Приложение к ОПОП
«Организация и программирование интеллектуальных систем»

СПбГЭТУ «ЛЭТИ»
ПЕРВЫЙ ЭЛЕКТРОТЕХНИЧЕСКИЙ

МИНОБРНАУКИ РОССИИ

федеральное государственное автономное образовательное учреждение высшего образования
**«Санкт-Петербургский государственный электротехнический университет
«ЛЭТИ» им. В.И.Ульянова (Ленина)»
(СПбГЭТУ «ЛЭТИ»)**

РАБОЧАЯ ПРОГРАММА

дисциплины

«АЛГОРИТМЫ И СТРУКТУРЫ ДАННЫХ»

для подготовки бакалавров

по направлению

09.03.01 «Информатика и вычислительная техника»

по профилю

«Организация и программирование интеллектуальных систем»

Санкт-Петербург

2024

ЛИСТ СОГЛАСОВАНИЯ

Разработчики:

ст. преподаватель Колинько П.Г.

Рабочая программа рассмотрена и одобрена на заседании кафедры ВТ 19.01.2024, протокол № 1

Рабочая программа рассмотрена и одобрена учебно-методической комиссией ФКТИ, 24.01.2024, протокол № 1

Согласовано в ИС ИОТ

Начальник ОМОЛА Загороднюк О.В.

1 СТРУКТУРА ДИСЦИПЛИНЫ

Обеспечивающий факультет	ФКТИ
Обеспечивающая кафедра	ВТ
Общая трудоемкость (ЗЕТ)	9
Курс	2
Семестр	4, 3
Виды занятий	
Лекции (академ. часов)	68
Лабораторные занятия (академ. часов)	34
Практические занятия (академ. часов)	34
Иная контактная работа (академ. часов)	6
Все контактные часы (академ. часов)	142
Самостоятельная работа, включая часы на контроль (академ. часов)	182
Всего (академ. часов)	324
Вид промежуточной аттестации	
Экзамен (курс)	2
Курсовая работа (курс)	2
Дифф. зачет (курс)	2
Курсовая работа (курс)	2

2 АННОТАЦИЯ ДИСЦИПЛИНЫ

«АЛГОРИТМЫ И СТРУКТУРЫ ДАННЫХ»

Изучаются способы реализации в ЭВМ абстрактных данных и вытекающие из этих способов свойства алгоритмов обработки этих данных. Обсуждаются способы генерации множеств для автоматизации тестирования программ и оборудования. Рассматриваются популярные алгоритмы на ненагруженных и нагруженных графах, жадные алгоритмы, эмпирические алгоритмы для переборных задач. Особое внимание при этом уделяется оптимальной организации данных для этих алгоритмов. Изучаются способы организации данных в реальных задачах, когда одному и тому же набору данных могут применяться одновременно несколько абстрактных моделей. Вводится понятие класса как способа реализации структуры данных в конкретной системе программирования. Даётся способ оценки временной сложности алгоритма в машинном эксперименте.

SUBJECT SUMMARY

«ALGORITHMS AND DATA STRUCTURES»

We study how to implement abstract data into a computer and the resulting properties of these methods of processing these data algorithms. Soba discussed spogenerating sets for the test automation software and equipment. We consider the popular algorithms unloaded and heat-conjugated graphs, greedy, empirical algorithms for search problems. Particular attention is paid to the optimal organization of data for these algorithms. Study ways of organizing data in real per-cottages, where the same data set can be used simultaneously, but more abstract models. The concept of class as a means of impletion of the data structure in a particular programming system. We give a method for evaluating the time complexity of the algorithm in the computer experiment.

3 ОБЩИЕ ПОЛОЖЕНИЯ

3.1 Цели и задачи дисциплины

1. Цели изучения дисциплины:

-изучение абстрактных типов данных и их реализации в объектно-ориентированной программе;

-получение знаний о связи между способами организации данных и вытекающими из них свойствами алгоритма;

-развитие навыков программирования, освоение языка C++ с учётом новейших стандартов, получение опыта применения ЭВМ для научных исследований.

2. Задачи изучения дисциплины:

-исследование возможностей популярных структур данных и влияния выбора структуры данных на свойства реализуемого алгоритма;

-изучение популярных алгоритмов и возможных областей их применения;

-приобретение опыта программирования в новейшей версии C++ и постановки экспериментов с полученным программным кодом.

3. Знания:

-особенностей новейших стандартов языка C++, Стандартной библиотеки шаблонов и механизма обработки исключительных ситуаций;

-популярных алгоритмов и возможных областей их применения;

-подходов к решению задач, для которых неизвестны эффективные алгоритмы;

-особенностей кодирования пользовательских структур данных и пользовательских контейнеров.

4. Умения:

-быстро создавать надёжные и эффективные программы в современных системах программирования;

-выбрать оптимальные структуры данных при реализации алгоритма и обосно-

вать свой выбор;

-эффективно реализовать и применить пользовательскую структуру данных и пользовательский контейнер, совместимый с алгоритмами Стандартной библиотеки шаблонов.

5. Навыки проектирования и отладки программ в парадигме объектно-ориентированного программирования в стандарте C++11/14/17/20/23, а также измерения свойств алгоритма в эксперименте на ЭВМ, применения ЭВМ в качестве инструмента для научных исследований.

3.2 Место дисциплины в структуре ОПОП

Дисциплина изучается на основе ранее освоенных дисциплин учебного плана:

1. «Информатика»

2. «Программирование»

и обеспечивает изучение последующих дисциплин:

1. «Web-программирование»

2. «Базы данных»

3. «Математическая логика и теория алгоритмов»

4. «Введение в искусственный интеллект»

5. «Производственная практика (технологическая (проектно-технологическая) практика)»

6. «Основы компьютерного зрения»

7. «Параллельные алгоритмы и системы»

3.3 Перечень планируемых результатов обучения по дисциплине, соотнесенных с планируемыми результатами освоения образовательной программы

В результате освоения образовательной программы обучающийся должен достичь следующие результаты обучения по дисциплине:

Код компетенции/ индикатора компетенции	**Наименование компетенции/индикатора компетенции**
ПК-1	Способен осуществлять проведение научно-исследовательских и опытно-конструкторских разработок по отдельным разделам темы
ПК-1.2	*Осуществляет выполнение экспериментов и оформление результатов исследований и разработок*
ПК-2	Способен осуществлять концептуальное, функциональное и логическое проектирование систем среднего масштаба и сложности
ПК-2.1	*Анализирует проблемную ситуацию, планирует разработку системы, осуществляет постановку целей*
ПК-2.3	*Организует оценку соответствия требованиям существующих систем и их аналогов*
ПК-3	Способен разрабатывать требования и проектировать программное обеспечение
ПК-3.1	*Анализирует требования к программному обеспечению*
ПК-3.3	*Проектирует структуры данных*

4 СОДЕРЖАНИЕ ДИСЦИПЛИНЫ

4.1 Содержание разделов дисциплины

4.1.1 Наименование тем и часы на все виды нагрузки

№ п/п	Наименование темы дисциплины	Лек, ач	Пр, ач	Лаб, ач	ИКР, ач	СР, ач
1	Введение	2				
2	Данные и алгоритмы	4	2	2		2
3	Генерация тестов	2	2	2		10
4	Множество как объект	4	2	2	2	10
5	Введение в Стандартную библиотеку шаблонов (STL)	4	2			10
6	Графы. Основные понятия, машинное представление и обходы	2		2		20
7	Алгоритмы на графах общего вида	4	2	2	2	10
8	Графы с нагруженными вершинами: сортировка	2	2	2		4
9	Графы с нагруженными рёбрами: поиск кратчайших путей	2	2	2		4
10	Графы с нагруженными вершинами и рёбрами: потоки в сетях и смежные задачи	4	2	2		12
11	Жадные алгоритмы решения информационных задач	2		2		8
12	Практические способы решения NP-полных задач	2	2			8
13	Иерархия классов	4	2	2		24
14	Исключительные ситуации	2	2	2		4
15	Способы хранения в памяти изменяющихся множеств	4	2	2		6
16	Деревья двоичного поиска с автобалансировкой	6	2	2		20
17	Поддержка последовательностей в структуре данных для множеств	4	2	2		12
18	Пользовательские дополнения к Стандартной библиотеке шаблонов (STL)	4	2	2		8
19	Ассоциативные контейнеры	2		2		6
20	Умные указатели	2	2			2
21	Измерение временной сложности алгоритма в эксперименте на ЭВМ	4	2	2	2	2
22	Заключение	2				
	Итого, ач	68	34	34	6	182
	Из них ач на контроль	0	0	0	0	35
	Общая трудоемкость освоения, ач/зе	324/9				

4.1.2 Содержание

№ п/п	Наименование темы дисциплины	Содержание
1	Введение	Предмет дисциплины, её объём, содержание и связь с другими дисциплинами учебного плана. Роль дисциплины в подготовке инженеров в области разработки средств вычислительной техники, цели и задачи дисциплины. Перечень дисциплин и разделов,усвоение которых необходимо для изучения дисциплины. Обзор литературы по курсу. Свойства алгоритма и их оценка. Временная и ёмкостная сложность. Способы оценки. Обозначения для сравнения скорости роста функций. Оценка сложности алгоритма по его структуре. Последовательные шаги. Выбор. Цикл.
2	Данные и алгоритмы	Понятие данных. Примитивные данные. Стандартные структуры данных. Множества, отображения, последовательности. Очереди и стеки. Память ЭВМ и структуры данных. Связь понятий «структура данных» в теории алгоритмов и «объект» в программировании. Множества. Основные понятия. Способы представления множеств в ЭВМ
3	Генерация тестов	Автоматическая генерация тестов для программ и оборудования. Получение всех подмножеств заданного множества. Правила работы с датчиком случайных чисел и получение случайного подмножества заданного множества. Получение всех подмножеств в лексикографическом порядке. Понятие о кодовом расстоянии и его применение при создании тестов. Код Грея и его генерация. Множества с повторениями. Генерация подмножеств множества с повторениями. Генерация К элементных подмножеств. Генерация перестановок.
4	Множество как объект	Объекты и функции члены: объявление и использование. Функции члены inline по умолчанию. Создание и уничтожение объектов: конструкторы и деструкторы. Умолчания. Списки инициализации. Константные члены класса и способы их инициализации. Массивы объектов. Перегрузка операций. Общие правила. Одноместные и двуместные операции. Перегрузка операций обычной и дружественной функцией. Перегрузка префиксного и постфиксного инкремента/декремента. Использование одного класса внутри другого. Дружественные классы. Конструктор копирования и перегрузка присваивания. Оптимизация воз-вращаемого значения. Конструктор копирования и присваивание с переносом.

№ п/п	Наименование темы дисциплины	Содержание
5	Введение в Стандартную библиотеку шаблонов (STL)	Шаблоны. Обобщённые функции. Обобщённые классы. Шаблоны для стека и очереди. Стандартная б-ка шаблонов: обзор. Контейнеры. Итераторы. Функторы. Алгоритмы. Перегрузка операций разыменования и взятия адреса. Итераторы: сущность и классификация. Умные указатели.
6	Графы. Основные понятия, машинное представление и обходы	Абстрактная структура данных «отношение». Бинарное отношение на произвольном множестве и граф как подходящая модель для этого. Машинное представление графов. Деревья: основные понятия и машинное представление. Рекурсивные алгоритмы обхода дерева. Нерекурсивные алгоритмы обхода дерева: в глубину и в ширину. Временная сложность обхода дерева. Обходы графа общего вида. Обход в глубину для орграфа. Обход в ширину. Стягивающие деревья.
7	Алгоритмы на графах общего вида	Фундаментальное множество циклов и его интерпретация. Отыскание фундаментального множества циклов. Компоненты двусвязности. Алгоритм оты-скания компонент двусвязности. Сильная связность в орграфе. Эйлеровы пути. Алгоритм нахождения эйлерова цикла. Задачи на полный перебор (NP-полные задачи) и алгоритм с возвратом. Получение стягивающего дерева полным перебором. Алгоритм отыскания Гамильтонова цикла.
8	Графы с нагруженными вершинами: сортировка	Общие понятия о сортировке. Классификация видов сортировки. Оптимальная сортировка. Дерево сортировки и сортировка деревом. Быстрая сортировка. Сортировка слиянием. Сортировка вычерпыванием и поразрядная. Сортировка цепочек.
9	Графы с нагруженными рёбрами: поиск кратчайших путей	Кратчайшие пути в орграфе. Алгоритм отыскания кратчайшего пути. Кратчайший путь от фиксированной вершины: алгоритм Форда-Беллмана. Кратчайший путь в графе с неотрицательными весами: алгоритм Дейкстры. Кратчайшие пути в бесконтурном графе. Кратчайшие пути между всеми парами вершин: алгоритм Флойда. Транзитивное замыкание отношения. Алгоритм Уоршалла. Стягивающее дерево наименьшей стоимости: алгоритмы Прима и Краскала.
10	Графы с нагруженными вершинами и рёбрами: потоки в сетях и смежные задачи	Потоки в сетях: основные понятия. Общая схема отыскания максимального потока в сети. Построение вспомогательного бесконтурного графа. Алгоритм отыскания псевдомаксимального потока во вспомогательном бесконтурном графе. Наибольшее паросочетание в двудольном графе: основные понятия и алгоритм. Трансверсаль и её отыскание. Вершинное покрытие в двудольном графе.

№ п/п	Наименование темы дисциплины	Содержание
11	Жадные алгоритмы решения информационных задач	Жадные алгоритмы: основные понятия. Понятие о матроидах как основе разработки жадных алгоритмов. Примеры матроидов. Жадный алгоритм для матриц: схема Гаусса. Жадные алгоритмы для графов: алгоритм Дейкстры для отыскания кратчайших путей; схемы Прима и Краскала для стягивающего дерева наименьшей стоимости. Жадный алгоритм для трансверсалей: отыскание частичной трансверсали с наибольшим весом.
12	Практические способы решения NP-полных задач	Некоторые важные NP-полные задачи. Варианты постановки: поиск оптимума и принятие решения. Причина малой практической полезности алгоритма перебора с возвратом. Применение жадных алгоритмов для решения NP-полных задач. Примеры жадных алгоритмов для задачи коммивояжёра, раскладки по ящикам, упаковки рюкзака. Обсуждение результатов. Приложения. Эмпирические алгоритмы решения NP-полных задач. Эмпирический полиномиальный алгоритм решения задачи о раскраске графа. Муравьиный алгоритм решения задачи коммивояжёра.
13	Иерархия классов	Производные классы — что это даёт. Функции-члены. Конструкторы и деструкторы. Пример иерархии классов. Управление доступом к объектам в иерархии: поля типа и их альтернатива — виртуальные функции. Абстрактные классы. Контроль наследования: override и final в новом стандарте. Проектирование иерархии классов: отношения «является» и «содержит». Понятие о паттернах проектирования как типовых схемах построения и организации взаимодействия классов. Закрытое наследование и его альтернатива — включение. Достоинства и недостатки каждого из способов. Контроль доступа к членам базовых классов. Защищённые члены и защищённое включение. Виртуальные базовые классы. Общая идея производных классов и абстрактного базового класса на примере библиотеки фигур. Классы «фигура» и «линия». Необязательность определения ненужных функций. Пример законченной программы с иерархией классов. Множественное наследование. Множественное вхождение базового класса. Разрешение неоднозначности. Виртуальные деструкторы.

№ п/п	Наименование темы дисциплины	Содержание
14	Исключительные ситуации	Использование механизма исключительных ситуаций для проектирования реакции программы на ошибки времени выполнения. Учёт исключительных ситуаций при построении библиотек. Базовые и расширенные гарантии. Функции, не генерирующие исключений. Функции, про-зрачные для исключений. Стандартные исключительные ситуации: exception, bad_alloc и др. Неожиданные исключительные ситуации и возможности для их обработки.
15	Способы хранения в памяти изменяющихся множеств	Хеширование как обобщение способа хранения множества в массиве битов. Открытые и закрытые хеш-таблицы: способы разрешения коллизий. Рекомендации по составлению хеш-функций. Двуместные операции с множествами в форме хеш-таблиц. Значение упорядоченности для двуместных операций над множествами в форме последовательностей. Дерево двоичного поиска (ДДП) как расширяемый список с сохранением упорядоченности. Упорядоченный массив как способ хранения дерева двоичного поиска. Двуместные операции над множествами за линейное время. Дерево двоичного поиска: алгоритмы вставки и удаления.
16	Деревья двоичного поиска с автобалансировкой	Вырождение при произвольной вставке последовательности узлов. Деревья с автобалансировкой. АВЛ-дерево: определение и алгоритмы балансировки при вставке и удалении элемента множества. Красно-чёрное дерево: балансировка при вставке и удалении. 2-3-дерево. Алгоритмы вставки и удаления. Рекомендация по кодированию в виде варианта ДДП. Получение дерева с автобалансировкой из упорядоченного массива: АВЛ-и красно-чёрное дерево. Получение 2-3-дерева из упорядоченной последовательности. Двуместные операции с ДДП: алгоритм пошагового обхода двоичного дерева для поддержки схемы слияния. Пошаговый обход 2-3-дерева.

№ п/п	Наименование темы дисциплины	Содержание
17	Поддержка последовательностей в структуре данных для множеств	Хранение последовательностей в структуре данных для множеств. Варианты: обход, номера, списки, справочный массив. Примитивы: определение номера по ключу и ключа по номеру. Двуместные операции над последовательностями: вставка, замена, удаление по интервалу номеров и по образцу, конкатенация и размножение. Множества с повторениями в хеш-таблице, ДДП, АВЛ-дереве, 2-3-дереве для обеспечения поддержки произвольной последовательности. Особенности кодирования ДДП, облегчающие реализацию зеркальных вариантов балансировки. Реализация универсального алгоритма двуместной операции на примере АВЛ-дерева со справочным массивом указателей.
18	Пользовательские дополнения к Стандартной библиотеке шаблонов (STL)	Особенности STL в новых стандартах языка (C++11/14/17/20). Функциональные объекты: указатели на функцию и функторы. Лямбда — функциональный объект, введённый стандартом C++11. Общая характеристика функциональных объектов с точки зрения применимости в алгоритмах. Пользовательский контейнер, пригодный для работы с алгоритмами STL. Пользовательские операторы чтения и вставки для хеш-таблицы и ДДП
19	Ассоциативные контейнеры	Ассоциативные контейнеры — краткий обзор. Словарь и множество. Мультисловарь и мультимножество. Множество битов. Неупорядоченные ассоциативные контейнеры.
20	Умные указатели	Умные указатели: auto_ptr и его последователи unique_ptr, shared_ptr, week_ptr в новом стандарте. Примеры использования unique_ptr. Указатель unique_ptr для работы с массивами.Пример использования shared_ptr и week_ptr.
21	Измерение временной сложности алгоритма в эксперименте на ЭВМ	Использование ЭВМ в научном эксперименте. Математические модели. Особенности постановки на ПЭВМ опытов со случайным результатом. Способы генерации исходных данных. Измерение времени решения задачи. Методика обработки результатов эксперимента: введение в регрессионный анализ. Учебный пример DemoSTL с хеш-таблицей и поддержкой последовательности с использованием вектора итераторов. Объявление класса MyCont и его составляющих, конструкторы и деструктор, перегрузка присваивания. Точные способы измерения времени прохождения теста -возможности STL: библиотека stl□chrono.
22	Заключение	Порядок итоговой аттестации

4.2 Перечень лабораторных работ

Наименование лабораторной работы	Количество ауд. часов
1. Данные и алгоритмы	2
2. Генерация тестов	2
3. Множество как объект	2
4. Машинное представление и обходы графов	2
5. Алгоритмы на графах общего вида	2
6. Графы с нагруженными вершинами: сортировка	2
7. Графы с нагруженными рёбрами: поиск кратчайших путей	2
8. Графы с нагруженными вершинами и рёбрами: потоки в сетях и смежные задачи	2
9. Жадные алгоритмы решения информационных задач	2
10. Иерархия классов	2
11. Исключительные ситуации	2
12. Способы хранения в памяти изменяющихся множеств	2
13. Деревья двоичного поиска с автобалансировкой	2
14. Поддержка последовательностей в структуре данных для множеств	2
15. Пользовательские дополнения к Стандартной библиотеке шаблонов	2
16. Ассоциативные контейнеры	2
17. Измерение временной сложности алгоритма в эксперименте на ЭВМ	2
Итого	34

4.3 Перечень практических занятий

Наименование практических занятий	Количество ауд. часов
1. Данные и алгоритмы	2
2. Генерация тестов	2
3. Множество как объект	2
4. Введение в Стандартную библиотеку шаблонов	2
5. Алгоритмы на графах общего вида	2
6. Графы с нагруженными вершинами: сортировка	2
7. Графы с нагруженными рёбрами: поиск кратчайших путей	2
8. Графы с нагруженными вершинами и рёбрами: потоки в сетях и смежные задачи	2
9. Практические способы решения NP-полных задач	2
10. Иерархия классов	2
11. Исключительные ситуации	2
12. Способы хранения в памяти изменяющихся множеств	2
13. Деревья двоичного поиска с автобалансировкой	2
14. Поддержка последовательностей в структуре данных для множеств	2

Наименование практических занятий	Количество ауд. часов
15. Пользовательские дополнения к Стандартной библиотеке шаблонов	2
16. Умные указатели	2
17. Измерение временной сложности алгоритма в эксперименте на ЭВМ	2
Итого	34

4.4 Курсовое проектирование

Цель работы (проекта): 1. (3 семестр) Реализация программы с пользовательской структурой данных и исследование алгоритма для работы с ней.
2. (4 семестр) Реализация в программе пользовательского контейнера с поддержкой заданного набора операций и измерение временной сложности цепочки операций с множествами и последовательностями в машинном эксперименте.

Содержание работы (проекта): 1. (3 семестр). Найти в рекомендованных литературных источниках описание алгоритма для работы с графами в соответствии с индивидуальным заданием. Сформулировать поставленную задачу в терминах теории множеств. Выбрать структуру данных, оптимальную для представления исходных данных и результата. Реализовать алгоритм в виде программы на ЭВМ. Предложить тесты для проверки правильности работы программы. Оценить временную сложность алгоритма и его реализации на ЭВМ. Найти в лекциях на Образовательном портале ЛЭТИ и рекомендованных литературных источниках описание алгоритма для работы с графами в соответствии с индивидуальным заданием. Сформулировать поставленную задачу в терминах теории множеств. Выбрать структуру данных, оптимальную для представления исходных данных и результата. Реализовать алгоритм в виде программы на ЭВМ. Предложить тесты для проверки правильности работы программы. Оценить временную сложность алгоритма и его реализации на ЭВМ.
2 (4 семестр). Составить и отладить программу, реализующую пользовательский контейнер, поддерживающий заданный набор операций с множествами

и последовательностями с помощью Стандартной библиотеки шаблонов. Организовать в этой программе измерение времени выполнения цепочки из реализованных операций для заданного размера входа. Получить набор измерений времени решения задачи для последовательно увеличивающихся размеров входа. Выполнить статистическую обработку полученных измерений для получения оценки временной сложности выполнения цепочки операций. Дать заключение о том, подтвердилась или не подтвердилась гипотеза о теоретической оценке временной сложности. Составить и отладить программу, реализующую пользовательский контейнер, поддерживающий заданный набор операций с множествами и последовательностями с помощью Стандартной библиотеки шаблонов. Организовать в этой программе измерение времени выполнения цепочки из реализованных операций для заданного размера входа. Получить набор измерений времени решения задачи для последовательно увеличивающихся размеров входа. Выполнить статистическую обработку полученных измерений для получения оценки временной сложности выполнения цепочки операций. Дать заключение о том, подтвердилась или не подтвердилась гипотеза о теоретической оценке временной сложности.

Оформление пояснительной записки на курсовую работу выполняется в соответствии с требованиями к студенческим работам, принятым в СПбГЭТУ. В частности, отчёт по курсовой работе должен содержать:

-титульный лист;

-аннотацию;

-лист содержания;

-задание на работу;

-описание поставленной задачи в терминах теории множеств с обязательной ссылкой на первоисточники (не менее двух);

-обоснование по выбору структур данных для эффективной реализации алгоритма;

-описание алгоритма и пояснения к его реализации;

-результаты тестирования, доказательство корректности реализации алгоритма;

-оценка временной сложности алгоритма;

-выводы по результатам исследования алгоритма;

-перечень ссылочных документов;

-приложение: текст программы (в составе отчёта) или перечень файлов.

Объём пояснительной записки составляет, таким образом, в среднем 8 -10 страниц, без учёта прилагаемых к нему текстов программ.

Отчётные документы по курсовым работам сдаются преподавателю на занятии или присылаются по почте в электронном виде. Тексты программ, включаемые в отчёты или прилагаемые к ним, должны быть пригодны для компиляции.

Темы:

№ п/п	Название темы	Перевод темы
1	Получение множества двудольных компонент ориентированного графа.	Preparation of a plurality of components of bipartite undirected graph
2	Проверка ориентированного графа на ацикличность	Checking on the a cyclic directed graph
3	Поиск изоморфного неориентированного подграфа	Searchun directed subgraph isomorphic
4	Отыскание максимального паросочетания в произвольном неориентированном графе	Finding a maximum matching in an arbitraryun directed graph
5	Построение полного множества циклов для ориентированного графа	Construction of the full set of cycles for a directed graph
6	Проверка наличия цикла отрицательной стоимости в ориентированном графе с нагруженными рёбрами	Check for negative cost cycles in a directed graph with loaded ribs
7	Построение фундаментального множества циклов в неориентированном графе	Construction of a fundamental set of cycles in an undirected graph

4.5 Реферат

Реферат не предусмотрен.

4.6 Индивидуальное домашнее задание

Индивидуальное домашнее задание не предусмотрено.

4.7 Доклад

Доклад не предусмотрен.

4.8 Кейс

Кейс не предусмотрен.

4.9 Организация и учебно-методическое обеспечение самостоятельной работы

Изучение дисциплины сопровождается самостоятельной работой студентов с рекомендованными преподавателем литературными источниками и информационными ресурсами сети Интернет.

Планирование времени для изучения дисциплины осуществляется на весь период обучения, предусматривая при этом регулярное повторение пройденного материала. Обучающимся, в рамках внеаудиторной самостоятельной работы, необходимо регулярно дополнять сведениями из литературных источников материал, законспектированный на лекциях. При этом на основе изучения рекомендованной литературы целесообразно составить конспект основных положений, терминов и определений, необходимых для освоения разделов учебной дисциплины.

Особое место уделяется консультированию, как одной из форм обучения и контроля самостоятельной работы. Консультирование предполагает особым образом организованное взаимодействие между преподавателем и студентами, при этом предполагается, что консультант либо знает готовое решение, которое он может предписать консультируемому, либо он владеет способами деятельности, которые указывают путь решения проблемы.

Текущая СРС	Примерная трудоемкость, ач
Работа с лекционным материалом, с учебной литературой	8
Опережающая самостоятельная работа (изучение нового материала до его изложения на занятиях)	0
Самостоятельное изучение разделов дисциплины	9
Выполнение домашних заданий, домашних контрольных работ	0
Подготовка к лабораторным работам, к практическим и семинарским занятиям	10
Подготовка к контрольным работам, коллоквиумам	0
Выполнение расчетно-графических работ	0
Выполнение курсового проекта или курсовой работы	100
Поиск, изучение и презентация информации по заданной проблеме, анализ научных публикаций по заданной теме	0
Работа над междисциплинарным проектом	0
Анализ данных по заданной теме, выполнение расчетов, составление схем и моделей, на основе собранных данных	10
Подготовка к зачету, дифференцированному зачету, экзамену	45
ИТОГО СРС	182

5 Учебно-методическое обеспечение дисциплины

5.1 Перечень основной и дополнительной литературы, необходимой для освоения дисциплины

№ п/п	Название, библиографическое описание	К-во экз. в библ.
Основная литература		
1	Колинько, Павел Георгиевич. Пользовательские структуры данных [Текст] : учеб.-метод. пособие / П. Г. Колинько, 2020. -63, [1] с.	110
Дополнительная литература		
1	Страуструп, Бьярне. Программирование: принципы и практика с использованием C++ [Текст] : пер. с англ. / Б. Страуструп, 2019. -1328 с.	15
2	Поздняков, Сергей Николаевич. Дискретная математика [Текст] : учеб. для вузов по направлениям подгот. ”Информатика и вычисл. техника”, ”Информационные системы”, ”Информационная безопасность” / С.Н. Поздняков, С.В. Рыбин, 2008. -448 с.	491
3	Новиков, Федор Александрович. Дискретная математика для программистов [Текст] : для студентов вузов по направлению подгот. ”Информатика и вычисл. техника” / Ф.А. Новиков, 2008. -383 с.	38
4	Ахо, Альфред В. Построение и анализ вычислительных алгоритмов [Текст] : монография / А.Ахо, Д.Хопкрофт, Дж.Д.Ульман; Пер. с англ. А.О.Слисенко; Под ред. Ю.В.Матиясевича, 1979. -536 с.	11
5	Ахо, Альфред В. Структуры данных и алгоритмы [Текст] : монография / А.В.Ахо, Д.Э.Хопкрофт, Дж.Д.Ульман, 2001. -382 с.	43
6	Липский, Витольд. Комбинаторика для программистов [Текст] / В. Липский ; пер. с польск. В. А. Евстигнеева и О.А. Логиновой ; под ред. А. П. Ершова, 1988. -213 с.	6
7	Макконелл Дж. Основы современных алгоритмов [Текст] : учеб. пособие для вузов по направлению подгот. специалистов ”Информатика и вычислительная техника” : пер. с англ. / Дж. Макконелл ; пер. с англ. под ред. С.К. Ландо ; доп. М.В. Ульянова, 2004. -366 с.	5
8	Кнут Д.Э. Искусство программирования для ЭВМ [Текст] : учеб. пособие : пер. с англ. Т. 3 : Сортировка и поиск / под ред. Ю. М. Баяковского, В. С. Штаркмана, 1978. -844 с.	26
9	Кормен, Томас. Алгоритмы: построение и анализ [Текст] : Учеб. / Т. Кормен; Ч.Лейзерсон, Р.Ривест; Пер. с англ. под ред. А.Шен, 1999. -955 с.	8
10	Седжвик, Роберт. Фундаментальные алгоритмы на C++ [Текст] : [Пер. с англ.]. Ч. 5 : Алгоритмы на графах : монография, 2002. -484 с.	45

5.2 Перечень ресурсов информационно-телекоммуникационной сети «Интернет», используемых при освоении дисциплины

№ п/п	Электронный адрес
1	C++ Reference: Актуальная справка о языке C++11/14/17/20/23, шаблонах, библиотечных функциях с примерами их использованияhttp://ru.cppreference.com
2	То же, на языке оригиналаhttp://en.cppreference.com
3	Язык программирования C++ для профессионалов (Учебный курс).https://intuit.ru/studies/courses/98/98/info

5.3 Адрес сайта курса

Адрес сайта курса: https://vec.etu.ru/moodle/course/view.php?id=14844

6 Критерии оценивания и оценочные материалы

6.1 Критерии оценивания

Для дисциплины «Алгоритмы и структуры данных» предусмотрены следующие формы промежуточной аттестации: экзамен, зачет с оценкой.

Экзамен

Оценка	Описание
Неудовлетворительно	Курс не освоен. Студент испытывает серьезные трудности при ответе на ключевые вопросы дисциплины
Удовлетворительно	Студент в целом овладел курсом, но некоторые разделы освоены на уровне определений и формулировок теорем
Хорошо	Студент овладел курсом, но в отдельных вопросах испытывает затруднения. Умеет решать задачи
Отлично	Студент демонстрирует полное овладение курсом, способен применять полученные знания при решении конкретных задач.

Зачет с оценкой

Оценка	Описание
Неудовлетворительно	Курс не освоен. Студент испытывает серьезные трудности при ответе на ключевые вопросы дисциплины
Удовлетворительно	Студент в целом овладел курсом, но некоторые разделы освоены на уровне определений и формулировок теорем
Хорошо	Студент овладел курсом, но в отдельных вопросах испытывает затруднения. Умеет решать задачи
Отлично	Студент демонстрирует полное овладение курсом, способен применять полученные знания при решении конкретных задач.

Особенности допуска

К экзамену в 3 семестре допускаются студенты, выполнившие лабораторные работы и подготовившие и защитившие отчёты по ним, а также выполнившие и защитившие курсовую работу.

Экзамен проводится в виде теста из 20 вопросов, на каждый вопрос даётся по одной минуте, правильный ответ оценивается в один балл. Результирующая оценка формируется по результатам итогового теста, защит отчетов лабораторных работ.

К дифференцированному зачету в 4 семестре допускаются студенты, выполнившие лабораторные работы и подготовившие и защитившие отчёты по ним, а также выполнившие и защитившие курсовую работу.

Дифференцированный зачёт проводится в виде теста из 20 вопросов, на каждый вопрос даётся по одной минуте, правильный ответ оценивается в один балл. Результирующая оценка формируется по результатам итогового теста, защит отчетов лабораторных работ.

6.2 Оценочные материалы для проведения текущего контроля и промежуточной аттестации обучающихся по дисциплине

Вопросы к экзамену

№ п/п	Описание
1	Какой класс временной сложности алгоритма является самым широким?
2	Алгоритм какой временной сложности является самым предпочтительным в общем случае?
3	Алгоритм какой временной сложности является оптимальным для поэлементной обработки множества мощностью n
4	Какой способ размещения множеств в памяти обеспечивает самый быстрый алгоритм выполнения двуместной операции над ними?
5	Сколько различных подмножеств можно получить для множества мощностью n?
6	Можно ли использовать ключевое слово struct в том же смысле, что и class?
7	Можно ли объявить класс, который будет играть роль элемента множества списка и множества в целом?
8	Что лучше — объявить класс-элемент списка как структуру внутри класса-списка или объявить два отдельных класса, связав их дружбой?

9	Каковы преимущества объектно-ориентированного подхода к программированию по сравнению с процедурным?
10	Каким способом можно перегрузить операцию «|»?
11	Каким способом можно перегрузить операцию «=»?
12	Как различить перегрузку префиксного и постфиксного инкрементов «++»?
13	Можно ли при перегруженных «|» и «=» применять комбинированную операцию «|=»?
14	Путь в графе -это ...
15	Можно ли ускорить выполнение оператора «A = B | C»
16	В каком случае функция-член должна возвращать ссылку на объект?
17	Граф какого вида больше подходит в качестве модели бинарного отношения на произвольном конечном множестве?
18	Путь в графе -это ...
19	Длина пути между парой вершин в ненагруженном графе — это…
20	Цикл в графе называется элементарным, если…
21	Можно ли для представления графа в памяти использовать машинное слово?
22	Какое машинное представление графа может быть сделано самым компактным?
23	В каком случае представление графа в виде списков смежности считается оптимальным?
24	Какой способ размещения двоичного дерева в памяти является самым универсальным?
25	Какой из алгоритмов обхода двоичного дерева с разметкой вершин является оптимальным?
26	Какой обход двоичного дерева рекомендуется применить для определения мощности каждого из образующих его поддеревьев?
27	Какой обход двоичного дерева не годится создания его в виде разветвляющегося списка из заданной последовательности битов?
28	Какой алгоритм обхода двоичного дерева не требует использования стека?
29	Какой способ размещения множества в памяти — самый удобный для проверки принадлежности элемента множеству?
30	template <class T> void Swap(T& a, T& b) \textbraceleft T c(a); a= b; b=c; \textbraceright Сколько вариантов функции Swap создаёт компилятор при определении её как шаблона, если функция не вызывается?
31	Сколько может быть рёбер в стягивающем дереве графа из 100 вершин?
32	Какова временная сложность оптимального алгоритма отыскания эйлерова пути в неориентированном графе из n вершин и m рёбер?
33	template <class T> void Swap(T& a, T& b) \textbraceleft T c(a); a= b; b=c; \textbraceright Сколько вариантов функции Swap создаёт компилятор при определении её как шаблона, если функция не вызывается?
34	Какой алгоритм рекомендуется использовать для определения расстояний от стартовой вершины до всех остальных вершин в неориентированном ненагруженном графе?
35	Сколько фундаментальных циклов имеется в неориентированном графе из 100 вершин и 1000 рёбер?
36	Какова максимальная длина элементарного пути между произвольной парой вершин в неориентированном графе из 100 вершин и 1000 рёбер?

37	Сколько различных циклов может быть обнаружено в неориентированном графе из 10 вершин и 45 рёбер?
38	Что является результатом работы алгоритма отыскания компонент двусвязности неорграфа G = <V, E>?
39	Что является результатом работы алгоритма отыскания стягивающего дерева орграфа G = <V, E>?
40	Какой способ размещения множества в памяти — самый удобный для проверки принадлежности элемента множеству?
41	Существует ли эйлеров путь в неориентированном графе, являющемся псевдоциклом?
42	Какова временная сложность оптимального алгоритма отыскания гамильтонова пути в неориентированном графе из n вершин и m рёбер?
43	Если элементы последовательности можно сравнивать только целиком, алгоритм сортировки какой сложности является оптимальным?
44	Какой из алгоритмов сортировки требует минимума дополнительной памяти?
45	Как следует сортировать цепочки, если они сильно различаются по длине?

Вариант экзаменационного теста

«АЛГОРИТМЫ И СТРУКТУРЫ ДАННЫХ»

Вариант 190103 Студент (Ф.И.О) _____ Номер студ.билета _____ Дата сдачи теста _____

1. Какой класс временной сложности алгоритма является самым широким?

а) O(n)

б) O(1)

в) **O(e^n)**

г) O(log n)

2. Алгоритм какой временной сложности является оптимальным для поэлементной обработки множества мощностью n?

а) O(n^2)

б) **O(n)**

в) O(log n)

г) O(e^n)

д) O(1)

3. Какой способ размещения множества в памяти — самый экономный?

а) массив элементов множества

б) массив битов

в) **машинное слово**

г) список элементов множества

4. При какой форме представления множеств в памяти можно задать множеств
константу?

а) массив элементов

б) массив битов

в) машинное слово

г) **пригодны все три формы**

5. Объявлен массив целых:

int A[100]{0};

Какие элементы массива инициализируются нулём?

а) никакие: ошибка компилятора

б) **все**

в) только нулевой

6. Какой способ размещения двоичного дерева в памяти является самым
компактным?

а) **машинное слово**

б) пара массивов номеров вершин

в) разветвляющийся список

г) вектор битов

7. Какой из алгоритмов обхода двоичного дерева с разметкой вершин является оптимальным?

а) прямой обход

б) обход в ширину

в) симметричный обход

г) **любой из перечисленных**

д) случайный обход

е) обратный обход

8. Какой обход двоичного дерева пригоден для определения расстояния от корня до каждой из вершин?

а) обход в ширину

б) прямой обход

в) симметричный обход

г) **любой из перечисленных**

9. Какой алгоритм обхода двоичного дерева пригоден для определения его высоты?

а) прямой обход

б) **все перечисленные**

в) обратный обход

г) обход в ширину

10. Сколько фаз увеличения потока может потребоваться в алгоритме Диница для сети из 500 вершин и 2500 рёбер?

а) 2499

б) 999

в) 501

г) **499**

11. Какова мощность совершенного паросочетания в произвольном графе из 100 вершин и 2500 рёбер?

а) 2500

б) **50**

в) 100

12. Имеется набор грузов различного объёма и стоимости. В каком порядке следует укладывать грузы в рюкзак заданного объёма, чтобы стоимость поместившихся грузов была максимальна?

а) в порядке неубывания объёма

б) в порядке неубывания отношения объёма к стоимости

в) в порядке неубывания стоимости

г) в порядке невозрастания объёма

д) **испытать все перестановки**

Вопросы к дифф.зачету

№ п/п	Описание
1	Какой базовый класс лучше всего использовать для производного класса «треугольник»?
2	Какой тип наследования следует выбрать: private, public или protected?
3	Можно ли вообще не указывать тип наследования?
4	В чём смысл объявления функций в базовом классе как виртуальных?
5	Нужно ли объявлять виртуальной функцию в производном классе, если в базовом она уже объявлена таковой?
6	Можно ли потребовать от компилятора проверить корректность объявления виртуальной функции в производном классе и как это сделать?
7	Можно ли запретить виртуальную функцию в классах-наследниках?
8	Что такое «чисто виртуальная функция»?

9	Обязательно ли переопределять все функции-члены базового класса в производном классе?
10	Как запретить для объекта вызов конструктора по умолчанию?
11	Как запретить вызов конструктора для использования в качестве неявного преобразователя типа?
12	Каким образом следует инициализировать объект базового класса в конструкторе производного класса? Всегда ли это нужно делать?
13	Каким требованиям должны удовлетворять классы, чтобы между ними можно было сформировать отношение «является»?
14	Как установить между двумя классами отношение «содержит»?
15	Что такое исключительная ситуация при выполнении программы?
16	Как можно выявить исключительную ситуацию?
17	Что можно предпринять при выявлении исключительной ситуации?
18	Можно ли передать в обработчик особых ситуаций какую-либо информацию о произошедшем событии?
19	Можно ли обработать неизвестную особую ситуацию?
20	Можно ли сделать обработчик ситуации пустым?
21	Что можно предпринять, если для корректной обработки ситуации в данном месте программы у обработчика недостаточно данных?
22	Если требуется несколько обработчиков особых ситуаций, в каком порядке следует их размещать в программе?
23	Можно ли перехватывать одним обработчиком несколько различных особых ситуаций?
24	Как действуют обработчики в случае, когда никакой особой ситуации не произошло?
25	Как следует размещать блоки контроля, чтобы получить безопасный программный код?
26	Можно ли продолжить выполнение программы с точки, в которой выявлена ошибка, после внесения исправлений в данные?
27	Какой объём памяти нужно выделять под хеш-таблицу для хранения множеств со средней мощностью 50?
28	Каким требованиям должна удовлетворять «хорошая» хеш-функция?
29	Можно ли построить хеш-таблицу, в которой не будет коллизий?
30	Каков оптимальный алгоритм выполнения двуместной операции над множествами в хеш-таблице? Какова его временная сложность?
31	Можно ли хранить в хеш-таблице множество с повторениями?
32	Какова временная сложность операций вставки и удаления элемента для хеш-таблицы?
33	Почему для операций с хеш-таблицей дают две оценки временной сложности — сложность в худшем случае и сложность в среднем?
34	Что такое вырождение хеш-таблицы и как его избежать?
35	Каким способом следует разметить дерево, чтобы в нём был возможен двоичный поиск? Как его следует нагрузить?
36	Почему для операций с двоичным деревом дают две оценки сложности — «в худшем случае» и «в среднем»? Почему не рассматривается «лучший» случай?
37	Можно ли воспрепятствовать вырождению ДДП?

38	Каков оптимальный алгоритм двуместной операции над множествами, представленными деревьями двоичного поиска?
39	Может ли при двуместной операции над множествами в ДДП получиться вырожденное дерево-результат?
40	Как сделать не вырождающееся ДДП? Зачем оно может понадобиться?
41	Какая структура данных является оптимальной для хранения дерева двоичного поиска?
42	Почему для хранения произвольной последовательности структуру данных для множества (хеш-таблицу или ДДП) приходится дорабатывать?
43	Влияет ли доработка структур данных для множеств для поддержки последовательностей на временную сложность операций над множествами?
44	Какова оптимальная доработка структуры данных и временная сложность для операции исключения части последовательности между указанными позициями?
45	Что такое стандартный контейнер библиотеки STL? Чем он отличается от обычного объекта?
46	Какой стандартный контейнер можно считать наиболее подходящим для работы с множествами?
47	Можно ли использовать стандартные контейнеры для множеств, на которых не определено отношение полного порядка?
48	Существуют ли ограничения на применение стандартных алгоритмов двуместных операций над множествами в контейнерах?
49	С какой целью может понадобиться оформить пользовательскую структуру данных как стандартный контейнер?
50	Каков минимально необходимый набор средств для превращения пользовательской структуры данных в полноценный контейнер?
51	Зачем нужны гарантии устойчивости алгоритмов относительно исключений?
52	Почему базовых гарантий устойчивости алгоритма к исключениям может быть недостаточно?
53	Почему базовых гарантий устойчивости алгоритма к исключениям может быть недостаточно?

Вариант теста (дифф. зачет)

Пример тестовых заданий

1. Сколько может быть рёбер в неориентированном графе из 100 вершин?

а) 99

б) 10000

в) **4950**

2. Какова максимальная длина элементарного пути между произвольной парой вершин в неориентированном графе из 100 вершин и 1000 рёбер?

а) **99**

б) 1000

в) 901

г) 101

3. Какова временная сложность оптимального алгоритма отыскания компонент двусвязности в неориентированном графе из n вершин и m рёбер?

а) O(n log n)

б) **O(n + m)**

в) O(nm)

г) O(n^2)

4. Какова временная сложность оптимального алгоритма отыскания сильно связных компонент в ориентированном графе из n вершин и m рёбер?

а) O(n log n)

б) O(n^2)

в) **O(n + m)**

г) O(nm)

5. Сколько нужно создать черпаков для сортировки вычерпыванием следующей последовательности:

<23, 34, 1, 55, 16, 10, 11, 43, 98, 35, 17, 19, 81, 22 >?

а) 10

б) 14

в) **100**

г) 1000

6. Какой элемент окажется в начале результата сортировки следующей

последовательности цепочек десятичных цифр: <23, 34, 9, 55, 16, 1000, 11, 403, 980, 35, 107, 19, 810, 2200 >?

а) 9

б) **1000**

в) 11

г) 2200

7. Задан граф из 100 вершин и 1000 рёбер. Рёбра нагружены весом, не превышающим 10. Какое значение «бесконечности» достаточно использовать в качестве веса несуществующих рёбер в матрице весов?

а) **1000**

б) 9999

в) 10000

г) 2000

08. Какой алгоритм целесообразно использовать для определения минимального вершинного покрытия в двудольном графе?

а) алгоритм Флойда

б) **алгоритм Хопкрофта-Карпа**

в) алгоритм Диница

г) перебор с возвратом

9. Является ли увеличивающая цепь самым коротким путём из источника в сток во вспомогательной бесконтурной сети, полученной первой фазой схемы Диница?

а) **да, несомненно**

б) это не путь

в) нет, самым длинным

10. Какова мощность совершенного паросочетания в произвольном графе из 100 вершин и 2500 рёбер?

а) 100

б) 2500

в) **50**

Примечание. Система тестирования предлагает студенту 20 вопросов, случайно выбранных из имеющегося набора, со случайным порядком возможных ответов.

Весь комплект контрольно-измерительных материалов для проверки сформированности компетенции (индикатора компетенции) размещен в закрытой части по адресу, указанному в п. 5.3

6.3 График текущего контроля успеваемости

Неделя	Темы занятий	Вид контроля
1	Данные и алгоритмы Генерация тестов	
2		
3		
4		Отчет по лаб. работе
5	Множество как объект	
6		
7		
8		Отчет по лаб. работе
9	Графы. Основные понятия, машинное представление и обходы Алгоритмы на графах общего вида	
10		
11		
12		Отчет по лаб. работе
13	Графы с нагруженными вершинами: сортировка Графы с нагруженными рёбрами: поиск кратчайших путей Графы с нагруженными вершинами и рёбрами: потоки в сетях и смежные задачи Жадные алгоритмы решения информационных задач	
14		
15		
16		Защита КР / КП
18	Иерархия классов	
19		
20		
21		Отчет по лаб. работе
22	Исключительные ситуации	
23		
24		
25		Отчет по лаб. работе
26	Способы хранения в памяти изменяющихся множеств Деревья двоичного поиска с автобалансировкой Поддержка последовательностей в структуре данных для множеств Пользовательские дополнения к Стандартной библиотеке шаблонов (STL)	
27		
28		
29		Отчет по лаб. работе
30	Ассоциативные контейнеры Измерение временной сложности алгоритма в эксперименте на ЭВМ	
31		
32		
33		
34		Защита КР / КП

6.4 Методика текущего контроля

на лекционных занятиях

Текущий контроль включает в себя контроль посещаемости (не менее ***80***

% занятий), по результатам которого студент получает допуск на экзамен.

на лабораторных занятиях

- Порядок выполнения лабораторных работ, подготовки отчётов и их защиты

В процессе обучения по дисциплине «Алгоритмы и структуры данных” в течение двух семестров студент обязан выполнить 17 лабораторных работ. Под выполнением лабораторных работ подразумевается подготовка к работе, разработка и тестирование программы на ЭВМ, подготовка отчета и его защита. Выполнение лабораторных работ студентами осуществляется в бригадах до 3 человек – для приобретения навыков работы в команде и парного программирования. Оформление отчёта студентами осуществляется в количестве одного отчёта на бригаду в электронном виде в соответствии с принятыми в СПбГЭТУ правилами оформления студенческих работ. Отчёт оформляется после завершения отладки программы, выполнения исследований с ней и представляется преподавателю на проверку после демонстрации работы программы. Если работа выполнялась бригадой, отчёт защищается каждым студентом индивидуально. Студент отвечает на вопросы по теоретической части, или по тексту программы, или по последующей обработке результатов. В случае если студент демонстрирует достаточное знание вопроса, работа считается защищённой.

На защите лабораторной работы студент должен показать: понимание созданного им программного кода, умение объяснять особенности реализованного алгоритма и возможные области его применения и т.д., умение обосновать выбор наиболее подходящих структур данных для алгоритма, доказать полноту его тестирования, зафиксировать навыки и умения, приобретенные при выполнении лабораторной работы.

Результаты работ оцениваются по десятибалльной системе. Работа может быть принята при оценке не менее 6 баллов, что соответствует ”удовле-

творительно” и означает, что студент не провёл качественное тестирование и не смог представить развёрнутых выводов по работе, допустил небрежность в представленном отчёте. При оценке 5 и менее баллов работа признаётся неудовлетворительной и должна быть доделана.

7 баллов (”хорошо”) означает, что тестирование проведено и выводы сделаны, но есть замечания к реализации алгоритма и качеству кодирования.

8 баллов (”отлично”) ставится за работы, выполненные без существенных замечаний как по исполнению, так и по оформлению отчёта.

9 баллов может получить студент, предложивший новые идеи для реализации заданного алгоритма, выполнивший серьёзное его исследование и получивший новые результаты.

на практических занятиях

Текущий контроль включает в себя контроль посещаемости (не менее 80 % занятий), по результатам которого студент получает допуск на экзамен.

В ходе проведения практических занятий студенты привлекаются к как можно более активному участию в дискуссиях, решении задач, обсуждению и т. д. При этом активность студентов учитывается преподавателем, как один из способов текущего контроля на практических занятиях.

Контроль самостоятельной работы студентов

Контроль самостоятельной работы студентов осуществляется на лекционных, лабораторных и практических занятиях студентов по методикам, описанным выше.

При выполнении курсовой работы

Текущий контроль при выполнении курсовой работы осуществляется в соответствии с методическими указаниями по курсовом проектированию и заданием на курсовую работу.

Оформление пояснительной записки на курсовую работу выполняется в соответствии с требованиями к студенческим работам, принятым в СПбГЭТУ.

Защита курсового проекта (работы) осуществляется в соответствии с требованиями «Положения о промежуточной аттестации».

Результаты курсовой работы тоже предварительно оцениваются по десятибалльной системе, как указано выше для лабораторных работ. Итоговая оценка - традиционная четырёхбалльная.

Курсовая работа за третий семестр оценивается следующим образом:

Оценку ”отлично” получает работа, выполненная в полном соответствии с установленными требованиями:

- имеется развёрнутая постановка задачи на языке теории множеств со ссылкой не менее чем на два первоисточника;

- сделан обоснованный выбор структур данных для представления основного и промежуточных результатов;

- приложенная к работе программа компилируется и действительно даёт объявленный результат;

- тестирование проведено достаточно полно и действительно доказывает работоспособность программы;

- сделаны выводы об оптимальности реализации алгоритма и его временной сложности.

При наличии серьёзных замечаний по одному - двум из указанных пунктов ставится оценка ”хорошо”, по трём и более - ”удовлетворительно”. Оценка ”неудовлетворительно” ставится, если приложенная к работе программа не компилируется, не решает поставленную задачу, не даёт заявленный в отчёте о работе результат.

Курсовая работа за четвёртый семестр получает оценку ”отлично”, ес-

ли удовлетворяет следующим требованиям:

- для каждой операции с данными реализован оптимальный алгоритм;

- цепочка тестируемых операций с данными является представительной и не вырождается:

- статистический эксперимент поставлен грамотно, приняты меры для предотвращения смещения в результатах;

- полученная в результате эксперимента оценка временной сложности операций с данными совпадает с ожидаемой или сделаны разумные предположения о причинах возможного несоответствия. При наличии замечаний по одному-двум указанным пунктам ставится оценка ”хорошо”, по трём и более - ”удовлетворительно”. Оценка ”неудовлетворительно” ставится, если экспериментальная оценка временной сложности оттестированной цепочки операций не совпадает с ожидаемой и студент не в состоянии указать причины этого.

Отчётные документы по лабораторным и курсовым работам сдаются преподавателю на занятии или присылаются по почте в электронном виде. Тексты программ, включаемые в отчёты или прилагаемые к ним, должны быть пригодны для компиляции.

Критерии оценивания результатов тестирования на экзамене и дифференцированном зачёте.

Тест состоит из выбранных из базы случайным образом 20 вопросов, за правильный ответ на каждый из которых присуждается 1 балл. На ответы даётся 20 минут. Для получения оценки ”отлично” студент должен набрать 19 или 20 баллов, на ”хорошо” - 17-18, на ”удовлетворительно” - 15-16. Результат теста утверждается после собеседования с преподавателем. Студенты, набравшие менее 15 баллов, получают ”неудовлетворительно”, т.е. не аттестуются по предмету и должны будут после подготовки пройти тестирование повторно.

7 Описание информационных технологий и материально-технической базы

Тип занятий	Тип помещения	Требования к помещению	Требования к программному обеспечению
Лекция	Лекционная аудитория	Количество посадочных мест – в соответствии с контингентом; рабочее место преподавателя, маркерная доска, экран, проектор с разрешением Full HD, ноутбук	1) ОС LINUX; 2) Open Office 3.1 или выше; 3) Система программирования Code Bloks 20.3 с поддержкой компилятора C++ стандарта не ниже C++17
Лабораторные работы	Лаборатория	Оснащено компьютерами с подключением к сети Интернет. Количество посадочных мест – в соответствии с контингентом; рабочее место преподавателя.	1) Ос linux; 2) Open Office 3.1 или выше; 3) Система программирования Code Bloks 20 с поддержкой компилятора C++ стандарта не ниже 17
Практические занятия	Лаборатория	Оснащено компьютерами с подключением к сети Интернет. Количество посадочных мест – в соответствии с контингентом; рабочее место преподавателя.	1) ОС LINUX; 2) Open Office 3.1 или выше; 3) Система программирования Code Bloks 20.3 с поддержкой компилятора C++ стандарта не ниже C++17
Самостоятельная работа	Помещение для самостоятельной работа	Оснащено компьютерной техникой с возможностью подключения к сети «Интернет» и обеспечением доступа в электронную информационно-образовательную среду университета.	1) ОС LINUX; 2) Open Office 3.1 или выше; 3) Система программирования Code Bloks 20.3 с поддержкой компилятора C++ стандарта не ниже C++17

8 Адаптация рабочей программы для лиц с ОВЗ

Адаптированная программа разрабатывается при наличии заявления со стороны обучающегося (родителей, законных представителей) и медицинских показаний (рекомендациями психолого-медико-педагогической комиссии). Для инвалидов адаптированная образовательная программа разрабатывается в соответствии с индивидуальной программой реабилитации.

ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ

№ п/п	Дата	Изменение	Дата и номер протокола заседания УМК	Автор	Начальник ОМОЛА