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

Институт прикладной математики и компьютерных наук

УТВЕРЖДЕНО:
Директор
А. В. Замятин

Оценочные материалы по учебной практике

Объектно-ориентированное программирование

по направлению подготовки

01.03.02 Прикладная математика и информатика

Направленность (профиль) подготовки:
Математические методы в цифровой экономике

Форма обучения
Очная

Квалификация
Бакалавр

Год приема
2024

СОГЛАСОВАНО:
Руководитель ОП
К.И. Лившиц

Председатель УМК
С.П. Сущенко

Томск – 2024

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

Целью освоения дисциплины является формирование следующих компетенций:

ОПК-2. Способен использовать и адаптировать существующие математические методы и системы программирования для разработки и реализации алгоритмов решения прикладных задач.

ОПК-4. Способен понимать принципы работы современных информационных технологий и использовать их для решения задач профессиональной деятельности.

ПК-1. Способен осуществлять научно-исследовательские и опытно-конструкторские разработки как по отдельным разделам темы, так и при исследовании самостоятельных тем.

ПК-2. Способен анализировать и оценивать риски, разрабатывать отдельные функциональные направления управления рисками.

Результатами освоения дисциплины являются следующие индикаторы достижения компетенций:

ИОПК-2.1. Обладает навыками объектно-ориентированного программирования для решения прикладных задач в профессиональной деятельности.

ИОПК-2.2. Проявляет навыки использования основных языков программирования, основных методов разработки программ, стандартов оформления программной документации.

ИОПК-2.3. Демонстрирует умение отбора среди существующих математических методов, наиболее подходящих для решения конкретной прикладной задачи.

ИОПК-2.4. Демонстрирует умение адаптировать существующие математические методы для решения конкретной прикладной задачи.

ИОПК-4.1. Обладает необходимыми знаниями в области информационных технологий, в том числе понимает принципы их работы.

ИОПК-4.2. Применяет знания, полученные в области информационных технологий, при решении задач профессиональной деятельности.

ИОПК-4.3. Использует современные информационные технологии на всех этапах решения задач профессиональной деятельности.

ИОПК-4.4. Демонстрирует умение составлять научные обзоры, рефераты и библиографии по тематике научных исследований.

ИПК-1.1. Осуществляет проведение работ по обработке и анализу научно-технической информации и результатов исследований.

ИПК-2.1. Определяет и идентифицирует риски в деятельности организации.

ИПК-2.2. Собирает и обрабатывает аналитическую информацию для анализа и оценки рисков.

ИПК-2.3. Определяет комплекс аналитических процедур и методов анализа и оценки рисков с позиции их идентификации по функциональным областям.

2. Оценочные материалы текущего контроля и критерии оценивания

Элементы текущего контроля:

– лабораторные работы;

Для оценки усвоения материала и приобретения навыка объектно-ориентированного программирования, студентам предлагается реализовать на лабораторных занятиях следующие классы.

1) Простой класс: дробь, круг, прямоугольник, прямоугольный треугольник, свободный вектор, прямая на плоскости, плоскость в пространстве, точка, отрезок, время, угол.

2) Класс «Массив».

3) Приложение с графическим интерфейсом «Калькулятор для массива»

4) Класс-шаблон «Массив».
5) Класс «Булев вектор произвольной длины».
6) Агрегированный класс «Булева матрица».
7) Агрегированный класс «Список»
8) Класс «Множество», наследник класса «Булев вектор произвольной длины»

По результатам каждой лабораторной работы студенту выставляются оценки «отлично», «хорошо», «удовлетворительно», «неудовлетворительно».

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

Оценка «хорошо» выставляется, если студент реализовал 90% методов или реализованы все методы, но в 25% методов использованы не самые эффективные алгоритмы.

Оценка «удовлетворительно» выставляется, если студент реализовал не менее 70% методов класса или реализованы все методы, но имеются отдельные методы, в которых не учитывается появление исключений.

Оценка «неудовлетворительно» выставляется, если студент реализовал менее 70% методов класса, либо в реализациях не менее 20% методов имеются грубые ошибки алгоритмического характера.

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

Промежуточная аттестация проводится в форме зачета на итоговом учебном занятии путем письменного ответа на билет. Билет содержит одно практическое задание, которое и призвано продемонстрировать полученные во время практики практические умения и навыки в области объектно-ориентированного программирования. Студент, получивший «отлично» и «хорошо» за все текущие задания практики, получает зачет автоматически.

Продолжительность зачета 1,5 часа.

Примеры практических заданий для проведения промежуточной аттестации

1. Реализовать агрегированный класс List – однонаправленный линейный список.

а) Член-данные класса List: указатель на голову(head); size – количество элементов.

б) Методы: List (); ~List(); AddToHead(int) – добавление элемента в голову; AddToPos(int) – добавление элемента на заданную позицию; DelHead()– удаление элемента из головы; DelPos(int) – удаление элемента по позиции; void reOder() – метод перестроения списка по принципу: сначала идут все элементы с положительными значениями, затем с отрицательными, порядок следования элементов в блоках сохранить.

в) Перегрузка операторов: operator- – удаление хвоста; operator== – сравнение.

2. Реализовать агрегированный класс Matrix – матрица.

а) Член-данные класса Matrix: Array *Line – массив строк матрицы; M – количество строк матрицы; N – количество столбцов матрицы.

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

в) Перегрузка операторов: operator =; operator+ – сложение матриц.

г) Класс Array (массив целых чисел) можно реализовать в самом минимальном варианте (конструкторы, деструкторы, перегрузка оператора присвоения).

3. Реализовать агрегированный класс Matrix – матрица.

а) Член-данные класса Matrix: Array *Line – массив строк матрицы; M – количество строк матрицы; N – количество столбцов матрицы.

б) Методы: Matrix (); Matrix (int m, int n); Matrix(const Matrix&); ~Matrix(); Scan() – ввод матрицы; int Track – след матрицы (сумма элементов главной диагонали матрицы).

в) Перегрузка оператора: operator* – умножение матрицы на число

г) Класс Array (массив целых чисел) можно реализовать в самом минимальном варианте (конструкторы, деструкторы, перегрузка оператора присвоения).

4. Реализовать класс Polinom – полином.

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

б) Методы: Polinom (); Polinom (int *b, int n); Polinom (const Matrix&); ~ Polinom();Scan() – ввод полинома

в) Перегрузка операторов: operator* – умножение полинома на число, operator-= – вычитание полиномов, потокового ввода

5. Реализовать класс Polinom – полином.

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

б) Методы: Polinom (); Polinom (int *b, int n); Polinom (const Matrix&); ~ Polinom(); Print() – вывод полинома

в) Перегрузка операторов: operator*= – умножение полинома на число, operator- – вычитание полиномов, потокового вывода

Результаты зачета оцениваются «зачтено» или «не зачтено».

Оценка «зачтено» выставляется, если выполнены следующие три условия:

а) студент выполнил не менее 80% заданий текущего контроля на положительную оценку;

б) представленный студентом на зачете код верен или содержит ошибки синтаксического характера;

в) код большинства методов оптимален, легко читаем, при написании кода использованы эффективные алгоритмы.

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

а) студент выполнил менее 80% заданий текущего контроля;

б) в представленном студенте на зачете классе реализованы не все методы;

в) код двух и более методов класса содержит серьёзные алгоритмические ошибки.

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

Примерный перечень теоретических вопросов

1. Каково назначение конструктора?

Варианты ответа:

a) Инициализация член-данных объекта.

b) Создание объекта.

c) Сохранение объекта.

d) Создание класса.

2. Каковы особенности конструктора?

Варианты ответа:

a) Его имя совпадает с именем класса.

b) Не имеет возвращаемого значения.
c) Работает неявно. Но возможен и его явный вызов.
d) Никогда не имеет аргументов.
e) Его возвращаемое значение всегда void.

3. В каком случае в классе необходимо определять конструктор копирования?
Варианты ответа:
a) В любом классе следует определять конструктор копирования.
b) Если для член-данных не задействована динамическая память, то программисту определять конструктор копирования не нужно.
c) Если существуют методы, которые принимают в качестве аргумента объект класса.
d) Конструктора копирования создаваемого в классе по умолчанию всегда достаточно. Можно его не переопределять.

4. Когда работает конструктор копирования?
Варианты ответа:
a) Всегда при создании точной копии объекта.
b) При инициализации одного объекта другим при объявлении объекта.
c) При передаче объекта в качестве аргумента функции по значению.
d) При передаче объекта в качестве аргумента функции по ссылку.
e) При возврате временного объекта из функции.
f) При возврате из функции объекта *this.

5. В чем состоит «правило трех»?
Варианты ответа:
a) В классе должно быть не менее трех конструкторов.
b) В классе должно быть ровно три конструктора: по умолчанию, с аргументами, конструктор копирования.
c) Если для член-данных задействована динамическая память, то следует определять конструктор копирования, деструктор и перегружать оператор присвоения.
d) В любом классе следует определять конструктор копирования, деструктор и перегружать оператор присвоения.
e) При наследовании конструктор копирования, деструктор и перегрузка оператора присвоения всегда наследуются от базового класса.

6. Каковы особенности деструктора?
Варианты ответа:
a) Его имя всегда совпадает с именем класса.
b) Его имя ~ИмяКласса.
c) У него нет аргументов.
d) У него только один аргумент.
e) У него нет возвращаемого значения.
f) Его возвращаемое значение всегда void.
g) Он работает неявно.

7. Каковы правила перегрузки операторов?
Варианты ответа:
a) Двуместный оператор можно перегрузить только как двуместный, одноместный – только как одноместный.

b) При перегрузке двуместных операторов аргументов будет два, при перегрузке одноместных операторов – аргумент один.
c) При перегрузке двуместных операторов аргумент будет один, при перегрузке одноместных операторов – аргументов не будет.
d) Перегруженный знак можно использовать привычным образом.
e) Можно перегрузить любую комбинацию знаков, создан новый знак.
f) Можно перегружать операции для любых типов данных, в том числе и стандартных.

8. Можно ли перегрузить потоковый ввод/вывод как метод класса?

Варианты ответа:

a) Да, можно.
b) Нет, потоковый ввод/вывод перегружается только как дружественная функция.
c) Потоковый ввод можно перегрузить как метод класса, потоковый вывод можно перегрузить только как дружественный метод.
d) Потоковый вывод можно перегрузить как метод класса, потоковый ввод можно перегрузить только как дружественный метод..

9. Что позволяет делать дружественность?

Варианты ответа:

a) Если функция объявлена дружественной классу, то она может использовать его приватные член-данные и методы.
b) Если функция объявлена дружественной классу, то она может использовать его приватные методы, но не может использовать его приватные член-данные.
c) Если класс A объявлен «другом» классу B, то класс A может использовать приватные член-данные и методы класса B.
d) Если класс A объявлен «другом» классу B, то класс B может использовать приватные член-данные и методы класса A.
e) Если класс A объявлен «другом» классу B, то класс A может использовать приватные методы класса B, но не может использовать его приватные член-данные.
f) Если класс A объявлен «другом» классу B, то класс B может использовать приватные методы класса A, но не может использовать его приватные член-данные.

10. Какие бывают типы доступа?

Варианты ответа:

a) public.
b) private.
c) protected.
d) static.
e) virtual.

Примеры практических заданий.

1. Определить класс Array (динамический массив). Определить обязательные методы. Кроме того, перегрузить операцию + и += (добавление элемента в конец массива), операцию == и != (сравнение массивов), оператор индексирования []. Перегрузить для класса операции потокового ввода и вывода.

2. Определить класс Matrix (динамическая матрица). Определить обязательные методы. Кроме того, перегрузить операцию + и += (сложение двух матриц), операцию ==

и != (сравнение матриц). Определить метод, позволяющий находить и изменять заданный элемент матрицы. Перегрузить для класса операции потокового ввода и вывода.

3. Определить класс Complex (комплексное число). Определить обязательные методы. Кроме того, перегрузить операцию + и += (сложение комплексных чисел), операцию / и /= (деление комплексных чисел), операцию == и != (сравнение комплексных чисел). Перегрузить для класса операции потокового ввода и вывода.

4. Определить класс FreeVector (свободный вектор). Определить обязательные методы. Кроме того, перегрузить операцию + и += (сложение векторов), операцию == и != (сравнение векторов). Определить векторное и смешанное произведение векторов. Перегрузить для класса операции потокового ввода и вывода.

2. Ответы к заданиям

№ задания	Ответ к заданию
1	a)
2	a), b), c)
3	b)
4	b), c), e)
5	c)
6	b), c), e), g)
7	a), c), d)
8	b)
9	a), c)
10	a), b), c)

Критерий оценивания остаточных знаний	Оценка
Выполнение не менее 90% теоретических заданий и полностью сделана практическая часть.	отлично
Выполнение не менее 70% теоретических заданий. Практическая часть написана полностью верно, либо допущены незначительные ошибки синтаксического характера.	хорошо
Выполнение не менее 50% теоретических заданий. Практическая часть написана полностью верно, либо допущены логические ошибки (например, не обработано исключение, допущена утечка памяти, один из методов написан не корректно).	удовлетворительно
Выполнение менее 50% теоретических заданий. Практическое задание не выполнено, либо допущены серьезные ошибки (не все обязательные методы написаны, не обработано исключение, допущена утечка памяти, два и более методов написаны не корректно).	неудовлетворительно

Информация о разработчиках

Пахомова Елена Григорьевна, канд. физ.-мат. наук, доцент, доцент кафедры компьютерной безопасности института прикладной математики и компьютерных наук НИ ТГУ.