теоретические вопросы даются в развернутой форме и проверяют ИОПК-3.1.

Практическое задание предполагает написание программного кода для поставленной задачи и анализ его работы. Оценивается оптимальность выбранного для решения задачи алгоритма и скорость его работы. Это задание проверяет ИПК-1.2.

Примерный перечень теоретических вопросов.

1. Базовые типы переменных в языке С, их объявление, инициализация. Неявное преобразование типов.
2. Основные арифметические и логические операции в языке С. Сокращенная форма записи арифметических операций. Операции инкремента и декремента, разница между их постфиксной и префиксной формой.
3. Форматированный ввод и вывод информации.
4. Условный оператор: краткая и полная форма, принцип работы.
5. Операторы цикла в С: цикл с предусловием, цикл с постусловием, параметрический цикл. Схема работы каждого оператора цикла. Условие работы цикла.
6. Функции в С: определение, системные и собственные функции. Как определяется функция. Вызов функции в программе. Упреждающее объявление функции.
7. Одномерный массив: определение, объявление массива фиксированного размера, нумерация элементов массива, размещение элементов массива в памяти. Инициализация элементов массива. Обращение к элементам массива. Массивы как аргументы функции.
8. Указатели: определение, объявление, операции для работы с указателем. Связь массивов и указателей. Одномерные динамические массивы: объявление, выделение памяти, расположение в памяти компьютера, инициализация, освобождение памяти. Понятие утечки памяти.
9. Понятие многомерного массива. Матрица, как пример двумерного массива: объявление матрицы фиксированного размера, нумерация элементов матрицы, размещение элементов матрицы в памяти. Инициализация элементов матрицы. Обращение к элементам матрицы. Матрицы фиксированного размера как аргументы функции.
10. Двумерные динамические массивы: объявление, выделение памяти, расположение в памяти компьютера, инициализация, освобождение памяти. Динамические матрицы как аргументы функции.

11. Тип char: назначение, инициализация переменных указанного типа. Кодировка символов и тип char как разновидность типа int.

12. Строка: определение, инициализация, объем занимаемой памяти, обращение к элементам строки. Базовые функции для работы со строками.

13. Файлы: схема работы с файлами, базовые функции языка С, обеспечивающие работу с файлами.

14. Структуры: понятие структуры, основные принципы работы со структурными типами данных.

15. Список, как форма организации данных. Виды списков. Реализация списка с помощью структуры.

16. Стек, как форма организации данных. Реализация стека на основе списка.

Примеры практических заданий.

1. Даны 3 произвольных числа. Можно ли построить треугольник с такими длинами сторон, и, если можно, то какой (равносторонний, равнобедренный, обычный).

2. Найти K – количество различных делителей числа Z.

3. Вычислить $S = \sum_{i=1}^{n} \prod_{j=1}^{m} \frac{1}{i^2 + j^2}$.

4. Выяснить, имеется ли в массиве С два идущих подряд нулевых элемента.

5. Найти в массиве самую длинную возрастающую последовательность, расположенную после максимального элемента. Вывести на экран номера ее первого и последнего элементов.

6. Найти сумму максимумов всех строк матрицы.

7. Удалить в матрице строки с нечетными номерами.

8. Дана строка s1, ,sn. Группы символов, разделенные пробелами (одним или несколькими) и не содержащие пробелов внутри себя, будем называть словами. Подсчитать количество слов в данной строке.

9. Дана строка s1,…,sn. Группы символов, разделенные пробелами (одним или несколькими) и не содержащие пробелов внутри себя, будем называть словами. Найти количество слов, у которых первый и последний символы совпадают между собой.

10. Создать файл из 20 случайных чисел разного знака. Выбрать из него во второй файл все положительные числа. Вывести оба файла на консоль.

11. Определить структуру, содержащую информацию о студенте:

struct STUD {char name[20]; int group, mark[5];};

Задать массив STUD kurs[10], инициализировав его данными из файла (файл с данными data.txt создайте сами). Вывести список студентов, не имеющих аттестацию хотя бы по одному предмету.

12. Сформируйте однонаправленный список (с фиктивной головой) из n целых случайных чисел диапазона от 0 до 20 (число n вводится с клавиатуры). Добавьте в список заданный элемент после первого такого же элемента. В случае, если элемент в списке не присутствует, добавьте его в хвост.

Результаты зачета определяются оценками «зачтено» или «не зачтено».

Оценка «**зачтено**» выставляется, если:

а) студент выполнил не менее 75% заданий текущего контроля;

б) ответ студента на теоретические вопросы полный или имеются незначительные замечания;

в) код практического задания верен или содержит ошибки синтаксического характера.

Оценка «**не зачтено**» выставляется, если выполнено одно из следующих условий:

а) студент выполнил менее 75% заданий текущего контроля;

б) ответ студента на теоретические вопросы не полный и содержит серьезные ошибки;

в) код практического задания содержит синтаксические и алгоритмические ошибки.

Если в течение семестра студент посетил не менее 75% занятий и выполнил все практические задания, то он освобождается от выполнения практической части билета.

Экзамен во втором семестре проводится в письменной форме по билетам. Экзаменационный билет состоит из двух частей. Продолжительность экзамена 1,5 часа.

Первая часть представляет собой два теоретических вопроса. Ответы на вопросы даются в развернутой форме и проверяют ИОПК-3.1.

Вторая часть состоит из двух практических заданий и проверяет ИОПК-3.2 и ИПК-1.2. Ответы на вопросы второй части предполагают написание программного кода для поставленной задачи и анализ его работы. Оценивается оптимальность выбранного для решения задачи алгоритма и скорость его работы.

Примерный перечень теоретических вопросов.

1. Принципы объектно-ориентированного программирования. Определение класса. Сокрытие информации. Объект. Что за операция :: ? Чем член-функции отличаются от обычных? Какие существуют типы доступа и чем они отличаются друг от друга? Чем отличаются функции, определенные внутри класса, от функций, определенных вне класса?
2. Конструкторы и деструкторы. Назначение конструктора. Особенности конструктора. Конструктор копирования. Деструктор. Особенности деструктора. В каком случае необходим конструктор копирования и почему? Чем конструктор копирования отличается от перегрузки операции =?
3. Неявный указатель this. Перегрузка операций. Перечислите правила перегрузки операций. Перегрузки операций + и +=. Чем отличается операция + от операции +=?
4. Перегрузка операций. Перечислите правила перегрузки операций. Перегрузка операций [ ], ( ).
5. Перегрузка операций. Перечислите правила перегрузки операций. Перегрузка операции =. Чем конструктор копирования отличается от перегрузки операции =?
6. Конструктор перемещения. Перегрузка оператора присвоения при перемещении: когда необходимы, когда работают. Правило пяти
7. Дружественность. Что может быть другом класса? Перегрузка операций потокового ввода >> и вывода <<. Что общего и в чем разница между перегрузкой операции потокового вывода << и перегрузкой операции потокового ввода >>? Можно ли перегрузить операции потокового ввода/вывода как методы класса, ответ аргументировать. Обязательно ли использовать ссылку при перегрузке операций потокового ввода/вывода?
8. Массивы объектов. Какие конструкторы можно использовать и как? Можно ли явно инициализировать массив объектов, определенных в динамической памяти? Как работает конструктор для массива объектов? Как работает деструктор для массива объектов?
9. Функции-шаблоны и классы-шаблоны. Что такое порожденная функция? Использование функций-шаблонов и классов-шаблонов. Описать работу компилятора.
10. Агрегированные классы. Каким образом можно определить конструктор агрегированного класса? Каким образом агрегированный класс может использовать член-данные используемого класса из части private?
11. Статические член-данные и методы класса: объявление, определение, особенности

12. Базовый и порожденный классы. Тип доступа protected. Типы наследования public и private. Ограничения наследования.

13. Простое наследование. Принцип доминирования в иерархии наследования.

14. Конструктор порожденного класса. Его вид. Стандартные преобразования при наследовании.

15. Множественное наследование. Прямые и не прямые базовые классы. Виртуальный базовый класс. Особенности инициализации его ч/данных.

16. Полиморфизм. Раннее и позднее связывание. Механизм позднего связывания при наличии виртуальных функций. Примеры.

17. Полиморфизм: виртуальные функции. Чистые виртуальные функции и абстрактный базовый класс

18. Правила определения виртуальных функций. Примеры.

Примеры практических заданий.

1. Написать следующие методы класса Array (Массив): конструктор без аргументов, конструкторы с аргументами, конструктор копирования, деструктор; перегрузка операторов << (вывод), = (присвоение.).

2. Написать следующие методы класса String: конструкторы, деструктор, перегрузка операций >> (потоковый ввод), << (потоковый вывод), = (присвоение), [] , = = (сравнение).

3. Написать следующие методы класса Polinom: конструкторы (все), перегрузка оператора присвоения, перегрузка операторов *(число) и *=(число).

4. Написать следующие методы класса List (список): конструктор по умолчанию, с аргументом (любой), конструктор копирования; перегрузка оператора << (вывод).

5. Написать следующие методы класса SET (множество): конструктор копирования, перегрузка оператора - (удаление элемента), оператора += (объединение).

Результаты экзамена определяются оценками «отлично», «хорошо», «удовлетворительно», «неудовлетворительно».

Оценка «**отлично**» выставляется, если:

а) студент дал полный и развернутый ответ на теоретические вопросы;

б) код практического задания верен, оптимален (по скорости или по объему памяти), легко читаем, при написании кода использованы эффективные алгоритмы.

Оценка «**хорошо**» выставляется, если:

а) ответ студента на теоретические вопросы в целом полный, но имеются незначительные замечания;

б) код практического задания верен, но не оптимален (по скорости или по объему памяти), при написании кода использованы трудоемкие алгоритмы.

Оценка «**удовлетворительно**» выставляется, если:

а) ответ студента на теоретические вопросы не полный;

б) код практического задания содержит ошибки синтаксического характера.

Оценка «**неудовлетворительно**» выставляется, если выполнено одно из следующих условий:

а) студент выполнил не менее 75% заданий текущего контроля;

б) ответ студента на теоретические вопросы не полный и содержит серьезные ошибки;

в) код практического задания содержит серьезные синтаксические и алгоритмические ошибки.

Если в течение семестра студент посетил не менее 75% занятий и выполнил все практические задания, то он освобождается от выполнения практической части билета.

4. Оценочные материалы для проверки остаточных знаний (сформированности компетенций)

Перечень теоретических вопросов. (ИОПК-3.1, ИОПК-3.2)

1. Принципы объектно-ориентированного программирования.

Ответ должен содержать перечисление основных принципов ООП и их краткую характеристику.

2. Какие существуют типы доступа.

Ответ должен содержать перечисление типов доступа и их определения.

3. Назначение конструктора. Особенности конструктора. Конструктор копирования. В каком случае необходим конструктор копирования.

Ответ должен содержать определение конструктора, особенности конструктора, информацию о том, какие конструкторы могут быть в классе, когда в классе необходим конструктор копирование, сигнатура конструктора копирования, назначение конструктора копирования, когда работает конструктор копирования.

4. Деструктор. Особенности деструктора.

Ответ должен содержать определение деструктора, особенности деструктора.

5. Перегрузка операций. Основные правила перегрузки операций.

Ответ должен содержать определение перегрузки операций, как объявляется перегрузка операций, как используется знак перегруженной операции, базовые правила перегрузки операций, ограничения перегрузки операций.

6. Перегрузка операции =. Когда ее необходимо выполнять?

В ответе следует указать, когда и почему необходимо выполнять перегрузку операции =, подчеркнуть необходимость обязательной проверки условия присвоения объекта самому себе.

7. Дружественность.

В ответе должно быть объяснено, что такое дружественность, кто может быть другом класса, как объявляется дружественная функция и класс, почему перегрузить потоковый ввод/вывод можно только как дружественные методы.

8. Функции-шаблоны и классы-шаблоны.

В ответе следует дать определение функции-шаблона и класса-шаблона, указать их назначение, когда происходить конкретизация шаблона компилятором.

9. Агрегированные классы. Каким образом можно определить конструктор агрегированного класс. Каким образом агрегированный класс может использовать член-данные используемого класса из части private.

Ответ должен содержать определение агрегированного класса, информацию об определение и работе конструкторов в агрегированном классе, описана работа с член-данными используемого класса, имеющими тип доступа private.

10. Базовый и порожденный классы. Тип доступа protected.

В ответе должна быть информация о понятии наследования, дано определение порожденного класса, приведен пример объявления порожденного класса, обоснована необходимость появления типа доступа protected, дана информация о типах наследования.

11. Простое наследование. Принцип доминирования в иерархии наследования.

Ответ должен содержать схему простого наследование и объяснение принципа доминирования. Также должно быть указано, как можно обойти принцип доминирования.

12. Конструктор порожденного класса. Деструктор порожденного класса.

В ответе должно быть указано, как определяются и работают конструкторы порожденного класса. Также указывается порядок вызовов в деструкторе порожденного класса.

13. Стандартные преобразования при наследовании. Конструктор копирования порожденного класса.

Ответ должен содержать информацию о том, какие преобразования между объектами базового и порожденного классов совершаются компилятором по умолчанию;

каким образом эти преобразования используются в конструкторе копирования порожденного класса.

14. Множественное наследование. Прямые и не прямые базовые классы.

В ответе должны быть приведены схемы множественного наследования, дано определение прямого и непрямого базового класса, объяснено, в каком случае и почему необходимо вводит виртуальный базовый класс.

15. Виртуальные функции. Чистые виртуальные функции и абстрактный базовый класс.

В ответе следует дать определение виртуальной функции, описать, какие возможности дают виртуальные функции при их виртуальных вызовах, дать определение чистой виртуальной функции и абстрактного базового класса, сформулировать ограничения для абстрактного базового класса.

Примеры практических заданий. (ИОПК-3.1, ИОПК-3.2, ИПК-1.2)

1. Реализовать класс Polinom (полином).

Член-данные класса Polinom: 1) N – степень многочлена; 2) int *A – массив коэффициентов.

Методы: Polinom(); Polinom(int *b, int n); Polinom (const Polinom&); ~ Polinom(); Scan() – ввод полинома; Print() – вывод полинома на консоль.

Перегрузка оператора: operator=, operator* – умножение полинома на число, operator-= – вычитание полиномов.

2. Реализовать класс Matrix (матрица).

Член-данные класса Matrix: 1) int** matr – массив элементов матрицы; 2) M – количество строк матрицы; 3) N – количество столбцов матрицы.

Методы: Matrix(); Matrix(int m, int n); Matrix(const Matrix&); ~Matrix(); Print() – вывод матрицы; Print() – вывод матрицы на консоль; void ShiftTop() – циклический сдвиг строк матрицы вверх (первая строка становится последней, вторая строка - первой, третья – второй и т.д.)

Перегрузка операторов: operator=; operator+ – сложение матриц, operator* – умножение матрицы на число.

Информация о разработчиках

Пахомова Елена Григорьевна, к.ф.-м.н., доцент, кафедра компьютерной безопасности, доцент