Федеральное государственное автономное образовательное учреждение высшего образования «Московский физико-технический институт (национальный исследовательский университет)»

УТВЕРЖДЕНО
Директор физтех-школы радиотехники и компьютерных технологий
Д.А. Гаврилов

Рабочая программа дисциплины (модуля)

по дисциплине: Введение в объектно-ориентированное программирование
по направлению: Информатика и вычислительная техника
профиль подготовки: Компьютерные технологии и вычислительная техника
Физтех-школа Радиотехники и Компьютерных Технологий
кафедра радиоэлектроники и прикладной информатики
курс: 2
квалификация: бакалавр

Семестр, формы промежуточной аттестации: 3 (осенний) - Дифференцированный зачет

Аудиторных часов: 90 всего, в том числе:
лекции: 0 час.
семинары: 90 час.
лабораторные занятия: 0 час.

Самостоятельная работа: 45 час.

Всего часов: 135, всего зач. ед.: 3

Количество контрольных работ, заданий: 2

Программу составил: И.Р. Дединский, старший преподаватель

Программа обсуждена на заседании кафедры радиоэлектроники и прикладной информатики 30.01.2024

Аннотация

Курс представляет собой введение в язык C++ и объектно-ориентированное программирование. Язык C++ вводится в терминах различий C и C++, методом рефакторинга ряда решений на языке C, рассматривавшихся в осеннем семестре. Основная часть семестра посвящена технологии применения C++ (ООД, ООП, компонентное программирование) в многомодульном проекте, использующем программный код группы разработчиков в виде динамически подключаемых библиотек.

Сложность задач курса легко регулируется их функциональным наполнением.

Для обучения используются следующие принципы:

1. Во главу угла ставится задача, ее решение и, главное, путь от задачи к решению. Во всякой задаче подчеркивается разделение на идею решения и технологию реализации.
2. Самостоятельность решения является ключевым условием.
3. Понимание студентами тех средств, с помощью которых он решил задачу, ставится выше уровня самих средств решения.
4. Аккуратность и надежность решения ставятся выше «программистских трюков», иногда позволяющих в отдельных случаях добиться несколько лучших результатов.

Преподавание курса ведется в предположении, что студенты уже знают языки Си и Ассемблера.

1. Цели и задачи

Цель дисциплины

– познакомить студентов с базовыми принципами языка C++ и объектно-ориентированного программирования, их положительных и отрицательных сторон, дать навыки конструирования программных интерфейсов.

Задачи дисциплины

Задача дисциплины заключается в демонстрации базовых принципов объектно-ориентированного программирования.

2. Перечень формируемых компетенций

Освоение дисциплины направлено на формирование следующих компетенций:

Код и наименование компетенции	Индикаторы достижения компетенции
УК-1 Способен осуществлять поиск, критический анализ и синтез информации, применять системный подход для решения поставленных задач	УК-1.2 Находит, критически анализирует и выбирает информацию, необходимую для решения поставленной задачи
ОПК-2 Способен использовать современные информационные технологии и программные средства при решении задач профессиональной деятельности, соблюдая требования информационной безопасности	ОПК-2.1 Способен применять современные вычислительную технику и сервисы сети Интернет в области (сфере) профессиональной деятельности
ОПК-3 Способен составлять и оформлять научные и (или) технические (технологические, инновационные) отчеты (публикации, проекты)	ОПК-3.3 Владеет методами визуального и графического представления результатов научной (научно-технической, инновационной технологической) деятельности в виде отчетов, научных публикаций
ПК-1 Способен ставить, формализовывать и решать задачи, в том числе разрабатывать и исследовать математические модели изучаемых явлений и процессов, системно анализировать научные проблемы, получать	ПК-1.1 Способен находить, анализировать и обобщать информацию об актуальных результатах исследований в рамках тематической области своей профессиональной деятельности
	ПК-1.2 Способен выдвигать гипотезы, строить математические модели для описания изучаемых явлений и процессов, оценивать качество разработанной модели

новые научные результаты	ПК-1.3 Способен применять теоретические и (или) экспериментальные методы исследований к конкретной научной задаче и интерпретировать полученные результаты
ПК-2 Способен самостоятельно или в качестве члена (руководителя) малого коллектива организовывать и проводить научные исследования и их апробацию	ПК-2.1 Знает принципы построения научной работы, методы сбора и анализа полученного материала, способы аргументации
	ПК-2.2 Способен планировать и проводить научные исследования самостоятельно или в качестве члена (руководителя) малого научного коллектива
	ПК-2.3 Способен проводить апробацию результатов научно-исследовательской работы посредством публикации научных статей и участия в конференциях

3. Перечень планируемых результатов обучения по дисциплине (модулю)

В результате освоения дисциплины обучающиеся должны

знать:

- основы реализации больших проектов на языке C++;
- основы объектно-ориентированного программирования;
- различные пути повышения производительности программы.

уметь:

- создавать программы на языке C++;
- понимать и разрабатывать программные интерфейсы;
- работать с графическими библиотеками.

владеть:

- навыками ведения простейших программных проектов в системах контроля версий.

4. Содержание дисциплины (модуля), структурированное по темам (разделам) с указанием отведенного на них количества академических часов и видов учебных занятий

4.1. Разделы дисциплины (модуля) и трудоемкости по видам учебных занятий

№	Тема (раздел) дисциплины	Трудоемкость по видам учебных занятий, включая самостоятельную работу, час.			
		Лекции	Семинары	Лаборат. работы	Самост. работа
1	Понятие объекта и класса		8		4
2	Абстракция операций в C++		8		4
3	Класс Массив		8		4
4	Проблема владения		8		4
5	Композиция классов		8		4
6	Исключения в C++		8		4
7	Реализация классов «Список» и «Дерево» с помощью C++		8		4
8	Наследование		8		4
9	Иерархия классов		8		4
10	Понятие о компонентном программировании		4		2
11	Пример компонентного программирования		4		2
12	Обмен технической информацией		10		5
Итого часов			90		45

Подготовка к экзамену	0 час.
Общая трудоёмкость	135 час., 3 зач.ед.

4.2. Содержание дисциплины (модуля), структурированное по темам (разделам)

Семестр: 3 (Осенний)

1. Понятие объекта и класса

Их реализация и синтаксис в C++. Создание и уничтожение объекта, конструкторы и деструктор. Список инициализации конструктора. Конструктор по умолчанию и конструкторы с единственным параметром, неявность их применения и опасности, с этим связанные. Ключевое слово explicit. Реализация и синтаксис инкапсуляции данных в C++. Функции get* и set*.
Динамическая память в C++. Динамическое размещение отдельных объектов и массивов объектов, имеющих деструкторы. Применение подходящей формы оператора delete и проблемы, связанные с применением неверной его формы. Реализация класса Стек на C++ через рефакторинг «структуры Стек в стиле C», реализованной ранее.

2. Абстракция операций в C++

Переопределенные операторы. Пример построения класса «Вектор линейного пространства» или «Натуральная дробь» с переопределенными операторами. Рефакторинг функции символьного дифференцирования с применением класса для представления узла дерева и арифметических операторов, переопределенных для него, для сокращения записи правил дифференцирования.

3. Класс Массив

Реализация класса «массив» с проверкой границ и переопределенными операторами. Этапы переопределения оператора «квадратные скобки». Необходимость обрабатывать его вхождения в левых частях выражений присваивания. Понятие Lvalue и Rvalue. Реализация Lvalue через возврат указателя на элемент массива, несинтаксичность этого подхода. Понятие ссылки. Ссылка как «синтаксический сахар» над указателем. Реализация переопределенного оператора «квадратные скобки» с возвратом ссылки на элемент массива. Опасность побочных эффектов в случае неявного применения ссылок.

4. Проблема владения

Проблема владения для контейнерных классов. Особенности и методики реализации контейнеров как объектов первого класса. Идиома RAII. Понятие конструктора копирования и оператора присваивания, их ключевая роль для контейнерных объектов и ресурсных классов в целом. Стратегии реализации копирования (запрет, поверхностное копирование, глубокое копирование, подсчет ссылок). Реализация класса «строка».

5. Композиция классов

Композиция классов. Наследование без виртуальных функций и приведений типов производных и базовых классов. Синтаксис и семантика открытого и закрытого наследования. Принципы Б. Лисков для верификации отношений наследования. Агрегирование как частая лучшая альтернатива наследованию с неясной природой и мотивацией.

6. Исключения в C++

Задача об обработке исключительных ситуаций. реализация обработки исключительных ситуаций средствами языка С, тяжеловесность синтаксических конструкций. Исключения С++ как синтаксический сахар для решения таких задач. Реализация исключений в С++. try/catch-блоки, оператор throw. Работа оператора throw, свертка стека, гарантии С++ для процесса свертки стека. Исключения в конструкторах и деструкторах, поведение системы обработки исключений в случае двойного исключения и отсутствия перехвата исключения. Использование библиотечных классов исключений.

7. Реализация классов «Список» и «Дерево» с помощью С++

Перечисление элементов контейнера. Понятие итератора. Необходимость в задании пределов итерации контейнера и функциях begin и end. Переопределенные операторы итераторов. Реализация итераторов для списка и дерева. Объявление переменной итератора в случае итерации по шаблону контейнера.

8. Наследование

Ситуации, приводящие к понятию наследования (с виртуальными функциями.) Реализация динамического полиморфизма разными средствами («теги типа», указатели на функции, виртуальные функции). Применение динамического полиморфизма для встраивания собственного класса в уже работающий программный механизм, приведение типа производного класса к типу базового класса как основной концептуальный синтаксический метод этого встраивания. Реализация собственного класса для хранения наиболее полных данных о возникшем исключении (имя файла, номер строки и т.д.) как наследника стандартных классов исключений. Переопределение виртуальной функции what для форматирования сохраненных данных. Работа переопределенной функции совместно с уже существующим механизмом обработки исключений.

9. Иерархия классов

Понятие иерархии классов, ее роль в обобщенном программировании для систем с высоким повторным использованием кода. Понятие о программном движке. Примеры задач, приводящих к таким системам. Архитектура «менеджер и подчиненные», ключевая роль виртуальных функций во взаимодействии «менеджера» и «подчиненного». Чисто виртуальные функции. Понятие события и событийной функции. Ситуации, требующие восстановления типов. Механизмы восстановления точного типа объекта, основанные на тегах типов и виртуальных функциях. Оператор dynamic_cast. Динамическое приведение типов. Задача о взаимодействии объектов, зависящего от их типов. «Ловушки» механизма наследования (путаница с размерами объектов при итерации массивов, множественное наследование).

10. Понятие о компонентном программировании

Задачи, приводящие к компонентной архитектуре. Реализация ядра системы компонентной архитектуры (основной программы) и подключаемых модулей. Динамически подключаемые библиотеки. Понятие фабрики классов и ее реализация. Реализация фабрики класса через обмен структур с указателями на функции. Сходство таких структур с таблицами виртуальных функций. Реализация через возврат объекта базового класса без данных и лишь с множеством чисто виртуальных функций. Понятие класса-интерфейса. Проблема реализации таблиц виртуальных функций в различных компиляторах. Проблема передачи исключений через границу динамически подключаемой библиотеки. Проблема стандарта API компонентов и его эволюции. Понятие компонента-адаптера. Понятие о шаблонах и анти-шаблонах проектирования.

11. Пример компонентного программирования

Разбор проекта мультипроцессорной системы с подключаемыми модулями (полигон для соревнований виртуальных роботов). Архитектура системы, разработка API среды исполнения процессоров и API виртуального процессора. Этапы разработки среды для соревнований виртуальных роботов. Разработка прототипов полигона и базового робота. Построение рабочей версии компонентов системы из прототипов.

12. Обмен технической информацией

Структура технического сообщения, технология реализации технической презентации (СТР). Примеры реализаций проектов по тематике Intel (С и С++, ООП и ООД «в реальной жизни»). Мини-конференция: сообщения от разных групп студентов, реализовавших тот или иной вариант компонентного проекта.

5. Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине (модулю)

Учебная аудитория, оснащенная компьютером и мультимедийным оборудованием (проектор, звуковая система).

6.Перечень рекомендуемой литературы

Основная литература

1. Язык программирования С++ [Текст] / Б. Страуструп ; пер. с англ. С. Анисимова, М. Кононова ; под ред. Ф. Андреева, А. Ушакова .— Спец. изд. с авт. изменениями и доп. — М. : Бином Пресс, 2008 .— 1104 с.

фонд литературы кафедры:

2. Б. Страуструп. Дизайн и эволюция языка С++.
3. Б. Эккель. Философия С++.
4. Э. Гамма, Р. Хелм, Р. Джонсон, Д. Влиссидес. Паттерны объектно-ориентированного программирования.

Дополнительная литература

7. Перечень ресурсов информационно-телекоммуникационной сети "Интернет", необходимых для освоения дисциплины (модуля)

1. http://refactoring.guru

8. Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине (модулю), включая перечень необходимого программного обеспечения и информационных справочных систем (при необходимости)

Операционная система Linux. Среда программирования.

9. Методические указания для обучающихся по освоению дисциплины (модуля)

Успешное освоение курса требует напряжённой самостоятельной работы студента. В программе курса приведено минимально необходимое время для работы студента над темой. Самостоятельная работа включает в себя проработку учебного материала (по учебной литературе), подготовку к практическим занятиям.

Промежуточный контроль знаний проводится в виде сдач проектных заданий.

ПРИЛОЖЕНИЕ

ОЦЕНОЧНЫЕ МАТЕРИАЛЫ ПО ДИСЦИПЛИНЕ (МОДУЛЮ)

по направлению: Информатика и вычислительная техника
профиль подготовки: Компьютерные технологии и вычислительная техника
Физтех-школа Радиотехники и Компьютерных Технологий
кафедра радиоэлектроники и прикладной информатики
курс: 2
квалификация: бакалавр

Семестр, формы промежуточной аттестации: 3 (осенний) - Дифференцированный зачет

Разработчик: И.Р. Дединский, старший преподаватель

1. Компетенции, формируемые в процессе изучения дисциплины

Код и наименование компетенции	Индикаторы достижения компетенции
УК-1 Способен осуществлять поиск, критический анализ и синтез информации, применять системный подход для решения поставленных задач	УК-1.2 Находит, критически анализирует и выбирает информацию, необходимую для решения поставленной задачи
ОПК-2 Способен использовать современные информационные технологии и программные средства при решении задач профессиональной деятельности, соблюдая требования информационной безопасности	ОПК-2.1 Способен применять современные вычислительную технику и сервисы сети Интернет в области (сфере) профессиональной деятельности
ОПК-3 Способен составлять и оформлять научные и (или) технические (технологические, инновационные) отчеты (публикации, проекты)	ОПК-3.3 Владеет методами визуального и графического представления результатов научной (научно-технической, инновационной технологической) деятельности в виде отчетов, научных публикаций
ПК-1 Способен ставить, формализовывать и решать задачи, в том числе разрабатывать и исследовать математические модели изучаемых явлений и процессов, системно анализировать научные проблемы, получать новые научные результаты	ПК-1.1 Способен находить, анализировать и обобщать информацию об актуальных результатах исследований в рамках тематической области своей профессиональной деятельности
	ПК-1.2 Способен выдвигать гипотезы, строить математические модели для описания изучаемых явлений и процессов, оценивать качество разработанной модели
	ПК-1.3 Способен применять теоретические и (или) экспериментальные методы исследований к конкретной научной задаче и интерпретировать полученные результаты
ПК-2 Способен самостоятельно или в качестве члена (руководителя) малого коллектива организовывать и проводить научные исследования и их апробацию	ПК-2.1 Знает принципы построения научной работы, методы сбора и анализа полученного материала, способы аргументации
	ПК-2.2 Способен планировать и проводить научные исследования самостоятельно или в качестве члена (руководителя) малого научного коллектива
	ПК-2.3 Способен проводить апробацию результатов научно-исследовательской работы посредством публикации научных статей и участия в конференциях

2. Показатели оценивания компетенций

В результате изучения дисциплины «Введение в объектно-ориентированное программирование» обучающийся должен:

знать:

- основы реализации больших проектов на языке C++;
- основы объектно-ориентированного программирования;
- различные пути повышения производительности программы.

уметь:

- создавать программы на языке C++;
- понимать и разрабатывать программные интерфейсы;
- работать с графическими библиотеками.

владеть:

- навыками ведения простейших программных проектов в системах контроля версий.

3. Перечень типовых (примерных) вопросов, заданий, тем для подготовки к текущему контролю

Примеры заданий:

1. Реализовать класс с переопределением операторов на языке C++.
2. Реализовать списка и дерева на языке C++.
3. Реализовать модель взаимодействия молекул газов на языке C++.
4. Реализовать простейший графический редактор на языке C++.
5. Разработать интерфейс плагинов для графического редактора.
6. Реализовать программу с использованием принципов компонентного программирования.

За каждое задание выставляется оценка в соответствии со следующими критериями оценивания:

Отлично

10) Полностью и вовремя решены все задачи без ошибок. Продемонстрирован грамотный подход к решению задач, реализованы оптимальные алгоритмы, код оформлен в едином удобочитаемом стиле.

9) Полностью и вовремя решены все задачи без ошибок. Продемонстрирован грамотный подход к решению задач, реализованы оптимальные алгоритмы.

8) Полностью и вовремя решены все задачи без ошибок. Продемонстрирован грамотный подход к решению задач.

Хорошо

7) Полностью решены все задачи. Допущены несущественные ошибки

6) Полностью решено большинство задач. В некоторых задачах допущены и не исправлены ошибки, либо некоторые задачи решены частично.

5) Полностью решено две трети задач. В некоторых задачах допущены и не исправлены ошибки, либо некоторые задачи решены частично.

Удовлетворительно.

4) Полностью решено более половины задач. В остальных задачах допущены и не исправлены ошибки, либо некоторые задачи решены частично.

3) Полностью решено более половины задач.

Неудовлетворительно

2) Решено менее половины задач.

1) Не решено ни одной задачи.

4. Перечень типовых (примерных) вопросов и тем для проведения промежуточной аттестации обучающихся

1. Объясните понятия compilation error, runtime error и undefined behaviour. Приведите по паре примеров того, другого и третьего.
2. Что такое выражения, операторы? Для чего предназначены и что по стандарту делают следующие операторы: тернарный оператор, оператор “запятая”, унарная звездочка, унарный амперсанд, операторы “точка” и “стрелочка”, двойное двоеточие, префиксный и постфиксный инкремент, бинарные & и &&, операторы простого и составного присваивания, операторы << и >>?
3. Что такое стековая память и динамическая память? Что такое stack overflow? Зачем нужен оператор new? Что такое утечки памяти? Что такое сборка мусора?
4. Объясните идею ссылок (references). Для чего они нужны, чем они отличаются от указателей? Что такое “передача аргументов по ссылке и по значению”? Как в C++03 реализовать функцию swap?
5. Что такое класс, структура, в чем разница между ними? Что такое поля, методы, модификаторы доступа, инкапсуляция?

6. Что такое конструкторы, деструкторы, для чего они нужны, каков синтаксис их определения? Те же вопросы про конструктор копирования и оператор присваивания.

7. Что такое перегрузка функций? Что такое перегрузка операторов? Приведите примеры операторов, которые можно и нельзя перегружать. Приведите пример перегрузки хоть какого-нибудь оператора (напишите сигнатуру его перегрузки).

8. Что такое наследование? В чем разница между приватным и публичным наследованием?

9. Что такое виртуальные функции, чисто виртуальные функции, абстрактные классы? Что такое виртуальное наследование?

10. Что такое полиморфизм? Приведите пример полиморфизма в C++.

11. Что такое шаблоны? Что такое инстанцирование шаблонов, специализация шаблонов?

12. Что такое исключения? Как пользоваться механизмом обработки исключений? Как бросить, поймать исключение? Приведите какой-нибудь пример.

Критерии оценивания

Оценка "отлично (10)" выставляется студенту, показавшему всесторонние, систематизированные, глубокие знания учебной программы дисциплины, проявляющему интерес к данной предметной области, продемонстрировавшему умение уверенно и творчески применять их на практике при решении конкретных задач, свободное и правильное обоснование принятых решений.

Оценка "отлично (9)" выставляется студенту, показавшему всесторонние, систематизированные, глубокие знания учебной программы дисциплины и умение уверенно применять их на практике при решении конкретных задач, свободное и правильное обоснование принятых решений.

Оценка "отлично (8)" выставляется студенту, показавшему всесторонние, систематизированные, глубокие знания учебной программы дисциплины и умение уверенно применять их на практике при решении конкретных задач, правильное обоснование принятых решений, с некоторыми недочетами.

Оценка "хорошо (7)" выставляется студенту, если он твердо знает материал, грамотно и по существу излагает его, умеет применять полученные знания на практике, но недостаточно грамотно обосновывает полученные результаты.

Оценка "хорошо (6)" выставляется студенту, если он твердо знает материал, грамотно и по существу излагает его, умеет применять полученные знания на практике, но допускает в ответе или в решении задач некоторые неточности.

Оценка "хорошо (5)" выставляется студенту, если он в основном знает материал, грамотно и по существу излагает его, умеет применять полученные знания на практике, но допускает в ответе или в решении задач достаточно большое количество неточностей.

Оценка "удовлетворительно (4)" выставляется студенту, показавшему фрагментарный, разрозненный характер знаний, недостаточно правильные формулировки базовых понятий, нарушения логической последовательности в изложении программного материала, но при этом он освоил основные разделы учебной программы, необходимые для дальнейшего обучения, и может применять полученные знания по образцу в стандартной ситуации.

Оценка "удовлетворительно (3)" выставляется студенту, показавшему фрагментарный, разрозненный характер знаний, допускающему ошибки в формулировках базовых понятий, нарушения логической последовательности в изложении программного материала, слабо владеет основными разделами учебной программы, необходимыми для дальнейшего обучения и с трудом применяет полученные знания даже в стандартной ситуации.

Оценка "неудовлетворительно (2)" выставляется студенту, который не знает большей части основного содержания учебной программы дисциплины, допускает грубые ошибки в формулировках основных принципов и не умеет использовать полученные знания при решении типовых задач.

Оценка "неудовлетворительно (1)" выставляется студенту, который не знает основного содержания учебной программы дисциплины, допускает грубейшие ошибки в формулировках базовых понятий дисциплины и вообще не имеет навыков решения типовых практических задач.

5. Методические материалы, определяющие процедуры оценивания знаний, умений, навыков и (или) опыта деятельности

Дифференцированный зачет проводится по итогам текущей успеваемости и сдачи заданий, предусмотренных программой дисциплины.

Дифференцированный зачет проводится в устной форме.

При проведении дифференцированного зачета обучающемуся предоставляется 20 минут на подготовку. Опрос обучающегося проводится в течение 30 минут.