**Санкт-Петербургский государственный университет**

**Р А Б О Ч А Я П Р О Г Р А М М А**

**УЧЕБНОЙ ДИСЦИПЛИНЫ**

Практикум на ЭВМ

Computer Workshop

**Язык(и) обучения**

русский

Трудоемкость в зачетных единицах: 14

Регистрационный номер рабочей программы: 003576

2021

**Раздел 1. Характеристики учебных занятий**

**1.1. Цели и задачи учебных занятий**

Основной целью курса является обучение основам программирования, основам современного промышленного программирования, основам программирования с учётом архитектуры современных электронных вычислительных машин, архитектуре современных приложений и умению представлять полученные результаты.

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

Программа курса предназначена для обучающихся 1-2 курса бакалавриата «Программная инженерия». Максимальная эффективность программы будет обеспечена при условии, что обучающийся:

* освоил школьный курс информатики,
* успешно сдал ЕГЭ по информатике,
* способен читать тексты на техническом английском языке.

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

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Наименование категории (группы) компетенций | Код и наименование компетенции | Планируемые результаты обучения, обеспечивающие формирование компетенции | Код индикатора и индикатор достижения универсальной компетенции |
|  | 1 | 2 | 3 | 4 |
| 1 | Общепрофессиональные компетенции | ОПК-1 Способен применять естественнонаучные и общеинженерные знания, методы алгебры, математического анализа и моделирования, теоретического и экспериментального исследования в профессиональной деятельности | Ориентируется в мире программирования | ОПК-1.1 Уметь идентифицировать возможные проблемы и пути их решения |
| 2 | Общепрофессиональные компетенции | ОПК-2 Способен понимать принципы работы современных информационных технологий и программных средств, в том числе отечественного производства, и использовать их при решении задач профессиональной деятельности | Обладает знаниями в области параллельных алгоритмов и алгоритмов многопоточной синхронизации | ОПК-2.1 Уметь писать программный код с использованием языков программирования, определения и манипулирования данными |
| 3 | Общепрофессиональные компетенции | ОПК-6 Способен разрабатывать алгоритмы и программы, пригодные для практического использования, применять основы информатики и программирования к проектированию, конструированию и тестированию программных продуктов; | Владеет основами алгоритмического мышления | ОПК-6.1 Уметь проверять и отлаживать программный код |
| 4 | Общепрофессиональные компетенции | ОПК-7 Способен применять в практической деятельности основные концепции, принципы, теории и факты, связанные с информатикой | Обладает навыками работы со средствами версионного контроля | ОПК-7.1 Уметь определять качественные характеристики каждого компонента |
| 5 | Профессиональные компетенции | ПКА-1 Способен использовать в педагогической деятельности научные основы образования в сфере ИКТ | Обладает навыками командной работы в области разработки программного обеспечения | ПКА-1.1 Быть способным осуществлять организационное и технологическое обеспечение кодирования на языках программирования |
| 6 | Профессиональные компетенции | ПКП-1 Способен проектировать программные системы | Обладает навыками использования языков объектного моделирования в области разработки программного обеспечения; | ПКП-1.1 Уметь разрабатывать технических спецификаций на программные компоненты и их взаимодействие |
| 7 | Профессиональные компетенции | ПКП-2 Способен использовать основные модели информационных технологий и способы их применения для решения задач в предметных областях | Обладает навыками написания программ на современных языках программирования: императивных, функциональных и объектно-ориентированных | ПКП-2.1 Уметь описывать алгоритмы компонентов, включая методы и схемы |
| 8 | Профессиональные компетенции | ПКП-5 Способен использовать современные системные программные средства: операционные системы, операционные и сетевые оболочки, сервисные программы | Обладает навыками написания программ на современных языках программирования: императивных, функциональных и объектно-ориентированных | ПКП-5.1 Уметь выбирать технологии и средства разработки программного обеспечения |
| 9 | Профессиональные компетенции | ПКП-6 Способен формировать суждения о проблемах современной информатики, ее категорий и связей с другими научными дисциплинами | Ориентируется в мире программирования | ПКП-6.1 Уметь разрабатывать процедуры проверки работоспособности и измерения характеристик программного обеспечения |
| 10 | Профессиональные компетенции | ПКП-7 Способен систематизировать и применять знания о содержании основных этапов и тенденций развития программирования, математического обеспечения и информационных технологий | Обладает навыками тестирования программ | ПКП-7.1 Умеет проводить технические и управленческие ревизии создаваемого программного средства |
| 11 | Универсальные компетенции | УК-1 Способен осуществлять поиск, критический анализ и синтез информации, применять системный подход для решения поставленных задач | Ориентируется в мире программирования | УК 1.1. Анализирует задачу, выделяя ее базовые составляющие |
| 12 | Универсальные компетенции | 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 | Обладает навыками активного общения – как с друг другом, так и с преподавателем, публичных выступлений | УК-3.4. Осуществляет обмен информацией, знаниями и опытом с членами команды; |

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

Аудиторная учебная работа:

* Практические занятия в объеме 2 часа в неделю в периодах С1-С3 и 4 часа в неделю в период С4.

Самостоятельная работа:

* Под руководством преподавателя в объеме 2 часа в неделю в периодах С1-С3;
* С использованием методических материалов (индивидуальная работа с доступными информационными и образовательными ресурсами, имеющимися в библиотеке, в открытом доступе в сети Интернет и локальной сети Университета с целью преодоления индивидуальных трудностей в освоении отдельных разделов курса).

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

**2.1. Организация учебных занятий**

**2.1.1 Основной курс**

Таблица к УП 21/5080/1:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 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 |  |  |  | 60 |  |  |  |  | 2 |  | 2 |  | 72 |  | 8 |  | 50 | 4 |
|  |  |  |  | 2-13 |  |  |  |  | 2-13 |  | 2-13 |  | 1-1 |  | 1-1 |  |  |  |
| Семестр 2 |  |  |  | 60 |  |  |  |  | 2 |  |  |  | 36 |  | 10 |  | 50 | 3 |
|  |  |  |  | 2-13 |  |  |  |  | 2-13 |  |  |  | 1-1 |  | 1-1 |  |  |  |
| Семестр 3 |  |  |  | 60 |  |  |  |  | 2 |  |  |  | 39 |  | 7 |  | 50 | 3 |
|  |  |  |  | 2-13 |  |  |  |  | 2-13 |  |  |  | 1-1 |  | 1-1 |  |  |  |
| Семестр 4 |  |  |  | 60 |  |  |  |  | 2 |  |  |  | 72 |  | 10 |  | 50 | 4 |
|  |  |  |  | 2-13 |  |  |  |  | 2-13 |  |  |  | 1-1 |  | 1-1 |  |  |  |
| ИТОГО |  |  |  | 240 |  |  |  |  | 8 |  | 2 |  | 219 |  | 35 |  |  | 14 |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Виды, формы и сроки текущего контроля успеваемости и промежуточной аттестации | | | | | | |
| Код модуля в составе дисциплины, практики и т.п. | Формы текущего контроля успеваемости | | Виды промежуточной аттестации | | Виды итоговой аттестации  (только для программ итоговой аттестации и дополнительных образовательных программ) | |
| Формы | Сроки | Виды | Сроки | Виды | Сроки |
| ОСНОВНАЯ ТРАЕКТОРИЯ | | | | | | |
| Форма обучения: очная | | | | | | |
| Семестр 1 |  |  | зачёт, устно, традиционная форма | по графику промежуточной аттестации |  |  |
| Семестр 2 |  |  | зачёт, устно, традиционная форма | по графику промежуточной аттестации |  |  |
| Семестр 3 |  |  | зачёт, устно, традиционная форма | по графику промежуточной аттестации |  |  |
| Семестр 4 |  |  | зачёт, устно, традиционная форма, защита проекта | по графику промежуточной аттестации |  |  |

**2.2. Структура и содержание учебных занятий**

**Основной курс Основная траектория Очная форма обучения**

Период обучения (модуль): Семестр 1

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Наименование темы (раздела, части) | Вид учебных занятий | Количество часов |
| 1. | Введение | Практические занятия | 4 |
| Самостоятельная работа по методическим материалам | 4 |
| 2. | Введение в язык Си | Практические занятия | 12 |
| Самостоятельная работа по методическим материалам | 12 |
| 3. | Стиль программирования | Практические занятия | 4 |
| Самостоятельная работа по методическим материалам | 4 |
| 4. | Работа программы в современной ОС | Практические занятия | 4 |
| Самостоятельная работа по методическим материалам | 6 |
| 5. | Базовые структуры и алгоритмы компьютерной обработки данных | Практические занятия | 28 |
| Самостоятельная работа по методическим материалам | 36 |
| Самостоятельная работа под руководством преподавателя | 2 |
| 6. | Компиляция программ. Конечные автоматы | Практические занятия | 8 |
| Самостоятельная работа по методическим материалам | 10 |

Период обучения (модуль): Семестр 2

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Наименование темы (раздела, части) | Вид учебных занятий | Количество часов |
| 7. | Основные понятия ООП | Практические занятия | 14 |
| Самостоятельная работа по методическим материалам | 9 |
| 8. | Элементы UML | Практические занятия | 4 |
| Самостоятельная работа по методическим материалам | 4 |
| 9. | Введение в .NET Framework и C# | Практические занятия | 38 |
| Самостоятельная работа по методическим материалам | 20 |
| 10. | Модульное тестирование в ПО | Практические занятия | 4 |
| Самостоятельная работа по методическим материалам | 3 |

Период обучения (модуль): Семестр 3

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Наименование темы (раздела, части) | Вид учебных занятий | Количество часов |
| 11. | Теоретические основы параллельного программирования | Практические занятия | 4 |
| Самостоятельная работа по методическим материалам | 3 |
| 12. | Примеры параллельных алгоритмов | Практические занятия | 6 |
| Самостоятельная работа по методическим материалам | 5 |
| 13. | Организация синхронизации в многопоточных программах | Практические занятия | 16 |
| Самостоятельная работа по методическим материалам | 10 |
| 14. | Высокоуровневые структуры данных и алгоритмы общего назначения для многопоточного программирования | Практические занятия | 30 |
| Самостоятельная работа по методическим материалам | 17 |
| 15. | Введение в разработку параллельных программ на высокоуровневых платформах | Практические занятия | 4 |
| Самостоятельная работа по методическим материалам | 4 |

Период обучения (модуль): Семестр 4

|  |  |  |  |
| --- | --- | --- | --- |
| № п/п | Наименование темы (раздела, части) | Вид учебных занятий | Количество часов |
| 16. | Основные понятия архитектуры программных систем | Практические занятия | 4 |
| 17. | Введение в инженерию требований | Практические занятия | 8 |
| Самостоятельная работа по методическим материалам | 12 |
| 18. | Примеры архитектур программных систем | Практические занятия | 26 |
| Самостоятельная работа по методическим материалам | 36 |
| 19. | Инженерные практики, связанные с архитектурой программных систем | Практические занятия | 19 |
| Самостоятельная работа по методическим материалам | 20 |
| 20. | Паттерны ООП как элементы архитектуры программной системы | Практические занятия | 3 |
| Самостоятельная работа по методическим материалам | 4 |

Семестр 1:

1. Введение.

Основы владения компьютером, инструктаж по поведению в ЛВС и технике безопасности.

2. Введение в язык Си.

Понятия последовательного выполнения, оператора, блока, переменной. Архитектура ЭВМ: биты, байты, переменные, адреса. Представление целых чисел в памяти компьютера. Представление вещественных чисел по IEEE 754. Препроцессор, структура программы. Основные конструкции языка С (if, for, while). Примитивные типы (char, int, long, signed/unsigned, float, double, char\*). L-value и R-value. Указатели. Работа с динамической памятью. Виртуальная память. Соглашения о вызове. Стек и куча. Ввод-вывод stdio.h

3. Стиль программирования.

Форматирование программ: стиль, важность, обоснование. Именование переменных.

4. Работа программы в современной ОС.

Знакомство с инструментальной средой (выбор конкретной среды за преподавателем). Понятие программы, компилятора, исполняемого модуля. Разница между интерпретацией и трансляцией. Понятие процесса. Файловый ввод-вывод: буферизованный, файлы, отображаемые на память.

5. Базовые структуры и алгоритмы компьютерной обработки данных.

Сложность алгоритмов, основные нотации. Сортировка Шелла. Быстрая сортировка. Сети сортировки, битоническая сортировка. Radix-сортировка (LSD и MSD). Бинарные деревья поиска. АВЛ-дерево. BR-дерево. B-дерево. Декартово дерево. R-дерево. Динамическое программирование. Алгоритмы длинной арифметики. Основные понятия теории графов. Обходы. Примеры алгоритмов (напр., Дейкстры, построения минимального остовного дерева). Хэширование, алгоритмы реализации хэш-таблиц.

6. Компиляция программ. Конечные автоматы.

Данный раздел является опциональным. По желанию преподавателя вместо него можно расширить 5 раздел дополнительными алгоритмами и структурами обработки данных на выбор преподавателя.

Основные этапы компиляции программы. Алгебра регулярных выражений. Реализация интерпретатора базового набора операций (загрузка, сохранение, арифметика, условные и безусловные переходы) для стековой виртуальной машины. Решение некоторых элементарных вычислительных задач с помощью реализованного набора операций. Понятие конечного автомата. Автоматы Мили и Мура. Эквивалентность конечных автоматов Мили и Мура. Распознавание строк регулярными выражениями.

Семестр 2:

7. Основные понятия ООП

Понятие объекта. Понятие класса и экземпляра класса. Понятие поля. Понятие метода. Статические поля и методы. Наследование. Интерфейс и абстрактные классы. Параметрический и ad hoc полиморфизм. Принцип инкапсуляции. Принципы SOLID. Обобщённое программирование (generics). Понятие рефлексии. Делегаты. События.

8. Элементы UML

Диаграмма классов UML. Диаграмма последовательности UML.

9. Практика на объектно-ориентированном языке программирования

Содержание данного модуля во многом определяется выбранным преподавателем языком программирования. В качестве примеров таких языков, рекомендуемых к рассмотрению, выступают C#, Java, Kotlin, Python. Далее приводится пример содержания данного модуля, реализованного как введение в .NET Framework и язык C#, который рекомендуется брать за основу при реализации данного курса:

Структура .NET Framework. Статическое и динамическое связывание. Понятие сборки в .NET. Модификаторы видимости в C#. Понятие свойства. Аксессоры. Ключевое слово static. Ключевое слово this. Наследование в языке C#. Обобщённое программирование в C#. Конструкция foreach и интерфейс IEnumerable. Ключевое слово yield. Синтаксис лямбда-выражений в языке C#. Методы расширения (extension methods). Набор методов расширений в LINQ. Язык LINQ. Элементы функционального программирования: функции высшего порядка, замыкания. Понятие рефлексии и её возможности в C#. Reflection.Emit. Деревья выражений. Сильные и слабые ссылки. Пользовательский интерфейс в .NET Framework: технологии WinForms и WPF.

10. Введение в тестирование ПО

Модульное тестирование. Регрессионное тестирование. Нагрузочное тестирование. Интеграционное тестирование.

Семестр 3:

11. Теоретические основы параллельного программирования

Таксономия Флинна и типы параллелизма. Процессы, потоки, файберы. Диспетчеризация в операционных системах. Закон Амдала. Закон Густафсона-Барсиса. Понятие взаимного исключения и критической секции. Deadlock, livelock, голодание. Барьер памяти. Понятие блокирующего алгоритма, lock-free, wait-free. Проблема ABA. Атомарные операции.

12. Примеры параллельных алгоритмов

Операция merge-split. Распараллеливание пузырьковой сортировки. Распараллеливание quicksort. Параллельный алгоритм Флойда. Параллельный алгоритм Прима.

13. Организация синхронизации в многопоточных программах

Алгоритм Петерсона. Доказательство корректности. Теорема Херлихи о консенсусе. Понятие спинлока. Примеры алгоритмов. Семафоры и мьютексы. Решение задачи производителей-потребителей с ограниченным буфером на семафорах и мьютексах. Мониторы и переменные условия. Реэнтрантный спинлок. Read-Write Lock.

14. Высокоуровневые структуры данных и алгоритмы общего назначения для многопоточного программирования

Барьерная синхронизация. Понятие Future. Алгоритмы синхронизации стека и приоритетных очередей. Высокоуровневая, детализированная, оптимистическая, ленивая и неблокирующая синхронизации для односвязного сортированного списка. Алгоритмы синхронизации хэш-таблицы на списках. Неблокирующая синхронизация хэш-таблицы на списках. Алгоритмы синхронизации хэш-таблицы с открытой адресацией. Алгоритмы синхронихации skiplists. Организация пулов потоков: стратегии work-stealing, work-sharing. Синхронизированные коллекции в платформах современных языков программирования (напр. ConcurrentList, ConcurrentDictionary в .NET).

15. Введение в разработку параллельных программ на высокоуровневых платформах

На выбор преподавателя можно давать один из трёх равнозначных треков:

А) MPI: основные понятия и операции.

Б) Введение в OpenMP.

В) AKKA/AKKA.NET: основы модели акторов.

Семестр 4:

16. Основные понятия архитектуры программных систем

Понятие архитектуры программной системы. Подходы к архитектурным решениям (или их отсутствие) в различных методологиях разработки. Понятие паттерна в архитектуре программных систем. Примеры.

17. Введение в инженерию требований

Понятие требования. Список заинтересованных лиц. Подходы к сбору требований. Участники. Спецификация программной системы согласно IEEE 830. Оформление требований в виде прецедентов. Элементы UML. Диаграмма прецедентов. Пользовательские истории (User stories).

18. Примеры архитектур программных систем

Многослойная архитектура. Состав слоёв, распределение ответственностей. Сценарии транзакции. Модель предметной области. Сервисно-ориентированная архитектура. Гексагональная архитектура. Плагинная архитектура. Архитектура «фильтры и трубы». Основные решаемые задачи и пути их решения. Пиринговая архитектура. Основные решаемые задачи и пути их решения. Шина сервисов. Микросервисная архитектура.

19. Инженерные практики, связанные с архитектурой программных систем

Инфраструктурные способы организации сервера приложения. Виртуальные машины и контейнерная виртуализация. Границы применимости. IoC-контейнеры. Агрегирование событий. Определение и практики археологии программного обеспечения. Понятие рефакторинга. «Зеленое» программное обеспечение и энергоэффективное программирование.

20. Паттерны ООП как элементы архитектуры программной системы

**Раздел 3. Обеспечение учебных занятий**

**3.1. Методическое обеспечение**

**3.1.1 Методические указания по освоению дисциплины**

Для освоения дисциплины обучающиеся должны посещать практические занятия и выполнять задания преподавателей.

Как правило, курс начинается с опроса обучающихся о степени владения компьютером, умении программировать и известных языках программирования. Опрос, как правило, включает в себя несколько умеренно сложных задач по программированию для уровня выпускника средней школы и решается письменно. По итогам опроса можно определить уровень обучающихся и, в соответствии с результатами, оптимизировать структуру занятий. При необходимости разделения на подгруппы результаты опроса помогают в принятии решения.

В целом разделение материала по 4 семестрам происходит следующим образом:

Семестр 1: Основы программирования, язык Си, знакомство с архитектурой современных ЭВМ. Трансляция и интерпретация. Структуры и алгоритмы компьютерной обработки данных.

Семестр 2: Принципы объектно-ориентированного программирования (ООП), практика на объектно-ориентированном языке. Введение в тестирование ПО.

Семестр 3: Введение в алгоритмы параллельного программирования и сопутствующие проблемы синхронизации потоков/процессов и организации данных.

Семестр 4: Введение в архитектуру программных систем, архитектурные паттерны и сопутствующие вопросы/

Примечание: в семестре 2 у преподавателя есть право на выбор языка программирования для практики (см. описание модуля 9) и сопутствующую технологическую корректировку сопровождающих материал задач при наличии соответствующей технической оснащенности аудиторий.

На практических занятиях преподаватель сопровождает самостоятельную работу студентов материалом курса согласно следующему содержанию в разбивке по модулям и разделам согласно п.2.2.

Раздел 20 является опциональным и по согласованию с текущим лектором курса [003669] Проектирование и архитектура программного обеспечения может не читаться. В этом случае освободившиеся часы могут быть добавлены к модулям 18 и 19.

**3.1.2 Методическое обеспечение самостоятельной работы**

При самостоятельном выполнении практических заданий и во время подготовки презентации целесообразно использовать рекомендованную основную и дополнительную литературу. По согласованию с преподавателем обучающийся может получать теоретические знания, требуемые для освоения курса, по одному из онлайн-курсов, представленных на образовательных платформах в п. 3.4. При этом преподаватель обязан сообщить обучающимся, какие разделы выбранного им онлайн-курса недостаточно полно раскрывают ту или иную тему курса и порекомендовать дополнительные источники по данной теме. Преподаватель вправе отказать обучающемуся в самостоятельном освоении теоретической части дисциплины по онлайн-курсу в случае нахождения в нем существенных расхождений с содержанием курса в разделе 2 и п. 3.1.1. Прошедший онлайн-курсы обучающийся получает допуск и проходит промежуточную аттестацию на общих основаниях.

Практическая часть дисциплины может быть полностью выполнена самостоятельно по методическим материалам.

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

Текущий контроль состоит из основных мероприятий и дополнительных мероприятий. Основные мероприятия обязательны к реализации. Дополнительные могут быть введены по желанию преподавателя. Успехи обучающегося в каждом мероприятии оцениваются баллами. Преподаватель обязан на первом занятии семестра ознакомить обучающихся со структурой и стоимостью в баллах всех контрольных мероприятий, проводимых в течение грядущего семестра.

К основным мероприятиям относятся:

* В 1-3 семестре – задания. Примеры заданий в разбивке по модулям можно взять в пособии С.Ю. Сартасова «Практикум на ЭВМ» (<http://hdl.handle.net/11701/15461>). Успешная сдача каждой задачи оценивается в 10 баллов. По желанию преподавателя можно добавлять дополнительные условия:
  + Некоторые задачи могут быть повышенной сложности и оцениваться в 20 баллов.
  + Для задач допустимо вводить промежуточные и предельные сроки сдачи. Сдача после промежуточного срока оценивается не более чем в половину максимальной оценки за задачу, после предельного срока – строго в 0 баллов.
  + В начале семестра преподаватель сообщает обучающимся правила хорошего оформления программ, соответствующие ведущим практикам индустрии разработки программного обеспечения. Обучающийся теряет 3 балла за каждую попытку сдачи с нарушениями этих правил (вплоть до отрицательной оценки за задачу).
  + Баллы за задачу могут быть снижены вплоть до 50% (с округлением вниз), если тестовое покрытие решения недостаточно.
  + Баллы за задачу могут быть снижены до 50%(с округлением вниз), если решение существенно неоптимально.

Во 2 и 3 семестре рекомендуется вести разработку на объектно-ориентированном языке.

* В семестре 4 обучающимся в качестве самостоятельной работы предлагается реализовать программный проект, начиная с требований и заканчивая его релизом (без сопроводительной релизной документации), на тему, которую они выбирают совместно с преподавателем. Допускается командная работа при условии, что каждый участник будет при презентации проекта рассказывать, что он конкретно сделал в рамках проекта. Обучающимся рекомендуется с самого начала пользоваться системой контроля версий, и история коммитов помогает подтвердить их слова о распределении ответственности в процессе разработки. Тема проекта не должна быть слишком легкой, а глубина проработки деталей не должна быть чрезмерной. Примеры тем проектов:

1. Игра в жанре Tower Defense.
2. Мобильное приложение для отслеживания доходов и расходов.
3. Игра платформер с адаптивным искусственным интеллектом.
4. Программа для управления квадрокоптером.
5. Мобильный клиент для социальной сети VKontakte.
6. Программа для чтения файлов в формате FB2.
7. Бот для Telegram, интегрирующийся с eBay.

Разработка проекта включает в себя следующие этапы:

1. Формулировка темы проекта – что обучающийся хочет сделать в свободной форме. После этого этапа тему менять нельзя.
2. Требования к проекту, оформленные в виде IEEE Software Requirements Specification, UML-диаграммы прецедентов и их описания или набора пользовательских историй (User stories).
3. Диаграмма классов – какой видится структура классов проектов в начале разработки в нотации UML.
4. Демонстрация кода проекта
5. Демонстрация готового проекта.
6. Фактическая диаграмма классов – структура классов проекта в том виде, в каком она получилась в результате разработки.
7. Список известных ошибок и недоработок проекта.

Для стимулирования работы обучающихся в течение семестра для каждого из этапов назначаются две даты – полудедлайна, к которому обучающийся должен продемонстрировать хотя бы минимальное начало работ по данному этапу, и дедлайна, к которому этап должен быть завершен. Изначально деятельность обучающегося в семестре оценивается в 100 баллов. Нарушение любой из этих дат без предупреждения и/или уважительной причины влечет за собой снятие 10 баллов за каждую дату до минимального значения в 0 баллов. Итоговая оценка за семестровый проект – это баллы, оставшиеся у него после последнего дедлайна.

Пример календаря на 2023 год приводится ниже.

|  |  |  |
| --- | --- | --- |
| Этапы | Полудедлайн | Дедлайн |
| Тема проекта | 26.02.2023 | 05.03.2023 |
| Требования | 12.03.2023 | 19.03.2023 |
| Диаграмма классов проектная | 26.03.2023 | 02.04.2023 |
| Первая демонстрация кода проекта | 09.04.2023 | 16.04.2023 |
| Вторая демонстрация кода проекта | 30.04.2023 | 07.05.2023 |
| Демонстрация готового проекта | - | 21.05.2023 |
| Диаграмма классов фактическая | - | 21.05.2023 |
| Известные ошибки и недоработки | - | 21.05.2023 |

К дополнительным мероприятиям относятся:

* В рамках раздела «20. Паттерны ООП как элементы архитектуры программной системы» обучающиеся делают доклады о классических паттернах ООП, описанных, например, в книге «Приемы объектно-ориентированного проектирования. Паттерны проектирования» (см. п. 3.4.1). Доклад включает в себя:
* Название паттерна;
* Соответствующая(-ие) ему схему(-ы) в нотации UML;
* Рассказ об устройстве паттерна;
* Примеры использования паттерна, не повторяющие таковые в книге «Приемы объектно-ориентированного проектирования. Паттерны проектирования»;
* Границы применимости и отличия от схожих паттернов.

Рекомендуемая продолжительность доклада – 15-20 минут. Для демонстрации обучающимся разрешается использовать для иллюстраций как доску, так и презентации на мультимедийном проекторе. Успешный доклад оценивается в 20 баллов.

* Преподаватель вправе давать обучающиеся контрольные работы для оценки степени освоенности материала. Задания подбираются по усмотрению преподавателя. Использование обучающимися дополнительных источников информации остаётся также на усмотрение преподавателя. Контрольные подразделяются на 2 типа:
  + Большие работы на 2 академических часа. Могут состоять из нескольких задач, каждая из которых оценивается в 15 баллов. Критерии оценки должны быть доведены до сведения обучающихся перед началом контрольной.
  + Малые работы на 10-15 минут. Содержат одну несложную задачу. Оцениваются максимум в 5 баллов.
* Преподаватель может давать обучающимся подготовить доклад по теме, связанной с программной инженерией, как входящей, так и не входящей в описанную в разделе 2.2 и 3.1.1 структуру курса. Доклад состоит из презентации их устного выступления. Максимальная оценка за каждый доклад назначается преподавателем в индивидуальном порядке, но не может превышать 20 баллов. Критерии оценки должны быть доведены до сведения обучающихся перед началом подготовки доклада.

Промежуточная аттестация может проходить на усмотрение преподавателя в двух формах – в форме с устным зачётом и без устного зачёта.

В случае без устного зачёта баллы, накопленные обучающимся в течение семестра за мероприятия текущего контроля пересчитываются в итоговый процент освоения курса.

В случае устного зачёта обучающийся допускается к нему, если за все мероприятия текущего контроля в семестре получена строго положительная оценка.

Билет состоит из одного вопроса. Преподаватель вправе выбрать одну из форм проведения зачёта:

* классическая форма, при которой время подготовки ответа на вопрос билета составляет не менее 0,5 академического часа, и на время ответа ограничения не накладывается;
* блиц-опрос, когда обучающийся начинает сразу же после получения билета, а время ответа ограничивается 10-15 минутами.

В обоих случаях использование конспектов и учебников, а также электронных устройств хранения, обработки или передачи информации при подготовке и ответе на вопросы зачета запрещено.

После ответа на вопрос билета, преподаватель вправе задать уточняющие вопросы по услышанному. Затем преподаватель задает дополнительные вопросы по любой теме из списка вопросов, вынесенных на зачет. В качестве уточняющих используются вопросы, не требующие длительного ответа, в том числе основные определения и понятия. Дополнительные вопросы должны строиться таким образом, чтобы с их помощью обучающийся мог продемонстрировать своё понимание или непонимание материала, а также способности к аргументированному обсуждению. Рекомендуется задавать 3-4 дополнительных вопроса на билет и не более 2-3 уточняющих вопросов. Преподаватель вправе увеличить количество вопросов в случае, если у него не возникает понимание, освоил обучающийся материал учебного курса или нет.

В любой момент обучающийся имеет право отказаться от ответа с выставлением оценки в 0 баллов.

Максимальная оценка K за устный зачёт равна сумме максимальных баллов за каждое контрольное мероприятие в течение семестра плюс один балл. Максимальный балл за основной и уточняющие вопросы не может составлять более 50% K. Оставшийся бюджет из 50% баллов делится поровну по дополнительным вопросам. Преподаватель вправе снижать баллы за неточности и ошибки в зависимости от их грубости. Если суммарная оценка за устный зачёт превышает порог в 70% К, считается, что обучающийся сдал устный зачёт на полученное количество баллов, которые добавляются к баллам, накопленным в семестре. В противном случае считается, что обучающийся сдал зачёт на 0 баллов.

По желанию преподавателя на экзамен допустимо приглашать других преподавателей с квалификацией не ниже изложенной в п. 3.2.1 как для независимого оценивания ответов обучающихся, так и для коллегиального. В последнем случае оценка за зачет ставится на основании голосования простого большинства. В спорных ситуациях преподаватель, ведущий дисциплину, имеет право принятия окончательного решения.

Перевод из баллов, полученных за мероприятия текущего и промежуточного контроля, в итоговый процент освоения курса делается по формуле

где I – итоговый процент освоения курса, N – максимальное количество баллов, зарабатываемое в течение семестра, n – заработанное обучающимся число баллов. Перевод в оценки делается по стандартной методике согласно приказу №7293/1 от 20.07.2018:

|  |  |  |
| --- | --- | --- |
| Итоговый процент освоения курса, % | Оценка СПбГУ при проведении зачёта | Оценка ECTS |
| 0-49 | Не зачтено | F |
| 50-60 | Зачтено | E |
| 61-69 | Зачтено | D |
| 70-79 | Зачтено | C |
| 80-89 | Зачтено | B |
| 90-100 | Зачтено | A |

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

Примерные списки вопросов к модулям курсов:

Семестр 1:

* + 1. Гарвардская, фон Неймановская и гибридная архитектуры ЭВМ.

1. Разрядность процессора и операционной системы. Машинное слово. Порядки следования байтов.
2. Прямое, обратное и дополнительное представление целых чисел в памяти компьютера.
3. Представление вещественных чисел по IEEE 754.
4. Физическая и виртуальная память. Сегментно-страничная организация памяти ЭВМ.
5. Понятие кэша центрального процессора. Влияние кэша на быстродействие программ.
6. Понятие указателя в языке С. Понятие косвенной адресации. Операции с указателями.
7. Стек и динамическая память. Операции с динамической памятью.
8. Сложность алгоритмов, основные нотации. Сортировка Шелла.
9. Быстрая сортировка.
10. Сети сортировки, битоническая сортировка. Radix-сортировка (LSD и MSD).
11. Бинарные деревья поиска. АВЛ-дерево.
12. BR-дерево.
13. B-дерево.
14. R-дерево.
15. Пирамида (куча), основные операции.
16. Алгоритм heapsort. Приоритетные очереди.
17. Декартово дерево.
18. Система непересекающихся множеств.
19. Динамическое программирование. Основные этапы и примеры алгоритмов.
20. Понятие о жадных алгоритмах.
21. Основные понятия теории графов. Обходы.
22. Алгоритм Дейкстры для нахождения минимальных путей.
23. Алгоритм Краскала для построения минимального остовного дерева.
24. Раскраска графов. Понятие хроматического многочлена.
25. Основные понятия теории автоматов. Принцип работы автоматов Мили и Мура.
26. Длинная арифметика. Сложение. Умножение в столбик.
27. Алгоритм умножения Карацубы.
28. Алгоритм умножения Тоома-Кука.
29. Алгоритм деления в длинной арифметике.

Семестр 2:

1. Понятие класса и экземпляра класса.
2. Понятие поля. Понятие метода.
3. Модификаторы видимости.
4. Жизненный цикл объекта.
5. Понятие свойства в C#. Аксессоры.
6. Ключевое слово static. Ключевое слово this.
7. Понятие наследования. Наследование в языке C#.
8. Понятие интерфейса. Понятие абстрактного класса.
9. Диаграмма классов UML.
10. Понятие инкапсуляции.
11. Принципы SOLID: S, O.
12. Принципы SOLID: L.
13. Принципы SOLID: I, D.
14. Параметрический и ad hoc полиморфизм.
15. Обобщённое программирование на примере дженериков.
16. Понятие делегата. Понятие события.
17. Понятие рефлексии.
18. IoC-контейнеры.

Семестр 3:

* 1. Таксономия Флинна и типы параллелизма. Процессы, потоки, файберы. Диспетчеризация в операционных системах.
  2. Закон Амдала. Закон Густафсона-Барсиса.
  3. Операция merge-split. Распараллеливание пузырьковой сортировки.
  4. Распараллеливание quicksort.
  5. Параллельный алгоритм Флойда.
  6. Параллельный алгоритм Прима.
  7. MPI: основные понятия и операции.
  8. Понятие взаимного исключения и критической секции. Deadlock, livelock, голодание. Инверсия приоритетов. Барьер памяти. Задача об обедающих философах.
  9. Алгоритм Петерсона. Доказательство корректности.
  10. Алгоритм Лэмпорта. Понятие консенсуса, теорема о консенсусе (без доказательства).
  11. Понятие блокирующего алгоритма, lock-free, wait-free. Проблема ABA.
  12. Атомарные операции.
  13. Понятие спинлока. Примеры реализации спинлоков по принципу Test-And-Set.
  14. Неявная и явная реализация спинлоков с помощью списков.
  15. Семафоры и мьютексы. Решение задачи производителей-потребителей с ограниченным буфером на семафорах и мьютексах.
  16. Мониторы и переменные условия.
  17. Реэнтрантный спинлок. Read-Write Lock.
  18. Понятие барьерной синхронизации. Простая барьерная синхронизация.
  19. Иерархическая барьерная синхронизация.
  20. Высокоуровневая и детализированная синхронизация для односвязного сортированного списка.
  21. Оптимистическая синхронизация для односвязного сортированного списка
  22. Ленивая синхронизация для односвязного сортированного списка.
  23. Неблокирующая синхронизация для односвязного сортированного списка.
  24. Алгоритмы синхронизации хэш-таблицы на списках.
  25. Неблокирующая синхронизация хэш-таблицы на списках.
  26. Алгоритмы синхронизации хэш-таблицы с открытой адресацией.
  27. Понятие об асинхронности. Понятие Future. Ключевые слова async и await в языке C#.
  28. Понятие о пуле потоков. Дисциплины обработки задач Work Stealing и Work Sharing.
  29. Понятие списка с пропусками. Ленивая синхронизация для списка с пропусками.
  30. Суперскалярные команды (AVX, SSE) как средства обеспечения параллелизма на одном ядре
  31. Базовые понятия GPGPU и технологии NVIDIA CUDA.

Семестр 4:

* 1. Понятие архитектуры программной системы. Подходы к архитектурным решениям (или их отсутствие) в различных методологиях разработки.
  2. Понятие паттерна в архитектуре программных систем. Примеры.
  3. Понятие требования. Список заинтересованных лиц. Подходы к сбору требований. Участники.
  4. Спецификация программной системы согласно IEEE 830.
  5. Оформление требований в виде прецедентов.
  6. Элементы UML. Диаграмма прецедентов.
  7. Пользовательские истории (User stories).
  8. Многослойная архитектура. Состав слоёв, распределение ответственностей.
  9. Сценарии транзакции. Описание и область применимости.
  10. Модель предметной области. Описание и область применимости.
  11. Сервисно-ориентированная архитектура. Описание и область применимости.
  12. Гексагональная архитектура. Основные решаемые задачи и пути их решения.
  13. Плагинная архитектура. Основные решаемые задачи и пути их решения.
  14. Архитектура «фильтры и трубы». Основные решаемые задачи и пути их решения.
  15. Пиринговая архитектура. Основные решаемые задачи и пути их решения.
  16. Шина сервисов. Основные решаемые задачи и пути их решения.
  17. Инфраструктурные способы организации сервера приложения.
  18. Виртуальные машины и контейнерная виртуализация. Границы применимости.
  19. «Зеленое» программное обеспечение и понятие энергетической эффективности программных систем.
  20. Определение и практики археологии программного обеспечения.
  21. Понятие рефакторинга.

|  |  |  |
| --- | --- | --- |
| № | Код индикатора и индикатор достижения универсальной компетенции | Контрольно-измерительные материалы (КИМ) (тестовые вопросы, контрольные задания, кейсы и пр.) |
|  | 1 | 2 |
| 1 | ОПК-1.1 Уметь идентифицировать возможные проблемы и пути их решения | Ответы на вопросы теоретических зачётов всех четырёх семестров независимо оцениваются по шкале от 0 (не сделано) до 100 (очень хорошо) и усредняются. |
| 2 | ОПК-2.1 Уметь писать программный код с использованием языков программирования, определения и манипулирования данными | Проект 4 семестра и его ведение обучающимся в семестре оценивается по шкале от 0 (не сделано) до 100 (очень хорошо) |
| 3 | ОПК-6.1 Уметь проверять и отлаживать программный код | Проект 4 семестра и его ведение обучающимся в семестре оценивается по шкале от 0 (не сделано) до 100 (очень хорошо) |
| 4 | ОПК-7.1 Уметь определять качественные характеристики каждого компонента | Проект 4 семестра и его ведение обучающимся в семестре оценивается по шкале от 0 (не сделано) до 100 (очень хорошо) |
| 5 | ПКА-1.1 Быть способным осуществлять организационное и технологическое обеспечение кодирования на языках программирования | Список баговпроекта 4 семестра оценивается по шкале от 0 (не сделано) до 100 (очень хорошо). |
| 6 | ПКП-1.1 Уметь разрабатывать технических спецификаций на программные компоненты и их взаимодействие | Требования к проекту 4 семестра оцениваются по шкале от 0 (не сделано) до 100 (очень хорошо). |
| 7 | ПКП-2.1 Уметь описывать алгоритмы компонентов, включая методы и схемы | Интерфейс проекта 4 семестра оценивается по шкале от 0 (не сделано) до 100 (очень хорошо). |
| 8 | ПКП-5.1 Уметь выбирать технологии и средства разработки программного обеспечения | Проект 4 семестра и его ведение обучающимся в семестре оценивается по шкале от 0 (не сделано) до 100 (очень хорошо) |
| 9 | ПКП-6.1 Уметь разрабатывать процедуры проверки работоспособности и измерения характеристик программного обеспечения | Ответы на вопросы теоретических зачётов всех четырёх семестров независимо оцениваются по шкале от 0 (не сделано) до 100 (очень хорошо) и усредняются. |
| 10 | ПКП-7.1 Умеет проводить технические и управленческие ревизии создаваемого программного средства | Проект 4 семестра и его ведение обучающимся в семестре оценивается по шкале от 0 (не сделано) до 100 (очень хорошо) |
| 11 | УК 1.1. Анализирует задачу, выделяя ее базовые составляющие | Требования к проекту 4 семестра оцениваются по шкале от 0 (не сделано) до 100 (очень хорошо). |
| 12 | УК-3.4. Осуществляет обмен информацией, знаниями и опытом с членами команды; | Ответы на вопросы теоретических зачётов всех четырёх семестров независимо оцениваются по шкале от 0 (не сделано) до 100 (очень хорошо) и усредняются. |

**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 преподаванию дисциплины могут быть допущены преподаватели, имеющие диплом о высшем техническом образовании по направлениям «Программная инженерия», «Математическое обеспечение и администрирование информационных систем» и другим смежным направлениям, связанным с информационными технологиями.

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

Для подготовки и проведения занятий требуется лаборант или инженер для следующих работ:

* + 1. Техническая подготовка каталогов исходных данных в форме, удобной для учебной работы.
    2. Настройка пакетов прикладных программ, перечисленных в п. 3.3.2.
    3. Поддержание работоспособности компьютерного класса.

**3.3. Материально-техническое обеспечение**

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

Учебные аудитории для проведения учебных занятий, оснащенные стандартным оборудованием, используемым для обучения в СПбГУ в соответствии с требованиями материально-технического обеспечения.

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

Стандартное оборудование, используемое для обучения в СПбГУ. MS Windows, MS Office, Mozilla FireFox, Google Chrome, Acrobat Reader DC, WinZip, Антивирус Касперского.

Перечень программного обеспечения общего пользования:

Модуль С1: OS Linux с установленным пакетом gcc, или OS Windows (7 и выше) с установленной IDE MS Visual Studio (2010 и выше) с пакетом для разработки на C++ или IDE NetBeans с пакетом cygwin или IDE CLion

Период С2-С4: При выборе языка C# - .NET Framework (4.6.2 и выше), IDE MS Visual Studio (2015 и выше) с пакетом разработки на C#. При выборе языка Java или Kotlin – Open JDK 9.0.4 с IDE IntelliJ Idea версии не меньше 2016.1.3.

При выборе преподавателем иного языка программирования установка необходимого программного обеспечения согласовывается отдельно с техническим персоналом компьютерных классов и системными администраторами университета.

**3.3.3 Характеристики специализированного оборудования**

Специализированное оборудование не требуется.

**3.3.4 Характеристики специализированного программного обеспечения**

Для обеспечения занятий в модулях С1-С4 требуется настроенный сервер системы контроля версий (Subversion, Git, Mercurial). Альтернативно возможно использование внешних открытых сервисов (например, github). Желательно наличие сервера системы для учёта ошибок (Jira, Redmine).

**3.3.5 Перечень и объёмы требуемых расходных материалов**

Для аудиторий с маркерными досками необходимы стирающиеся маркеры в объеме, достаточном для проведения курса. Для аудиторий с меловыми досками необходим мел в объеме, достаточном для проведения курса. В обоих случаях необходимы соответствующие средства для удаления записей с поверхности доски.

**3.4. Информационное обеспечение**

* + 1. **Список литературы**

1. Приемы объектно-ориентированного проектирования. Паттерны проектирования / Гамма, Хелм, Джонсон, Влиссидес, - СПб.: Питер, 2016. - 366 с.
2. Shavit N., Herlihy M. The Art of Multiprocessor Programming. – 2-е изд. - Burlington, MA: Morgan Kaufmann Publishers, 2012. - 552 с.

- **ЭР по подписке СПбГУ:**

<https://proxy.library.spbu.ru:2065/library/view/the-art-of/9780123973375/?ar&orpq&email=Q%2fMxoNcLvqg%2bkX8NcVSO4A%3d%3d&tstamp=1588175752&id=D2B60E014EB3C0FCCDE53910AFF0FB83D51370F4>

1. Сартасов С.Ю. Практикум на ЭВМ. [Электронный ресурс] <http://hdl.handle.net/11701/15461>

**3.4.2 Перечень иных информационных источников**

* Сайт Научной библиотеки им. М. Горького СПбГУ: <http://www.library.spbu.ru/>
* Электронный каталог Научной библиотеки им. М. Горького СПбГУ: <http://www.library.spbu.ru/cgi-bin/irbis64r/cgiirbis_64.exe?C21COM=F&I21DBN=IBIS&P21DBN=IBIS>
* Перечень электронных ресурсов, находящихся в доступе СПбГУ: <http://cufts.library.spbu.ru/CRDB/SPBGU/>
* Перечень ЭБС, на платформах которых представлены российские учебники, находящиеся в доступе СПбГУ: <http://cufts.library.spbu.ru/CRDB/SPBGU/browse?name=rures&resource%20type=8>

**Приложение 1**

**Фонд оценочных средств по дисциплине «Практикум на ЭВМ»**

Входной контроль обучающихся

**Показатели и критерии оценивания, шкала оценивания**

Входной контроль для оценки знаний осуществляется на первом занятии по данной дисциплине. Продолжительность письменного контроля – два академических часа.

В ходе входного контроля обучающимся предлагается письменно ответить на следующие вопросы и решить следующие задачи:

1. Какое наименьшее количество операции умножения достаточно для вычисления значения формулы x4 + x3 + x2 + x + 1?
2. Укажите условия, при которых формулы “a + a – a” и “a + (a – a)” не эквивалентны.
3. Поменять значения двух целочисленных переменных местами (без привлечения третьей переменной и файлов).
4. Написать алгоритм нахождения неполного частного от деления a на b (целые числа), используя только операции сложения, вычитания и умножения.
5. 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 местами начало и конец.
6. Подсчитать число «счастливых билетов» (билет считается «счастливым», если сумма первых трёх цифр его номера равна сумме трёх последних).
7. Написать алгоритм проверки баланса скобок в исходной строке (т.е. число открывающих скобок равно числу закрывающих и выполняется правило вложенности скобок).
8. Заданы две строки: S и S1. Найдите количество вхождений S1 в S как подстроки.
9. Напишите программу, печатающую все простые числа, не превосходящие заданного числа.
10. Напишите программу, считающую количество нулевых элементов в массиве.

Задачи, требующие программной реализации (с 3 по 10) допускается решать на любом языке программирования по выбору обучающегося, либо излагать решение в виде псевдокода или словесного описания. Задачи 1 и 2 следует сопровождать решением, в результате которого был получен тот или иной ответ. Использование литературы и общение не допускается.

**Критерии оценки**

Критерии оценивания сообщаются обучающимся в начале тестирования. Каждая задача оценивается по шкале от 0 до 10 баллов, где 0 баллов ставится за отсутствующее или в корне неверное решение, 10 баллов ставится за правильное и алгоритмически оптимальное решение, демонстрирующее владение выбранным языком программирования, структурами данных и алгоритмами (для тех заданий, где это применимо). За словесное описание решения задач 3-10 с итоговой оценки снимается 5 баллов независимо от корректности решения до минимума в 0 баллов, а за решение на псевдокоде – снимается 3 балла. Итоговая оценка выставляется как сумма оценок за каждую задачу и находится в диапазоне от 0 до 100 баллов.

Результаты входного контроля используются при распределении обучающихся по подгруппам и оптимизации программы и материалов курса «Практикум на ЭВМ» под уровень подготовленности обучающихся. Результаты оценивания обучающимся не объявляются.

**Методические материалы, определяющие процедуры оценивания результатов обучения по дисциплине (модулю), практике.**

Методические материалы, определяющие процедуры оценивания, содержатся в п.4 «Промежуточная аттестация» «Правил обучения в Санкт-Петербургском университете».

**Раздел 4. Разработчики программы**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ФИО | Ученая  степень | Ученое звание | Должность | Контакты |
| Сартасов  Станислав Юрьевич |  |  | ст.преподаватель кафедры системного программирования | [Stanislav.Sartasov@spbu.ru](mailto:Stanislav.Sartasov@spbu.ru) |
| Журавлёв Максим Михайлович |  |  | ст.преподаватель кафедры системного программирования | [Maxim.Zhuravlev@spbu.ru](mailto:Maxim.Zhuravlev@spbu.ru) |
| Кириленко Яков Александрович |  |  | ст.преподаватель кафедры системного программирования | [I.Kirilenko@spbu.ru](mailto:I.Kirilenko@spbu.ru) |
| Куликов Егор Константиович |  |  | ст.преподаватель кафедры системного программирования | [e.kulikov@spbu.ru](mailto:e.kulikov@spbu.ru) |