**Правительство Российской Федерации**

**Санкт-Петербургский государственный университет**

**Р А Б О Ч А Я П Р О Г Р А М М А**

**УЧЕБНОЙ ДИСЦИПЛИНЫ**

Программирование

Software Design

**Язык(и) обучения**

русский

Трудоемкость в зачетных единицах: 8

Регистрационный номер рабочей программы: 002212

Санкт-Петербург

2020

**Раздел 1. Характеристики учебных занятий**

**1.1. Цели и задачи учебных занятий**

Цель изучения дисциплины: получение обучающимися основных навыков программирования. После изучения дисциплины обучающиеся должны обладать навыками, достаточными для эффективного решения реализационных задач в научно-исследовательских и промышленных проектах под руководством опытного разработчика, а также быть способными выполнять практическую часть учебных и производственных практик, выпускной квалификационной работы на высоком техническом уровне.

Задачи обучения:

1. Приобретение практических навыков программирования в структурном и объектно-ориентированном стиле.
2. Владение классическими алгоритмами и структурами данных, понимание сложности алгоритмов, умение выбрать алгоритм или структуру данных, наиболее пригодные для решения задачи.
3. Приобретение навыков кодирования в соответствии со стилями кодирования, навыков грамотного оформления исходного кода.
4. Умение использовать системы контроля версий, системы сборки, системы непрерывной интеграции, системы и библиотеки модульного тестирования при разработке программных проектов. Понимание назначения и места этих инструментов в процессе разработки.
5. Свободное владение хотя бы одним объектно-ориентированным языком программирования.
6. Навыки отладки, тестирования, статического анализа и документирования программ.
7. Понимание основных концепций архитектуры программного обеспечение, владение ключевыми шаблонами объектно-ориентированного проектирования.
8. Навыки многопоточного программирования, программирования сетевых приложений, веб-приложений, настольных приложений с пользовательским интерфейсом.
9. Владение методологией эксперимента в области программирования, умение корректно выполнить замеры.
10. Понимание гибких методологий разработки программного обеспечения.

Дисциплина реализуется в нескольких вариантах, которые, помимо решения перечисленных тут задач обучения, направлены на удовлетворение индивидуальных образовательных потребностей обучающихся и позволяют им с ранних курсов специализироваться в различных направлениях программной инженерии и компьютерных наук. Варианты реализации читаются на примере разных языков программирования и имеют разные программы, ориентированные на разные уровни подготовленности и способностей обучающихся. Аннотации вариантов реализации и программы каждого варианта приведены в разделе 2.2. Распределение обучающихся по вариантам реализации выполняется на основании их личных предпочтений с учётом результатов входного тестирования.

**1.2. Требования подготовленности обучающегося к освоению содержания учебных занятий (пререквизиты)**

Программа курса предназначена для обучающихся 1 и 2 курса бакалавриата. Предварительные знания не требуются. Для обучающихся, имеющих начальную подготовку в области программирования, существуют варианты реализации, позволяющие им эффективно использовать свои знания, для обучающихся, не имеющих начальных знаний, существуют варианты реализации, начинающие с нуля.

**1.3. Перечень результатов обучения (learning outcomes)**

Результатами обучения являются знания, умения и навыки, соответствующие задачам обучения, а также представление о возможностях применения этих знаний, умений и навыков в профессиональной деятельности.

Курс способствует формированию следующих компетенций:

* ОПК-2 – способен применять современный математический аппарат, связанный с проектированием, разработкой, реализацией и оценкой качества программных комплексов в различных областях человеческой деятельности;
* ОПК-4 – способен участвовать в разработке технической документации программных продуктов и программных комплексов;
* ОПК-5 – способен инсталировать и сопровожать программное обеспеченение для информационных систем и баз данных, в том числе отечественного производства;
* ПКП-3 – способен преподавать математику и информатику в средней школе, специальных учебных заведениях на основе полученного фундаментального образования и научного мировоззрения;
* ПКП-4 – способен применять современные информационные технологии при проектировании, реализации, оценке качества и анализа эффективности программного обеспечения для решения задач в различных предметных областях;
* УКБ-3 – способен понимать сущность и значение информации в развитии общества, использовать основные методы получения и работы с информацией с учетом современных технологий цифровой экономики и информационной безопасности.

**1.4. Перечень и объём активных и интерактивных форм учебных занятий**

Основной формой обучения программированию являются практические занятия в аудитории, проводимые в активной форме: в форме работы над задачами, в том числе групповой, в форме мозгового штурма, а также в форме активных лекций, предполагающих дискуссию с преподавателем. Также дисциплина предполагает занятия в виде докладов, делаемых обучающимися.

Общий объём активных и интерактивных форм учебных занятий составляет 40 часов в первом семестре, 10 часов во втором семестре, 30 часов в третьем семестре.

**Раздел 2. Организация, структура и содержание учебных занятий**

**2.1. Организация учебных занятий**

**2.1.1 Основной курс**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Трудоёмкость, объёмы учебной работы и наполняемость групп обучающихся | | | | | | | | | | | | | | | | | | |
| Код модуля в составе дисциплины,  практики и т.п. | Контактная работа обучающихся с преподавателем | | | | | | | | | | | | Самостоятельная работа | | | | Объём активных и интерактивных  форм учебных занятий | Трудоёмкость |
| лекции | семинары | консультации | практические  занятия | лабораторные работы | контрольные работы | коллоквиумы | текущий контроль | промежуточная  аттестация | итоговая аттестация | под руководством преподавателя | в присутствии  преподавателя | сам. раб. с использованием  методических материалов | текущий контроль (сам.раб.) | промежуточная аттестация (сам.раб.) | итоговая аттестация  (сам.раб.) |
| ОСНОВНАЯ ТРАЕКТОРИЯ | | | | | | | | | | | | | | | | | | |
| Форма обучения: очная | | | | | | | | | | | | | | | | | | |
| Семестр 1 |  |  |  | 43 | 15 | 4 |  |  | 2 |  |  |  | 75 |  | 5 |  | 40 | 4 |
|  |  |  |  | 2-15 | 2-15 | 2-15 |  |  | 1-15 |  |  |  | 1-1 |  | 1-1 |  |  |  |
| Семестр 2 |  |  |  | 11 | 15 | 4 |  |  | 2 |  |  |  | 32 |  | 8 |  | 10 | 2 |
|  |  |  |  | 2-15 | 2-15 | 2-15 |  |  | 1-15 |  |  |  | 1-1 |  | 1-1 |  |  |  |
| Семестр 3 |  |  |  | 15 | 15 |  |  |  | 2 |  |  |  | 37 |  | 3 |  | 30 | 2 |
|  |  |  |  | 2-15 | 2-15 |  |  |  | 1-15 |  |  |  | 1-1 |  | 1-1 |  |  |  |
| ИТОГО |  |  |  | 69 | 45 | 8 |  |  | 6 |  |  |  | 144 |  | 16 |  |  | 8 |

|  |  |  |  |
| --- | --- | --- | --- |
| Виды, формы и сроки текущего контроля успеваемости и промежуточной аттестации | | | |
| Период обучения (модуль) | Формы текущего контроля успеваемости | Виды промежуточной аттестации | Виды итоговой аттестации  (только для программ итоговой аттестации и дополнительных образовательных программ) |
| ОСНОВНАЯ ТРАЕКТОРИЯ | | | |
| Форма обучения: очная | | | |
| Семестр 1 |  | зачёт |  |
| Семестр 2 |  | зачёт |  |
| Семестр 3 |  | зачёт |  |

**2.2. Структура и содержание учебных занятий**

Структура и содержание учебных занятий зависят от варианта реализации дисциплины. Обучающимся предлагается семь вариантов реализации:

1. Промышленное программирование, C, C# — данный вариант ориентирован на обучающихся без предварительных знаний и нацелен на быстрое погружение «с нуля» в программирование с основным упором на индустриальные практики. Курс знакомит с программированием в структурном стиле на языке C, практически в самом начале вводятся системы контроля версий, стиль кодирования, документирование и тестирование ПО. Алгоритмическая теория используется прежде всего как материал, на котором можно «набить руку», однако, несмотря на вспомогательную роль в данном курсе, преподаётся в первом семестре довольно плотно — в результате обучающиеся могут самостоятельно реализовать на C классические структуры данных и алгоритмы, включая списки, хеш-таблицы, деревья (в том числе, самобалансирующиеся), графы. Со второго семестра преподаётся объектно-ориентированное программирование на примере языка C# и различные прикладные технологии, включая модульное тестирование и разработку пользовательских интерфейсов. После этой части курса обучающиеся способны, в частности, разработать несложную игру на движке Unity. В третьем семестре также на примере C# и платформы .NET преподаётся многопоточное программирование, сетевое программирование, работа с базами данных, веб-программирование. После третьего семестра обучающиеся обладают минимальным набором знаний, необходимых для решения практических задач в профессиональной деятельности. Данный трек рассчитан на продолжение в 4-м семестре в рамках дисциплины «Структуры и алгоритмы компьютерной обработки данных», где материал курса дополняется введением в функциональное программирование на примере языка F#, а также рассмотрением дополнительных прикладных задач, что позволяет после второго курса сформировать у обучающихся широкий кругозор, богатый инструментарий навыков и относительно целостную, хоть и неглубокую, картину мира программной инженерии. Данный вариант реализации отличается большим количеством теоретического материала, сообщаемого на занятиях, и большим количеством домашних заданий, проверяемых индивидуально, прежде всего на соответствие хорошим практикам индустриального программирования, и, несмотря на то, что ориентирован на обучающихся без предварительных знаний, требует большого количества усилий и самодисциплины для освоения.

Распределение обучающихся по вариантам реализации выполняется на основании их личных предпочтений с учётом результатов входного тестирования.

**Вариант реализации 1: промышленное программирование, C, C#.**

Период обучения (модуль): семестр 1.

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Наименование темы (раздела, части) | Вид учебных занятий | Количество часов |
| I. | Введение в программирование | практические занятия | 13 |
| лабораторные работы | 5 |
| контрольные работы | 2 |
| самостоятельная работа | 20 |
| II. | «Динамические» структуры данных | практические занятия | 20 |
| лабораторные работы | 4 |
| контрольные работы | 2 |
| самостоятельная работа | 40 |
| III. | Парадигмы программирования, дополнительные алгоритмы | практические занятия | 10 |
| лабораторные работы | 6 |
| самостоятельная работа | 15 |
| IV. | Промежуточная аттестация | самостоятельная работа | 5 |
| зачёт | 2 |

Раздел 1: Введение в программирование.

1. Введение в C, структура программы, основные языковые конструкции (функции и рекурсия, переменные, элементарные типы и арифметические операции, массивы, строки, ввод-вывод с консоли), среда разработки.
2. Стиль кодирования. Процесс сборки и запуска программы — компилятор, линковщик, IDE. Практика, написание первых программ на C. После этого занятия следование стилю кодирования обязательно для зачёта заданий.
3. Понятие сложности алгоритмов, O-символика, основные техники оценки трудоёмкости программы. Примеры задач, имеющих алгоритмы с разной трудоёмкостью: сортировки (пузырьком и подсчётом), вычисление факториала (рекурсивный и итеративный варианты алгоритма), вычисление чисел Фибоначчи (рекурсивно, итеративно и через возведение в степень матрицы 2х2 специального вида).
4. Отладка и тестирование. Практика по пользованию отладчиком. Понятие модульных тестов. После этого занятия написание тестов к программам обязательно.
5. Сортировки: вставкой, выбором, сортировка Шелла, быстрая сортировка, сортировка кучей. Оценки трудоёмкости. Двоичный поиск.
6. Системы контроля версий, git. Основные команды. Внутреннее устройство репозитория, коммиты, ветки, слияние веток, конфликты, работа с удалёнными репозиториями. Git flow. Хорошие практики использования систем контроля версий. Графические клиенты git. С этого занятия все домашние работы сдаются в виде пуллреквестов на сервисе GitHub.
7. Внутреннее представление данных. Целые числа, вещественные числа (IEEE 754), строки (ASCII, Unicode и UTF-8
8. Структуры, указатели, модули и файлы. Общие правила разбиения на модули, модули в C. Практика по разрешению конфликтов в git.
9. О разработке программных продуктов. Понятие жизненного цикла программного обеспечения, фазы жизненного цикла, водопадная и спиральная модели разработки. Понятие методологии разработки, понятие парадигмы программирования. Git flow.
10. Контрольная работа.

Раздел 2: «Динамические» структуры данных.

1. Стек и очередь на указателях. Практика по написанию стека.
2. Списки на указателях, односвязные, двусвязные и циклические. Практика по написанию односвязного списка.
3. Понятие абстрактного типа данных. Стек и список как абстрактные типы данных. Понятие инварианта. Пример применения АТД для реализации «полиморфного» кода — сортировка слиянием.
4. Переписывание контрольной работы.
5. Деревья. Дерево как математический объект и как АТД, обходы деревьев. Деревья выражений, двоичные деревья. Пример: алгоритм Хаффмана. Реализация деревьев. Деревья поиска, основные операции.
6. Практика по написанию двоичного дерева поиска в командах с использованием GitHub.
7. Самобалансирующиеся деревья. АВЛ-дерево, балансировка, основные операции. Красно-чёрные деревья. Splay-деревья. Декартовы деревья.
8. Переписывание контрольной.
9. Хеш-таблицы. Хеш-функции, их выбор и свойства, совершенные хеш-функции, универсальные хеш-функции, комбинирование хеш-функций. Варианты реализации хеш-таблиц (списки значений и открытая адресация), их свойства.
10. Работа с консолью, системы сборки. Либо командная практика по написанию хеш-таблицы, в зависимости от успехов группы.
11. Графы. Граф как математический объект и как АТД, способы представления графа: матрица смежности, матрица инцидентности, список смежности. Задача достижимости, обходы в глубину и в ширину. Проверка на ацикличность, глубинное остовное дерево. Задача поиска кратчайшего пути, алгоритм Дейкстры. Алгоритм Флойда-Уоршелла. О связи графов и бинарных отношений над множеством, топологическая сортировка.
12. Практика: реализация обхода графа в ширину.
13. Контрольная работа.

Раздел 3: Парадигмы программирования, дополнительные алгоритмы.

1. Парадигмы программирования. Структурное программирование: машины Тьюринга, архитектура фон Неймана, языки-представители, подробнее про Ada. Объектно-ориентированное программирование, основные понятия, инкапсуляция, наследование, полиморфизм. Языки-представители. Пример объектно-ориентированного кода на C++. Функциональное программирование. Понятие лямбда-исчисления, основные принципы и особенности функционального программирования. Языки представители, подробнее Haskell, F#.
2. Доклады: алгоритмы поиска подстроки в строке (Кнута-Морриса-Пратта, Рабина-Карпа, Бойера-Мура), алгоритм A\*, пакет визуализации графов GraphViz и язык dot.
3. Парадигмы программирования-2. Логическое программирование, Пролог. Рекурсивное программирование, Рефал. Стековое программирование, Форт. Визуальное программирование, визуальное моделирование, UML, предметно-ориентированное моделирование.
4. Автоматы, лексический анализ. Назначение лексических анализаторов, языки и регулярные выражения, диаграммы переходов, построение кода по диаграммам, НКА, ДКА, построение НКА по регулярному выражению, работа ДКА с таблицей переходов.
5. Практика, написание ДКА для несложного регулярного выражения.
6. Переписывание контрольной работы.
7. Первая попытка написания зачётной работы.
8. Вторая попытка написания зачётной работы.

Период обучения (модуль): семестр 2.

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Наименование темы (раздела, части) | Вид учебных занятий | Количество часов |
| I. | Объектно-ориентированное программирование | практические занятия | 4 |
| лабораторные работы | 0 |
| самостоятельная работа | 6 |
| II. | Качество программного обеспечения: модульные тесты, обработка ошибок, непрерывная интеграция, инструменты разработки и управления проектами | практические занятия | 6 |
| лабораторные работы | 0 |
| самостоятельная работа | 8 |
| III. | Аспекты современных языков программирования: событийно-ориентированное программирование, генерики | практические занятия | 1 |
| лабораторные работы | 5 |
| контрольные работы | 2 |
| самостоятельная работа | 8 |
| IV. | Элементы архитектуры программного обеспечения | практические занятия | 0 |
| лабораторные работы | 10 |
| контрольные работы | 2 |
| самостоятельная работа | 10 |
| V. | Промежуточная аттестация | самостоятельная работа | 8 |
| зачёт | 2 |

Раздел 1: Объектно-ориентированное программирование.

1. Введение в C#, понятие байт-кода и виртуальной машины, CIL, сборка мусора. Основы синтаксиса C# — структура программы, методы, элементарные типы, массивы, перечисления, структуры, стиль кодирования.
2. Объектно-ориентированное программирование: основные понятия, объекты и классы, инварианты. Абстракция, инкапсуляция, наследование, полиморфизм, типы времени компиляции и времени выполнения. Абстрактные классы. ООП в C#: ссылочные типы и типы-значения, передача параметров по ссылке, конструкторы, наследование, интерфейсы, абстрактные классы, виртуальные методы, абстрактные методы, перевведение методов. Модификаторы видимости, модификаторы partial, sealed, static. Вложенные классы, преобразования типов, иерархия основных классов стандартной библиотеки. Представление объектов в памяти.

Раздел 2: Качество программного обеспечения: модульные тесты, обработка ошибок, непрерывная интеграция, инструменты разработки и управления проектами.

1. Комментирование кода на C#, XML Documentation. Модульное тестирование. Тестовые сценарии, примеры. Модульные тесты в C#: NUnit, Microsoft Unit Test Framework, демонстрация по использованию NUnit. Хорошие практики модульного тестирования. Data-driven-тесты.
2. Исключения и обработка ошибок. Бросание и обработка исключений, try-catch-finally, фильтры исключений. Иерархия классов-исключений стандартной библиотеки. Свойства класса Exception. Перебрасывание исключений. Объявление своих классов-исключений. «Интересные» классы-исключения стандартной библиотеки.
3. Экосистема проектов с открытым исходным кодом. Непрерывная интеграция: задачи, облачный сервис AppVeyor, настройка сборки, матрица сборки. Облачный сервис Travis. Инструменты анализа качества: CodeCov, Codacy. Инструменты планирования и управления проектом: Trello, Pivotal Tracker. Средства коммуникации: Slack, Gitter. Багтрекер GitHub Issues. Другие средства управления проектом GitHub. Авторское право и лицензии.

Раздел 3: Аспекты современных языков программирования: событийно-ориентированное программирование, генерики.

1. Событийно-ориентированное программирование. Паттерн «Наблюдатель». События в .NET: делегаты, обработчики, генерики Func и Action. События. Анонимные методы и замыкания. Лямбда-выражения. Объявление событий в .NET. Ручное управление подпиской на события (ключевые слова add и remove).
2. Пользовательский интерфейс. Библиотека Windows Forms: обзор библиотеки, класс Control, обработка пользовательского ввода, валидация, привязка данных, хорошие практики. Мастер-класс по разработке приложения на Windows Forms.
3. Контейнеры и генерики. Контейнеры стандартной библиотеки .NET, паттерн «Итератор», энумераторы. Типизация, понятия ad-hoc и универсального полиморфизма, полиморфизм подтипов и параметрический полиморфизм. Генерики в .NET, внутреннее устройство и использование. Генерик-методы и генерик-классы. Открытые и закрытые типы, генерики и вложенные классы. Ограничения на типы. Понятие вариантности, ковариантность и контравариантность, примеры.
4. Контрольная работа.

Раздел 4: Элементы архитектуры программного обеспечения.

1. Визуальное моделирование, UML. Метафора моделирования, цель моделирования. Диаграммы UML. Диаграмма классов: синтаксис, синтаксис свойств, агрегация и композиция. Диаграмма компонентов. Диаграмма случаев использования. Диаграммы активностей, последовательностей, конечных автоматов. Генерация кода по диаграммам конечных автоматов. Диаграммы развёртывания. Примеры CASE-инструментов. Предметно-ориентированные визуальные языки.
2. Переписывание контрольной.
3. Доклады.
4. Правила написания хорошего кода в ООП. Сложность ПО, сопряжение и связность, модульность, свойства модулей. Понятие объекта с точки зрения архитектуры ПО, выделение объектов. Наследование и композиция. Принципы SOLID, закон Деметры. Примеры абстракций: принцип единственности ответственности, уровень абстракции, общие рекомендации по дизайну абстракций. Инкапсуляция: принцип минимизации доступности, общие рекомендации. Наследование, общие рекомендации, конструкторы. Мутабельность. Преждевременная оптимизация. Принцип Fail Fast. Другие рекомендации.
5. Первая попытка написания зачётной работы.
6. Вторая попытка написания зачётной работы

Период обучения (модуль): семестр 3.

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Наименование темы (раздела, части) | Вид учебных занятий | Количество часов |
| I. | Многопоточное программирование | практические занятия | 4 |
| лабораторные работы | 4 |
| самостоятельная работа | 10 |
| II. | Сетевое программирование | практические занятия | 3 |
| лабораторные работы | 3 |
| самостоятельная работа | 8 |
| III. | Рефлексия | практические занятия | 2 |
| лабораторные работы | 2 |
| самостоятельная работа | 5 |
| IV. | Базы данных | практические занятия | 1 |
| лабораторные работы | 1 |
| самостоятельная работа | 3 |
| V. | Современные пользовательские интерфейсы | практические занятия | 2 |
| лабораторные работы | 2 |
| самостоятельная работа | 5 |
| VI. | Веб-программирование | практические занятия | 3 |
| лабораторные работы | 3 |
| самостоятельная работа | 6 |
| VII. | Промежуточная аттестация | самостоятельная работа | 3 |
| зачёт | 2 |

Раздел 1: Многопоточное программирование.

1. Многопоточное программирование: предназначение, потенциальные проблемы. Понятия процесса и потока, многопроцессной и многопоточной программы. Планировщик, пример: планировщик и представление потока в ОС Windows. Поток как абстракция длительной задачи и поток как абстракция вычислителя. Потоки в .NET: Thread, пример гонки по данным. Race condition и deadlock. Понятия конвейера процессора, процессорного кеша.
2. Примитивы синхронизации, синхронизация режима ядра и режима пользователя. Атомарные операции, Volatile. Понятие модели памяти. Критические области. Активное ожидание. Задача производителя и потребителя. Семафоры, мьютексы, мониторы, conditional variables (события).
3. Практическое занятие: моделирование классической проблемы «Обедающие философы».
4. Высокоуровневая многопоточность в .NET. Foreground- и Background-потоки, пул потоков, классы Task, CancellationToken, TaskScheduler. Модель асинхронного программирования C#: async/await, примеры, потенциальные проблемы, связь с Task. Task Parallel Library. Потокобезопасные и немутабельные коллекции .NET.

Раздел 2: Сетевое программирование.

1. Работа с сетью, низкий уровень. Архитектура глобальных сетей, модель OSI, стек протоколов TCP/IP, обзор уровней модели OSI. Протокол IP: IP-адреса, формат пакета. DNS, NAT. Порты и сокеты. Консольные утилиты для работы с сетью. Работа с сетью в .NET, минимальный пример клиент-серверного приложения на сокетах, асинхронный клиент и сервер. Работа с UDP.
2. Работа с сетью в .NET на высоком уровне. Протоколы прикладного уровня, HTTP. Класс HttpClient. Веб-сервисы, архитектурный стиль REST, пример: Google Drive API. Отладка веб-сервисов. Основы сетевой безопасности.
3. Практическое занятие: клиент для социальной сети «ВКонтакте».

Раздел 3: Рефлексия.

1. Рефлексия, вообще и в .NET. Сборки в .NET, сильные и слабые имена сборок. Получение информации о сборках, типах, полях, методах и т.д., создание экземпляров объектов. Атрибуты. Пример использования рефлексии: библиотеки сериализации (System.Xml.Serialization, Newtonsoft.Json). Ключевое слово dynamic. Генерация кода посредством ILGenerator.
2. Проверочная работа.

Раздел 4: Базы данных.

1. Базы данных, виды СУБД. Реляционная модель данных, язык SQL, ключи и ограничения, основные операции над данными. Низкоуровневая работа с данными из .NET, ADO.NET. Высокоуровневая работа с данными, Entity Framework.

Раздел 5: Современные пользовательские интерфейсы.

1. Библиотека WPF, обзор библиотеки. Язык разметки интерфейсов XAML. Архитектура WPF. Зависимые свойства и маршрутизируемые события. Привязка данных, свойство DataContext, конвертеры. Паттерн «Model-View-ViewModel». Геометрия элементов пользовательского интерфейса.
2. Паттерн «Команда», команды в WPF. Валидация пользовательского ввода. Шаблоны и стили визуального оформления, триггеры. Ресурсы. Практика: GUI для клиента «ВКонтакте».

Раздел 6: Веб-программирование.

1. Веб-программирование. Архитектура типичных веб-приложений, веб-приложения и веб-сервисы. Фронтенд: язык HTML, DOM, CSS, азы JavaScript, асинхронные запросы. Бэкенд: механизм обработки HTTP-запросов в Windows, ASP.NET MVC. Язык описания правил генерации Razor. Роутинг запросов. Привязка моделей.
2. Практический пример: приложение для регистрации на конференцию. Модель, вёрстка, работа с СУБД, Bootstrap, валидация.
3. Проверочная работа.

В целях оперативной актуализации программы обучения допустимы отклонения от обозначенного здесь плана занятий (вплоть до полной замены темы тех или иных занятий) при условии сохранения общей структуры курса и следования учебному плану.

**Вариант реализации 2: …**

Период обучения (модуль): семестр 1.

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Наименование темы (раздела, части) | Вид учебных занятий | Количество часов |
| I. |  | практические занятия |  |
| лабораторные работы |  |
| контрольные работы |  |
| N. | Промежуточная аттестация | зачёт | 2 |

Раздел 1:

Раздел 2:

Период обучения (модуль): семестр 2.

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Наименование темы (раздела, части) | Вид учебных занятий | Количество часов |
| I. |  | практические занятия |  |
| лабораторные работы |  |
| контрольные работы |  |
| N. | Промежуточная аттестация | зачёт | 2 |

Раздел 1:

Раздел 2:

Период обучения (модуль): семестр 3.

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Наименование темы (раздела, части) | Вид учебных занятий | Количество часов |
| I. |  | практические занятия |  |
| лабораторные работы |  |
| контрольные работы |  |
| N. | Промежуточная аттестация | зачёт | 2 |

Раздел 1:

Раздел 2:

**Вариант реализации 3: …**

Период обучения (модуль): семестр 1.

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Наименование темы (раздела, части) | Вид учебных занятий | Количество часов |
| I. |  | практические занятия |  |
| лабораторные работы |  |
| контрольные работы |  |
| N. | Промежуточная аттестация | зачёт | 2 |

Раздел 1:

Раздел 2:

Период обучения (модуль): семестр 2.

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Наименование темы (раздела, части) | Вид учебных занятий | Количество часов |
| I. |  | практические занятия |  |
| лабораторные работы |  |
| контрольные работы |  |
| N. | Промежуточная аттестация | зачёт | 2 |

Раздел 1:

Раздел 2:

Период обучения (модуль): семестр 3.

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Наименование темы (раздела, части) | Вид учебных занятий | Количество часов |
| I. |  | практические занятия |  |
| лабораторные работы |  |
| контрольные работы |  |
| N. | Промежуточная аттестация | зачёт | 2 |

Раздел 1:

Раздел 2:

**Вариант реализации 4: …**

Период обучения (модуль): семестр 1.

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Наименование темы (раздела, части) | Вид учебных занятий | Количество часов |
| I. |  | практические занятия |  |
| лабораторные работы |  |
| контрольные работы |  |
| N. | Промежуточная аттестация | зачёт | 2 |

Раздел 1:

Раздел 2:

Период обучения (модуль): семестр 2.

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Наименование темы (раздела, части) | Вид учебных занятий | Количество часов |
| I. |  | практические занятия |  |
| лабораторные работы |  |
| контрольные работы |  |
| N. | Промежуточная аттестация | зачёт | 2 |

Раздел 1:

Раздел 2:

Период обучения (модуль): семестр 3.

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Наименование темы (раздела, части) | Вид учебных занятий | Количество часов |
| I. |  | практические занятия |  |
| лабораторные работы |  |
| контрольные работы |  |
| N. | Промежуточная аттестация | зачёт | 2 |

Раздел 1:

Раздел 2:

**Вариант реализации 5: …**

Период обучения (модуль): семестр 1.

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Наименование темы (раздела, части) | Вид учебных занятий | Количество часов |
| I. |  | практические занятия |  |
| лабораторные работы |  |
| контрольные работы |  |
| N. | Промежуточная аттестация | зачёт | 2 |

Раздел 1:

Раздел 2:

Период обучения (модуль): семестр 2.

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Наименование темы (раздела, части) | Вид учебных занятий | Количество часов |
| I. |  | практические занятия |  |
| лабораторные работы |  |
| контрольные работы |  |
| N. | Промежуточная аттестация | зачёт | 2 |

Раздел 1:

Раздел 2:

Период обучения (модуль): семестр 3.

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Наименование темы (раздела, части) | Вид учебных занятий | Количество часов |
| I. |  | практические занятия |  |
| лабораторные работы |  |
| контрольные работы |  |
| N. | Промежуточная аттестация | зачёт | 2 |

Раздел 1:

Раздел 2:

**Вариант реализации 6: …**

Период обучения (модуль): семестр 1.

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Наименование темы (раздела, части) | Вид учебных занятий | Количество часов |
| I. |  | практические занятия |  |
| лабораторные работы |  |
| контрольные работы |  |
| N. | Промежуточная аттестация | зачёт | 2 |

Раздел 1:

Раздел 2:

Период обучения (модуль): семестр 2.

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Наименование темы (раздела, части) | Вид учебных занятий | Количество часов |
| I. |  | практические занятия |  |
| лабораторные работы |  |
| контрольные работы |  |
| N. | Промежуточная аттестация | зачёт | 2 |

Раздел 1:

Раздел 2:

Период обучения (модуль): семестр 3.

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Наименование темы (раздела, части) | Вид учебных занятий | Количество часов |
| I. |  | практические занятия |  |
| лабораторные работы |  |
| контрольные работы |  |
| N. | Промежуточная аттестация | зачёт | 2 |

Раздел 1:

Раздел 2:

**Вариант реализации 7: …**

Период обучения (модуль): семестр 1.

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Наименование темы (раздела, части) | Вид учебных занятий | Количество часов |
| I. |  | практические занятия |  |
| лабораторные работы |  |
| контрольные работы |  |
| N. | Промежуточная аттестация | зачёт | 2 |

Раздел 1:

Раздел 2:

Период обучения (модуль): семестр 2.

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Наименование темы (раздела, части) | Вид учебных занятий | Количество часов |
| I. |  | практические занятия |  |
| лабораторные работы |  |
| контрольные работы |  |
| N. | Промежуточная аттестация | зачёт | 2 |

Раздел 1:

Раздел 2:

Период обучения (модуль): семестр 3.

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Наименование темы (раздела, части) | Вид учебных занятий | Количество часов |
| I. |  | практические занятия |  |
| лабораторные работы |  |
| контрольные работы |  |
| N. | Промежуточная аттестация | зачёт | 2 |

Раздел 1:

Раздел 2:

**Раздел 3. Обеспечение учебных занятий**

**3.1. Методическое обеспечение**

**3.1.1 Методические указания по освоению дисциплины**

Для успешного усвоения дисциплины необходимо посещение занятий и самостоятельная работа с материалами и рекомендованной литературой.

**3.1.2 Методическое обеспечение самостоятельной работы**

Для самостоятельной работы рекомендуется использовать рекомендованную литературу и материалы курса (презентации, конспекты), размещаемые в системе поддержки обучения.

**3.1.3 Методика проведения текущего контроля успеваемости и промежуточной аттестации и критерии оценивания**

***3.1.3.1. Методика проведения текущего контроля успеваемости и промежуточной аттестации***

В начале курса проводится входное тестирование, по результатам которого и на основании личных предпочтений обучающиеся распределяются на варианты реализации дисциплины. Методика проведения входного тестирования такова: … (Blackboard, скорее всего. Одна-две пары, открытые вопросы и задачи).

В формировании итоговой оценки по курсу участвуют домашние задания, контрольные работы, зачётная работа, оценки за работы на практиках (включая доклады). Конкретные критерии оценивания и методики проведения текущего контроля и промежуточной аттестации варьируются в зависимости от варианта реализации дисциплины, чтобы учесть особенности программы и индивидуальные особенности обучающихся. Методики проведения текущего контроля и промежуточной аттестации по вариантам реализации представлены ниже.

**Вариант реализации 1: промышленное программирование, C, C#.**

Тут будет методика проверки домашек, регламент контрольных и докладов, регламент зачёта.

**Вариант реализации 2: …**

**Вариант реализации 3: …**

**Вариант реализации 4: …**

**Вариант реализации 5: …**

**Вариант реализации 6: …**

**Вариант реализации 7: …**

***3.1.3.2. Критерии оценивания итогового процента освоения дисциплины***

Критерии оценивания итогового процента освоения дисциплины варьируются в зависимости от варианта реализации дисциплины, чтобы учесть особенности программы и индивидуальные особенности обучающихся. Критерии по вариантам реализации представлены ниже.

**Вариант реализации 1: промышленное программирование, C, C#.**

Тут будет написано, как получаем итоговую оценку по шкале от 0 до 100. Скорее всего, это будет сложная шкала с «мажорирующими двойками», которая не позволяет аттестовать обучающегося, который не сдал зачёт или 95% домашек, и которая хитро вычисляет итоговую оценку ECTS по баллам за домашки, которые сами зависят от дедлайнов, количества попыток и общего качества задач (при этом каждая задача оценивается по своей шкале и своим критериям, описанным в разделе 3.1.4.2 вместе с условиями задач).

**Вариант реализации 2: …**

**Вариант реализации 3: …**

**Вариант реализации 4: …**

**Вариант реализации 5: …**

**Вариант реализации 6: …**

**Вариант реализации 7: …**

Вне зависимости от варианта реализации дисциплины перед вычислением итогового процента освоения баллы за дисциплину приводятся к диапазону 0–100 линейным преобразованием. Далее применяется следующее правило выставления оценки:

|  |  |  |
| --- | --- | --- |
| Итоговый процент  выполнения, % | Оценка СПбГУ при  проведении зачёта | Оценка  ECTS |
| 90-100 | зачтено | A |
| 80-89 | зачтено | B |
| 70-79 | зачтено | C |
| 61-69 | зачтено | D |
| 50-60 | зачтено | E |
| менее 50 | не зачтено | F |

**3.1.4 Методические материалы для проведения текущего контроля успеваемости и промежуточной аттестации (контрольно-измерительные материалы, оценочные средства)**

***3.1.4.1. Формируемые дисциплиной компетенции***

**ОПК-2** — способен применять современный математический аппарат, связанный с проектированием, разработкой, реализацией и оценкой качества программных комплексов в различных областях человеческой деятельности.

✓ Формируется дисциплиной.

□ Развивается дисциплиной.

□ Полностью сформирована по результатам освоения дисциплины.

Шкала оценивания: линейная, определяется долей успешно выполненных заданий, проверяющих данную компетенцию.

**ОПК-4** — способен участвовать в разработке технической документации программных продуктов и программных комплексов.

✓ Формируется дисциплиной.

□ Развивается дисциплиной.

□ Полностью сформирована по результатам освоения дисциплины.

Шкала оценивания: линейная, определяется долей успешно выполненных заданий, проверяющих данную компетенцию.

**ОПК-5** — способен инсталировать и сопровожать программное обеспеченение для информационных систем и баз данных, в том числе отечественного производства.

✓ Формируется дисциплиной.

□ Развивается дисциплиной.

□ Полностью сформирована по результатам освоения дисциплины.

Шкала оценивания: линейная, определяется долей успешно выполненных заданий, проверяющих данную компетенцию.

**ПКП-3** — способен преподавать математику и информатику в средней школе, специальных учебных заведениях на основе полученного фундаментального образования и научного мировоззрения.

✓ Формируется дисциплиной.

□ Развивается дисциплиной.

□ Полностью сформирована по результатам освоения дисциплины.

Шкала оценивания: линейная, определяется долей успешно выполненных заданий, проверяющих данную компетенцию.

**ПКП-4** — способен применять современные информационные технологии при проектировании, реализации, оценке качества и анализа эффективности программного обеспечения для решения задач в различных предметных областях.

✓ Формируется дисциплиной.

□ Развивается дисциплиной.

□ Полностью сформирована по результатам освоения дисциплины.

Шкала оценивания: линейная, определяется долей успешно выполненных заданий, проверяющих данную компетенцию.

**УКБ-3** — способен понимать сущность и значение информации в развитии общества, использовать основные методы получения и работы с информацией с учетом современных технологий цифровой экономики и информационной безопасности.

✓ Формируется дисциплиной.

□ Развивается дисциплиной.

□ Полностью сформирована по результатам освоения дисциплины.

Шкала оценивания: линейная, определяется долей успешно выполненных заданий, проверяющих данную компетенцию.

***3.1.4.2. Контрольно-измерительные материалы (примеры)***

**Контрольно-измерительные материалы входного тестирования**

Тут будут условия задач и вопросы

**Контрольно-измерительные материалы текущего контроля и промежуточной аттестации**

Контрольно-измерительные материалы различны для каждого варианта реализации дисциплины.

**Вариант реализации 1: промышленное программирование, C, C#.**

***Домашняя работа 1:***

1. Написать программу, считающую значение формулы x4 + x3 + x2 + x + 1 за два умножения.
2. Написать программу нахождения неполного частного от деления a на b (целые числа), используя только операции сложения, вычитания и умножения.
3. Дан массив целых чисел x[1] ... x[m + n], рассматриваемый как соединение двух его отрезков: начала x[1] ... x[m] длины m и конца x[m + 1] ... x[m + n] длины n. Не используя дополнительных массивов, переставить начало и конец (обращением двух частей массива, а потом его самого).
4. Посчитать число "счастливых билетов" (билет считается "счастливым", если сумма первых трёх цифр его номера равна сумме трёх последних), подсчётом числа билетов с заданной суммой трёх цифр.
5. Написать программу проверки баланса скобок в исходной строке (т.е. число открывающих скобок равно числу закрывающих и выполняется правило вложенности скобок).
6. Заданы две строки: S и S1. Найти количество вхождений S1 в S как подстроки.
7. Написать программу, печатающую все простые числа, не превосходящие заданного числа.
8. Написать программу, считающую количество нулевых элементов в массиве.

***Проверяемые компетенции*:** ОПК-2, ОПК-5, ПКП-4

***Критерии оценивания*:** решение каждой задачи оценивается по шкале от 0 (нет решения или решение имеет существенные недостатки) до 1 (решение работоспособно, аккуратно реализовано, алгоритмически аккуратно). Допустимы дробные оценки. Допустимы оценки выше максимума за особо качественное и остроумное решение.

Тут будут условия и критерии оценивания других домашек, примеры контрольных.

**Вариант реализации 2: …**

**Вариант реализации 3: …**

**Вариант реализации 4: …**

**Вариант реализации 5: …**

**Вариант реализации 6: …**

**Вариант реализации 7: …**

**3.1.5 Методические материалы для оценки обучающимися содержания и качества учебного процесса**

Для оценки обучающимися содержания и качества учебного процесса применяется анкетирование в соответствии с методикой и графиком, утвержденными в установленном порядке.

**3.2. Кадровое обеспечение**

**3.2.1 Образование и (или) квалификация штатных преподавателей и иных лиц, допущенных к проведению учебных занятий**

К проведению лекционных занятий должны привлекаться преподаватели, имеющие диплом о высшем образовании по соответствующему направлению.

**3.2.2 Обеспечение учебно-вспомогательным и (или) иным персоналом**

Не требуется.

**3.3. Материально-техническое обеспечение**

**3.3.1 Характеристики аудиторий (помещений, мест) для проведения занятий**

Требуется стандартно оборудованная аудитория с проектором.

**3.3.2 Характеристики аудиторного оборудования, в том числе неспециализированного компьютерного оборудования и программного обеспечения общего пользования**

Доска для письма маркером, мультимедийный проектор.

**3.3.3 Характеристики специализированного оборудования**

Не требуется.

**3.3.4 Характеристики специализированного программного обеспечения**

Не требуется.

**3.3.5 Перечень и объёмы требуемых расходных материалов**

Маркеры для доски, губка.

**3.4. Информационное обеспечение**

Не требуется.

**Раздел 4. Разработчики программы**

Литвинов Юрий Викторович, кандидат технических наук, доцент кафедры системного программирования, [y.litvinov@spbu.ru](mailto:y.litvinov@spbu.ru)

Брыксин Тимофей Александрович, кандидат технических наук, доцент кафедры системного программирования, [t.bryksin@spbu.ru](mailto:t.bryksin@spbu.ru)

Тут надо вписать себя