МИНОБРНАУКИ РОССИИ

ВЛАДИВОСТОКСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ЭКОНОМИКИ И СЕРВИСА

КАФЕДРА ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ И СИСТЕМ

Рабочая программа дисциплины (модуля)

ОБЪЕКТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ

Направление и направленность (профиль)

38.03.05 Бизнес-информатика. Бизнес-аналитика

Год набора на ОПОП
2020

Форма обучения
очная

Владивосток 2021

Рабочая программа дисциплины (модуля) «Объектно-ориентированное программирование» составлена в соответствии с требованиями ФГОС ВО по направлению(ям) подготовки 38.03.05 Бизнес-информатика (утв. приказом Минобрнауки России от 11.08.2016г. №1002) и Порядком организации и осуществления образовательной деятельности по образовательным программам высшего образования – программам бакалавриата, программам специалитета, программам магистратуры (утв. приказом Минобрнауки России от 05.04.2017 г. N301).

Составитель(и):

Ивин В.В., кандидат экономических наук, доцент, Кафедра информационных технологий и систем, Vyacheslav.Ivin@vvsu.ru

Утверждена на заседании кафедры информационных технологий и систем от 31.05.2021 , протокол № 9

СОГЛАСОВАНО:

Заведующий кафедрой (разработчика)
Кийкова Е.В.

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ	
Сертификат	1575633692
Номер транзакции	000000000074AB66
Владелец	Кийкова Е.В.

Заведующий кафедрой (выпускающей)
Мазелис Л.С.

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ	
Сертификат	1575656200
Номер транзакции	000000000074C8F9
Владелец	Мазелис Л.С.

1. Цель и задачи освоения дисциплины (модуля)

Целью изучения дисциплины «Объектно-ориентированное программирование» является теоретическая и практическая подготовка студентов в области разработки программного обеспечения с использованием объектно-ориентированной модели современных языков программирования. Знания, полученные в результате освоения дисциплины, помогут при разработке системных программных компонентов современных информационных и расчетных программ, в проектировании и реализации системных компонентов операционных систем в такой степени, чтобы студенты могли самостоятельно выбирать средства реализации, находить необходимые программные и технологические решения для практически важных системных и предметно-ориентированных задач.

Основные задачи изучения дисциплины:

– приобретение студентами знаний о сущности объектно-ориентированного подхода в программировании;

– ознакомление с технологиями создания новых типов данных в различных языках программирования;

– приобретение практических навыков по использованию средств переопределения операций, обработки исключений.

2. Перечень планируемых результатов обучения по дисциплине (модулю), соотнесенных с планируемыми результатами освоения образовательной программы

Планируемыми результатами обучения по дисциплине являются знания, умения, навыки, соотнесенные с компетенциями, которые формирует дисциплина, и обеспечивающие достижение планируемых результатов по образовательной программе в целом. Перечень компетенций, формируемых в результате изучения дисциплины (модуля), приведен в таблице 1.

Таблица 1 – Компетенции обучающегося, формируемые в результате освоения дисциплины (модуля)

Название ОПОП ВО, сокращенное	Код компетенции	Формулировка компетенции	Планируемые результаты обучения	
38.03.05 «Бизнес-информатика» (Б-БИ)	ПК-13	Умение проектировать и внедрять компоненты ИТ-инфраструктуры предприятия, обеспечивающие достижение стратегических целей и поддержку бизнес-процессов	Знания:	объектно-ориентированной методологии программирования и ИТ-инфраструктуры предприятия
			Умения:	проектировать программное обеспечение для достижения стратегических целей и поддержки бизнес-процессов предприятия
			Навыки:	внедрения компонентов ИТ-инфраструктуры предприятия с применением объектно-ориентированной методологии программирования
	ПК-16	Умение разрабатывать контент и ИТ-сервисы предприятия и интернет-ресурсов	Знания:	методов разработки программного обеспечения и интернет-ресурсов с применением объектно-ориентированной методологии программирования
			Навыки:	разработки контента и ИТ-сервисов для нужд предприятия с применением объектно-ориентированной методологии программирования

	ОПК-3	Способность работать с компьютером как средством управления информацией, работать с информацией из различных источников, в том числе в глобальных компьютерных сетях	Знания:	современные информационные технологии разработки программных продуктов с применением объектно-ориентированной методологии программирования
			Умения:	разрабатывать и применять на практике современные алгоритмы поиска информации в глобальных компьютерных сетях
			Навыки:	использования компьютера как инструмента разработки программного обеспечения и средства управления информацией

3. Место дисциплины (модуля) в структуре основной образовательной программы

Дисциплина относится к базовой части блока 1 дисциплин учебного плана направления «Бизнес-информатика».

Входными требованиями, необходимыми для освоения дисциплины, является наличие у обучающихся компетенций, сформированных при изучении дисциплин и/или прохождении практик «Информатика и программирование модуль 1», «Основы алгоритмизации и языки программирования». На данную дисциплину опираются «Информационные технологии управления персоналом», «Операционные системы», «Проектирование информационных систем», «Проектный практикум», «Сети ЭВМ и телекоммуникации», «Теория информационных процессов и систем», «Хранилища данных».

4. Объем дисциплины (модуля)

Объем дисциплины в зачетных единицах с указанием количества академических часов, выделенных на контактную работу с обучающимися (по видам учебных занятий) и на самостоятельную работу, приведен в таблице 2.

Таблица 2 – Общая трудоемкость дисциплины

Название ОПОП ВО	Форма обучения	Часть УП	Семестр (ОФО) или курс (ЗФО, ОЗФО)	Трудоемкость	Объем контактной работы (час)						СРС	Форма аттестации
				(З.Е.)	Всего	Аудиторная			Внеаудиторная			
						лек.	прак.	лаб.	ПА	КСР		
38.03.05 Бизнес-информатика	ОФО	Бл1.Б	3	3	55	18	36	0	1	0	53	ДЗ

5. Структура и содержание дисциплины (модуля)

5.1 Структура дисциплины (модуля) для ОФО

Тематический план, отражающий содержание дисциплины (перечень разделов и тем), структурированное по видам учебных занятий с указанием их объемов в соответствии с учебным планом, приведен в таблице 3.1

Таблица 3.1 – Разделы дисциплины (модуля), виды учебной деятельности и формы текущего контроля для ОФО

№	Название темы	Кол-во часов, отведенное на				Форма текущего контроля
		Лек	Практ	Лаб	СРС	
1	Объектно-ориентированный подход в программировании	2	0	0	6	текущий опрос
2	Классы и объекты	4	16	0	6	текущий опрос, отчёт о выполнении практической работы
3	Объектно-ориентированная методология программирования	2	6	0	6	текущий опрос, отчёт о выполнении практической работы
4	Наследование, базовый и производный классы	2	4	0	8	текущий опрос, отчёт о выполнении практической работы
5	Функции и классы	2	4	0	8	текущий опрос, отчёт о выполнении практической работы
6	Шаблоны функций и классов	2	6	0	6	текущий опрос, отчёт о выполнении практической работы
7	Потоки и файлы	2	0	0	6	текущий опрос
8	Объектный подход к разработке программного обеспечения	2	0	0	7	текущий опрос
Итого по таблице		**18**	**36**	**0**	**53**	

5.2 Содержание разделов и тем дисциплины (модуля) для ОФО

Тема 1 Объектно-ориентированный подход в программировании.

Содержание темы: Сущность объектно-ориентированного подхода в программировании. Цикл разработки программного обеспечения (ПО), назначение и содержание этапов. Роль анализа в процессе разработки программного обеспечения. Основные понятия объектно-ориентированного анализа. Язык C++. Язык Java.

Формы и методы проведения занятий по теме, применяемые образовательные технологии: лекция.

Виды самостоятельной подготовки студентов по теме: подготовка к промежуточному тестированию.

Тема 2 Классы и объекты.

Содержание темы: Отношения, основные типы отношений. Язык UML. Основные средства анализа и моделирования предметной области в языке UML. Статические данные. Конструктор, деструктор. Операции new и delete.

Формы и методы проведения занятий по теме, применяемые образовательные технологии: лекция, практическая работа.

Виды самостоятельной подготовки студентов по теме: подготовка к промежуточному тестированию, практическим работам.

Тема 3 Объектно-ориентированная методология программирования.

Содержание темы: Технология применения объектно-ориентированных языков, их классификация и архитектура. Перегрузка операций. Преобразование типов.

Формы и методы проведения занятий по теме, применяемые образовательные технологии: лекция, практическая работа.

Виды самостоятельной подготовки студентов по теме: подготовка к промежуточному тестированию, практическим работам.

Тема 4 Наследование, базовый и производный классы.

Содержание темы: Простое и сложное наследование. Абстракция данных, наследование и полиморфизм.

Формы и методы проведения занятий по теме, применяемые образовательные технологии: лекция, практическая работа.

Виды самостоятельной подготовки студентов по теме: подготовка к промежуточному тестированию, практическим работам.

Тема 5 Функции и классы.

Содержание темы: Виртуальные функции. Дружественные функции. Дружественные классы.

Формы и методы проведения занятий по теме, применяемые образовательные технологии: лекция, практическая работа.

Виды самостоятельной подготовки студентов по теме: подготовка к промежуточному тестированию, практическим работам.

Тема 6 Шаблоны функций и классов.

Содержание темы: Шаблоны функций. Шаблоны классов. Исключения. Стандартная библиотека шаблонов.

Формы и методы проведения занятий по теме, применяемые образовательные технологии: лекция, практическая работа.

Виды самостоятельной подготовки студентов по теме: подготовка к промежуточному тестированию, практическим работам.

Тема 7 Потоки и файлы.

Содержание темы: Стандартная библиотека классов для управления потоками. Методы и средства организации и программирования интерфейса.

Формы и методы проведения занятий по теме, применяемые образовательные технологии: лекция.

Виды самостоятельной подготовки студентов по теме: подготовка к промежуточному тестированию.

Тема 8 Объектный подход к разработке программного обеспечения.

Содержание темы: Стандарты кодирования и их проекция на объектно-ориентированную модель программирования. Объектный подход к разработке ПО для распределенных систем.

Формы и методы проведения занятий по теме, применяемые образовательные технологии: лекция.

Виды самостоятельной подготовки студентов по теме: подготовка к промежуточному тестированию.

6. Методические указания по организации изучения дисциплины (модуля)

При реализации дисциплины (модуля) применяется электронный учебный курс, размещённый в системе электронного обучения Moodle.

В ходе изучения дисциплины «Объектно-ориентированное программирование» студенты могут посещать аудиторные занятия (лекции, практические занятия, консультации).

Особое место в овладении частью тем данной дисциплины может отводиться самостоятельной работе, при этом во время аудиторных занятий могут быть рассмотрены и проработаны наиболее важные и трудные вопросы по той или иной теме дисциплины, а второстепенные и более легкие вопросы, а также вопросы, специфичные для ОПОП, могут быть изучены студентами самостоятельно.

В соответствии с учебным планом процесс изучения дисциплины может

предусматривать проведение лекций, практических занятий, консультаций, а также самостоятельную работу студентов. Обязательным является проведение практических занятий в специализированных компьютерных аудиториях, оснащенных подключенными к центральному серверу терминалами или персональными компьютерами.

В рамках общего объема часов, отведенных для изучения дисциплины, предусматривается выполнение следующих видов самостоятельных работ студентов: изучение теоретического материала при подготовке к защите практических работ, итоговое повторение теоретического материала.

Для закрепления материала и приобретения навыков самостоятельного написания объектно-ориентированных программ рекомендуется выполнение следующих задач:

1. Написать шаблон класса для работы с очередью FIFO. Определить функции включения и исключения элементов. Добавить механизм обработки исключений при превышении размера очереди и при попытке удалить данные из пустой очереди. Это можно сделать, добавив элемент данных – счетчик текущего числа элементов. Исключения генерируются, если счетчик превысил размер массива или если он стал меньше 0.

2. Опишите классы PointXY и PointPolar, объекты которых задают декартовы и полярные координаты точки на плоскости. Перегрузите для этих классов операции сложения, вычитания и умножения как скалярного произведения, так, чтобы в них могли участвовать объекты как одного, так и обоих классов. Кроме того, задайте функцию преобразования одного класса в другой (для обоих классов).

3. Определить класс stack, который позволяет реализовать структуру данных типа стек для хранения целых чисел. Конструктор класса должен содержать параметр, определяющий размер стека. Определить для класса функции pop() (достать из стека), push() (положить в стек) и операцию определения текущего размера стека. Функции должны осуществлять проверку на выход за пределы стека. Определить класс fifo, реализующий структуру данных типа очередь для хранения целых чисел, породив его от класса stack, добавив нужные поля и переопределив функции pop(), push() и определение текущего размера очереди. Продемонстрировать работу.

4. В любой визуальной среде создать класс Figure с виртуальным методом draw(), осуществляющим прорисовку объекта на визуальном компоненте. Создать производные классы: Rectangle (прямоугольник), Circle (круг), Triangle (треугольник). Описать в производных классах функции draw() для каждой из фигур, продемонстрировать работу механизма виртуальных функций.

Особенности организации обучения для лиц с ограниченными возможностями здоровья и инвалидов.

При необходимости обучающимся из числа лиц с ограниченными возможностями здоровья и инвалидов (по заявлению обучающегося) предоставляется учебная информация в доступных формах с учетом их индивидуальных психофизических особенностей:

- для лиц с нарушениями зрения: в печатной форме увеличенным шрифтом; в форме электронного документа; индивидуальные консультации с привлечением тифлосурдопереводчика; индивидуальные задания, консультации и др.

- для лиц с нарушениями слуха: в печатной форме; в форме электронного документа; индивидуальные консультации с привлечением сурдопереводчика; индивидуальные задания, консультации и др.

- для лиц с нарушениями опорно-двигательного аппарата: в печатной форме; в форме электронного документа; индивидуальные задания, консультации и др.

7. Фонд оценочных средств для проведения текущего контроля и промежуточной аттестации обучающихся по дисциплине (модулю)

В соответствии с требованиями ФГОС ВО для аттестации обучающихся на соответствие их персональных достижений планируемым результатам обучения по дисциплине созданы фонды оценочных средств. Типовые контрольные задания,

методические материалы, определяющие процедуры оценивания знаний, умений и навыков, а также критерии и показатели, необходимые для оценки знаний, умений, навыков и характеризующие этапы формирования компетенций в процессе освоения образовательной программы, представлены в Приложении 1.

8. Учебно-методическое и информационное обеспечение дисциплины (модуля)

8.1 Основная литература

1. Зыков С. В. ПРОГРАММИРОВАНИЕ. ОБЪЕКТНО-ОРИЕНТИРОВАННЫЙ ПОДХОД. Учебник и практикум для вузов [Электронный ресурс] , 2020 - 155 - Режим доступа: https://urait.ru/book/programmirovanie-obektno-orientirovannyy-podhod-451488
2. Язык C++ и основы технологии объектно ориентированного программирования. Ч. 1 [Электронный ресурс] , 2017 - 64 - Режим доступа: https://lib.rucont.ru/efd/673192
3. Язык C++ и основы технологии объектно ориентированного программирования. Ч. 2 [Электронный ресурс] , 2017 - 56 - Режим доступа: https://lib.rucont.ru/efd/673193

8.2 Дополнительная литература

1. Объектно-ориентированное программирование [Электронный ресурс] , 2018 - 112 - Режим доступа: https://lib.rucont.ru/efd/705248
2. Уйманова Н. А. Основы объектно-ориентированного программирования [Электронный ресурс] , 2017 - 156 - Режим доступа: https://lib.rucont.ru/efd/646120
3. Широков А. С. Объектно-ориентированное программирование на языке C++ [Электронный ресурс] , 2018 - 32 - Режим доступа: https://lib.rucont.ru/efd/673466

8.3 Ресурсы информационно-телекоммуникационной сети "Интернет", включая профессиональные базы данных и информационно-справочные системы (при необходимости):

1. http://programmersforum.ru – форум программистов
2. Комлев Н.Ю. Объектно Ориентированное Программирование. Хорошая книга для Хороших Людей : Практическое пособие [Электронный ресурс] : СОЛОН-Пресс - Режим доступа: https://znanium.com/catalog/document?id=392258
3. СПС КонсультантПлюс - Режим доступа: http://www.consultant.ru/
4. Электронная библиотечная система «РУКОНТ» - Режим доступа: https://lib.rucont.ru/
5. Электронно-библиотечная система издательства "Юрайт" - Режим доступа: https://urait.ru/
6. Open Academic Journals Index (OAJI). Профессиональная база данных - Режим доступа: http://oaji.net/
7. Президентская библиотека им. Б.Н.Ельцина (база данных различных профессиональных областей) - Режим доступа: https://www.prlib.ru/

9. Материально-техническое обеспечение дисциплины (модуля) и перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине (модулю) (при необходимости)

Основное оборудование:

- Компьютеры

· Проектор
· Мульт. медийный комплект № 2: Проектор Panasonic PT-LX26HE, потолочное крепление Tuarex Corsa, клеммный модуль Kramer WX -1N, коннектор VGA, экран Lumien Ecopicture
· Мультимедийный комплект №2 в составе:проектор Casio XJ-M146,экран 180*180,крепление потолочное
· Облачный монитор 23" LG CAV42K
· Облачный монитор LG Electronics черный +клавиатура+мышь
· Усилитель-распределитель Kramer VP-200N 1:2
· Экран рулонный

Программное обеспечение:

· Adobe Acrobat Reader
· C++ Builder 2006 Russian
· C++Builder
· C++Builder 2010 Professional
· Microsoft Office 2010 Standard Russian
· Microsoft Windows 10 Professional OEM
· КонсультантПлюс

10. Словарь основных терминов

Абстрактный класс – это класс, содержащий хотя бы один виртуальный метод. Абстрактные классы не бывают изолированными, т.е. всегда абстрактный класс должен быть наследуемым. Поскольку у чисто виртуального метода нет тела, то создать объект абстрактного класса невозможно. Абстрактным классом можно назвать класс, специально определенный для обеспечения наследования характеристик порождёнными классами.

Абстракция – процесс изменения уровня детализации программы. Когда мы абстрагируемся от проблемы, мы предполагаем игнорирование ряда подробностей с тем, чтобы свести задачу к более простой.

Абстракция через параметризацию – приём программирования, позволяющий, используя параметры, представить фактически неограниченный набор различных вычислений одной программой, которая есть абстракция этих наборов.

Абстракция через спецификацию – приём программирования, позволяющий абстрагироваться от процесса вычислений, описанных в теле процедуры, до уровня знания того, что данная процедура делает. Это достигается путём задания спецификации, описывающей эффект её работы, после чего смысл обращения к данной процедуре становится ясным через анализ этой спецификации, а не самого тела процедуры. Мы пользуемся абстракцией через спецификацию всякий раз, когда связываем с процедурой некий комментарий, достаточно информативный для того, чтобы иметь возможность работать без анализа тела процедуры. Абстракция через спецификацию позволяет абстрагироваться от процесса вычислений, описанных в теле процедуры, до уровня знания того, *что данная процедура делает.* Это достигается путём задания *спецификации*, описывающей эффект её работы, после чего смысл обращения к данной процедуре становится ясным через анализ этой спецификации, а не самого тела процедуры. Мы пользуемся абстракцией через спецификацию всякий раз, когда связываем с процедурой некий комментарий, достаточно информативный для того, чтобы иметь возможность работать без анализа тела процедуры.

Аспектно-ориентированное сборочное программирование – разновидность сборочного программирования, основанная на сборке полнофункциональных приложений из многоаспектных компонентов, инкапсулирующих различные варианты реализации.

Декомпозиция программы – создание модулей, которые в свою очередь представляют собой небольшие программы, взаимодействующие друг с другом по хорошо определенным и простым правилам.

Диаграмма деятельности, Activity diagram – методология объектно-ориентированного проектирования, предназначенная для детализации особенностей алгоритмической и логической организации системы. При этом каждое действие расчленяется на фундаментальные процессы. На диаграмме деятельности управление осуществляется: - либо через потоки управления (явно); - либо через определяемые потоки данных (неявно).

Диаграмма классов, Class diagram – методология объектно-ориентированного проектирования, предназначенная для представления статической структуры модели системы в терминологии классов объектно-ориентированного программирования.

Диаграмма компонентов, Component diagram – метод объектно-ориентированного проектирования, описывающий особенности физического представления системы. Диаграмма компонентов позволяет определить архитектуру разрабатываемой системы, устанавливая зависимости между компонентами.

Диаграмма кооперации, Collaboration diagrams – метод объектно-ориентированного проектирования, основанный на графическом представлении всех структурных отношений между объектами, участвующими во взаимодействии. Диаграмма кооперации представляет собой граф, в вершинах которого располагаются объекты, соединенные дугами-связями. При этом дуги могут быть аннотированы сообщениями, которыми обмениваются объекты.

Диаграмма последовательности, Sequence diagram – методология объектно-ориентированного проектирования, предназначенная для моделирования взаимодействия во времени. Диаграмма последовательности позволяет отслеживать поведение взаимодействующих групп объектов.

Диаграмма развертывания, Диаграмма применения, Диаграмма размещения Deployment diagram – метод объектно-ориентированного проектирования, отображающий физические взаимосвязи между программными и аппаратными компонентами системы.

Диаграмма состояний, Statechart diagram – методология объектно-ориентированного проектирования, предназначенная для представления жизненного цикла объектов в реальном или абстрактном мире. Диаграмма состояний состоит - из множества состояний объектов; - из множества событий, сообщающих о перемещении чего-либо в новое состояние; - из множества правил переходов, определяющих новое состояние объекта при возникновении тех или иных событий; - из множества действий, которые должны быть выполнены объектом, когда он переходит в новое состояние.

Инкапсуляция, Encapsulation – От лат.In – в + Capsula – ящичек, в объектно-ориентированном программировании – сокрытие внутренней структуры данных и реализации методов объекта от остальной программы. Другим объектам доступен только интерфейс объекта, через который осуществляется все взаимодействие с ним.

Карты класс-ответственность-кооперация, Class-responsibility-collaboration – Карты класс-ответственность-кооперация – методология объектно-ориентированного проектирования, предназначенная для описания классов и оперирующая понятиями: - ответственность – суть – высокоуровневое описание функций, которые выполняет класс; - кооперация – суть – ссылка на другие классы, с которыми необходимо кооперироваться для реализации функций.

Класс, Class – Класс – в программировании – множество объектов, которые обладают одинаковой структурой, поведением и отношением с объектами из других классов.

Компонентное сборочное программирование – объектно-ориентированное сборочное программирование, основанное на распространении классов в бинарном виде и предоставление доступа к методам класса через строго определенные интерфейсы. Компонентное сборочное программирование поддерживают технологические подходы COM, CORBA, .Net.

Конструкторы – Эти операции используют в качестве аргументов объекты соответствующего им типа и создают другие объекты такого же типа. Например, операция сложения матриц создаёт новую матрицу.

Локальность – означает, что реализация одной абстракции может быть создана и рассмотрена без необходимости анализа реализации какой-либо другой абстракции. Принцип локальности позволяет составлять программу из абстракций, создаваемых людьми, работающими независимо друг от друга. Один человек может создать абстракцию, которая использует абстракцию, созданную кем-то другим.

Метод объектно-ориентированной декомпозиции – основной метод объектно-ориентированного программирования, описывающий: - статическую структуру системы в терминах объектов и связей между ними; - поведение системы в терминах обмена сообщениями между объектами.

Модификаторы – эти операции модифицируют объекты соответствующего им типа. Например, операция push для стека.

Модульность – это такая организация объектов, когда они заключают в себе полное определение их характеристик, никакие определения методов и свойств не должны располагаться вне его, это делает возможным свободное копирование и внедрение одного объекта в другие.

Наблюдатели – эти операции используют в качестве аргумента объекты соответствующего им типа и возвращают элемент другого типа, они используются для получения информации об объекте. Сюда относятся, например, операции типа size.

Наследование, Inheritance – Наследование – в объектно-ориентированном программировании – свойство объекта, заключающееся в том, что характеристики одного объекта (объекта-предка) могут передаваться другому объекту (объекту-потомку) без их повторного описания. Наследование упрощает описание объектов.

Объект, Object – Объект – в программировании – программный модуль: - объединяющий в себе данные (свойства) и операции над ними (методы); - обладающий свойствами наследования, инкапсуляции и полиморфизма. Объекты взаимодействуют между собой, посылая друг другу сообщения.

Объектно-ориентированное программирование – технология программирования, при которой программа рассматривается как набор дискретных объектов, содержащих, в свою очередь, наборы структур данных и процедур, взаимодействующих с другими объектами.

Объектно-ориентированное сборочное программирование – разновидность сборочного программирования: - основанная на методологии объектно-ориентированного программирования; и - предполагающая распространение библиотек классов в виде исходного кода (obj) или упаковку классов в динамически компонуемую библиотеку (dll).

Полиморфизм, Polymorphism – в объектно-ориентированном программировании – способность объекта выбирать правильный метод в зависимости от типа данных, полученных в сообщении.

Примитивные конструкторы – эти операции создают объекты, соответствующего им типа, не используя никаких объектов в качестве аргументов. Примером такой операции является создание пустого списка.

Процедурная абстракция, процедура – наиболее известный в программировании тип абстракции. Всякий, кто применял для выполнения функции подпрограмму, реализовывал тем самым процедурную абстракцию. Процедуры объединяют в себе методы абстракции через параметризацию и спецификацию, позволяя абстрагировать отдельную операцию или событие.

Свойство объекта – в объектно-ориентированном программировании – характеристика объекта. Обычно свойства изменяются с помощью методов.

Программный сниппет – (англ. snippet — фрагмент, отрывок) в практике программирования — небольшой фрагмент исходного кода или текста, пригодный для повторного использования. Сниппеты не являются заменой процедур, функций или других

подобных понятий структурного программирования. Они обычно используются для более лёгкой читаемости кода функций, которые без их использования выглядят слишком перегруженными деталями, или для устранения повторения одного и того же общего участка кода. Интегрированные среды разработки (IDE) содержат встроенные средства для ввода конструкций языка. Например, в Microsoft Visual Studio, Borland Developer Studio, для этого необходимо ввести ключевое слово и нажать определённую клавишную комбинацию. В IDE Geany существует специальный файл snippets.conf (путь к файлу: /home/user/.config/geany) позволяющий создавать свои сниппеты. Другие программы, такие как Macromedia Dreamweaver и Zend Studio, позволяют использовать сниппеты в Веб-программировании.

Событийно-управляемое программирование – объектно-ориентированное программирование, при котором задаются реакции программы на различные события.

Спецификация – описывает соглашение между разработчиками и пользователями. Разработчик берётся написать модуль, а пользователь соглашается не полагаться на знания о том, как именно этот модуль реализован, т.е. не предполагать ничего такого, что не было бы указано в спецификации. Такое соглашение позволяет разделить анализ реализации от собственно использования программы. Спецификации дают возможность создавать логические основы, позволяющие успешно "разделять и властвовать".

Технология программирования, Инжиниринг ПО, Software engineering – дисциплина, изучающая технологические процессы программирования и порядок их прохождения.

Экземпляр объекта, Instance – в объектно-ориентированном программировании – конкретный объект из набора объектов данного класса. Все экземпляры одного класса имеют одинаковый набор операций.