МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ СИСТЕМ
УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ»
(ТУСУР)

УТВЕРЖДАЮ
Директор департамента образования

Документ подписан электронной подписью
Сертификат: 1c6cfa0a-52a6-4f49-aef0-5584d3fd4820
Владелец: Троян Павел Ефимович
Действителен: с 19.01.2016 по 16.09.2019

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ

Алгоритмы и структуры данных

Уровень образования: **высшее образование - бакалавриат**
Направление подготовки / специальность: **09.03.04 Программная инженерия**
Направленность (профиль) / специализация: **Проектирование и разработка программных продуктов**
Форма обучения: **заочная**
Факультет: **ЗиВФ, Заочный и вечерний факультет**
Кафедра: **АОИ, Кафедра автоматизации обработки информации**
Курс: **1**
Семестр: **1, 2**
Учебный план набора 2014 года

Распределение рабочего времени

№	Виды учебной деятельности	1 семестр	2 семестр	Всего	Единицы
1	Лекции	2	4	6	часов
2	Практические занятия	4	0	4	часов
3	Лабораторные работы	4	12	16	часов
4	Всего аудиторных занятий	10	16	26	часов
5	Самостоятельная работа	62	16	78	часов
6	Всего (без экзамена)	72	32	104	часов
7	Подготовка и сдача зачета	0	4	4	часов
8	Общая трудоемкость	72	36	108	часов
				3.0	З.Е.

Контрольные работы: 2 семестр - 1
Зачет: 2 семестр

Томск 2018

Рассмотрена и одобрена на заседании кафедры
протокол № 318 от « 3 » 7 2018 г.

ЛИСТ СОГЛАСОВАНИЯ

Рабочая программа дисциплины составлена с учетом требований федерального государственного образовательного стандарта высшего образования (ФГОС ВО) по направлению подготовки (специальности) 09.03.04 Программная инженерия, утвержденного 12.03.2015 года, рассмотрена и одобрена на заседании кафедры АОИ «___» _____ 20__ года, протокол №_____.

Разработчик:

старший преподаватель каф. АОИ _____ Н. В. Пермякова

Заведующий обеспечивающей каф. АОИ _____ Ю. П. Ехлаков

Рабочая программа дисциплины согласована с факультетом и выпускающей кафедрой:

Декан ЗиВФ _____ И. В. Осипов

Заведующий выпускающей каф. АОИ _____ Ю. П. Ехлаков

Эксперты:

Доцент кафедры автоматизации обработки информации (АОИ) _____ Н. Ю. Салмина

Доцент кафедры автоматизации обработки информации (АОИ) _____ А. А. Сидоров

1. Цели и задачи дисциплины

1.1. Цели дисциплины

Изучение классических алгоритмов сортировки и поиска. Ознакомление с различными способами хранения и представления данных.

1.2. Задачи дисциплины

– Формирование у студента знаний классических алгоритмов и их характеристик;

– формирование знаний существующих способов представления, хранения и обработки данных;

– получение студентами навыков реализации классических алгоритмов на языке программирования высокого уровня;

– формирование навыков владения языками структурного программирования, отладки и тестирования программ.

2. Место дисциплины в структуре ОПОП

Дисциплина «Алгоритмы и структуры данных» (Б1.В.ОД.9) относится к блоку 1 (вариативная часть).

Предшествующими дисциплинами, формирующими начальные знания, являются: Дискретная математика, Информатика и программирование.

Последующими дисциплинами являются: Компьютерная графика, Научно-исследовательская работа (рассред.), Практика по получению первичных профессиональных умений и навыков, в том числе первичных умений и навыков научно-исследовательской деятельности.

3. Требования к результатам освоения дисциплины

Процесс изучения дисциплины направлен на формирование следующих компетенций:

– ОПК-1 владением основными концепциями, принципами, теориями и фактами, связанными с информатикой;

– ПК-1 готовностью применять основные методы и инструменты разработки программного обеспечения;

В результате изучения дисциплины обучающийся должен:

– **знать** простые алгоритмы сортировки; улучшенные алгоритмы сортировки; алгоритмы поиска подстроки в строке; алгоритмы на BST-деревьях; различные представления очередей с приоритетом;

– **уметь** осуществлять операции сортировки данных; осуществлять поиск данных по заданному ключу; определять вычислительную сложность алгоритмов;

– **владеть** навыками реализации и отладки программ на алгоритмических языках программирования; использования различных структур данных при решении задач.

4. Объем дисциплины и виды учебной работы

Общая трудоемкость дисциплины составляет 3.0 зачетных единицы и представлена в таблице 4.1.

Таблица 4.1 – Трудоемкость дисциплины

Виды учебной деятельности	Всего часов	Семестры	
		1 семестр	2 семестр
Аудиторные занятия (всего)	26	10	16
Лекции	6	2	4
Практические занятия	4	4	0
Лабораторные работы	16	4	12
Самостоятельная работа (всего)	78	62	16
Оформление отчетов по лабораторным работам	4	2	2

Подготовка к лабораторным работам	4	2	2
Проработка лекционного материала	4	2	2
Самостоятельное изучение тем (вопросов) теоретической части курса	59	54	5
Подготовка к практическим занятиям, семинарам	2	2	0
Выполнение контрольных работ	5	0	5
Всего (без экзамена)	104	72	32
Подготовка и сдача зачета	4	0	4
Общая трудоемкость, ч	108	72	36
Зачетные Единицы	3.0		

5. Содержание дисциплины

5.1. Разделы дисциплины и виды занятий

Разделы дисциплины и виды занятий приведены в таблице 5.1.

Таблица 5.1 – Разделы дисциплины и виды занятий

Названия разделов дисциплины	Лек., ч	Прак. зан., ч	Лаб. раб., ч	Сам. раб., ч	Всего часов (без экзамена)	Формируемые компетенции
1 семестр						
1 Сортировка	2	0	4	34	40	ОПК-1, ПК-1
2 Поиск	0	4	0	28	32	ОПК-1, ПК-1
Итого за семестр	2	4	4	62	72	
2 семестр						
3 Анализ эффективности алгоритмов	0	0	0	10	10	ОПК-1, ПК-1
4 Двоичные деревья	4	0	12	6	22	ОПК-1, ПК-1
Итого за семестр	4	0	12	16	32	
Итого	6	4	16	78	104	

5.2. Содержание разделов дисциплины (по лекциям)

Содержание разделов дисциплин (по лекциям) приведено в таблице 5.2.

Таблица 5.2 – Содержание разделов дисциплин (по лекциям)

Названия разделов	Содержание разделов дисциплины (по лекциям)	Трудоемкость, ч	Формируемые компетенции
1 семестр			
1 Сортировка	Определения и свойства алгоритмов. Виды алгоритмов. Понятие эффективности и временной сложности. Понятие сортировки элементов. Простые сортировки. Сортировка выбором,	2	ОПК-1

	сортировка вставками, сортировка обменом. Анализ эффективности сортировок. Понятие устойчивости и естественности сортировок.		
	Итого	2	
Итого за семестр		2	
2 семестр			
4 Двоичные деревья	Двоичные деревья (создание деревьев, обходы деревьев, фундаментальные операции над деревьями, поиск элемента в дереве, разделение дерева, удаление элемента из дерева, объединение двух деревьев, балансировка деревьев, AVL-деревья)	4	ОПК-1
	Итого	4	
Итого за семестр		4	
Итого		6	

5.3. Разделы дисциплины и междисциплинарные связи с обеспечивающими (предыдущими) и обеспечиваемыми (последующими) дисциплинами

Разделы дисциплины и междисциплинарные связи с обеспечивающими (предыдущими) и обеспечиваемыми (последующими) дисциплинами представлены в таблице 5.3.

Таблица 5.3 – Разделы дисциплины и междисциплинарные связи

Наименование дисциплин	№ разделов данной дисциплины, для которых необходимо изучение обеспечивающих и обеспечиваемых дисциплин			
	1	2	3	4
Предшествующие дисциплины				
1 Дискретная математика				+
2 Информатика и программирование	+	+	+	+
Последующие дисциплины				
1 Компьютерная графика	+		+	+
2 Научно-исследовательская работа (рассред.)	+	+	+	+
3 Практика по получению первичных профессиональных умений и навыков, в том числе первичных умений и навыков научно-исследовательской деятельности	+	+	+	+

5.4. Соответствие компетенций, формируемых при изучении дисциплины, и видов занятий

Соответствие компетенций, формируемых при изучении дисциплины, и видов занятий представлено в таблице 5.4.

Таблица 5.4 – Соответствие компетенций, формируемых при изучении дисциплины, и видов занятий

Компетенции	Виды занятий				Формы контроля
	Лек.	Прак. зан.	Лаб. раб.	Сам. раб.	

ОПК-1	+	+	+	+	Контрольная работа, Отчет по лабораторной работе, Опрос на занятиях, Зачет, Тест, Отчет по практическому занятию
ПК-1		+	+	+	Контрольная работа, Отчет по лабораторной работе, Опрос на занятиях, Зачет, Тест

6. Интерактивные методы и формы организации обучения

Не предусмотрено РУП.

7. Лабораторные работы

Наименование лабораторных работ приведено в таблице 7.1.

Таблица 7.1 – Наименование лабораторных работ

Названия разделов	Наименование лабораторных работ	Трудоемкость, ч	Формируемые компетенции
1 семестр			
1 Сортировка	Простые сортировки на месте	4	ОПК-1, ПК-1
	Итого	4	
Итого за семестр		4	
2 семестр			
4 Двоичные деревья	Двоичные деревья - 1	6	ОПК-1, ПК-1
	Двоичные деревья - 2	6	
	Итого	12	
Итого за семестр		12	
Итого		16	

8. Практические занятия (семинары)

Наименование практических занятий (семинаров) приведено в таблице 8.1.

Таблица 8.1 – Наименование практических занятий (семинаров)

Названия разделов	Наименование практических занятий (семинаров)	Трудоемкость, ч	Формируемые компетенции
1 семестр			
2 Поиск	Поиск	4	ОПК-1, ПК-1
	Итого	4	
Итого за семестр		4	
Итого		4	

9. Самостоятельная работа

Виды самостоятельной работы, трудоемкость и формируемые компетенции представлены в таблице 9.1.

Таблица 9.1 – Виды самостоятельной работы, трудоемкость и формируемые компетенции

Названия разделов	Виды самостоятельной работы	Трудоемкость, ч	Формируемые компетенции	Формы контроля
1 семестр				
1 Сортировка	Самостоятельное изучение тем (вопросов) теоретической части курса	28	ОПК-1, ПК-1	Зачет, Отчет по лабораторной работе, Тест
	Проработка лекционного материала	2		
	Подготовка к лабораторным работам	2		
	Оформление отчетов по лабораторным работам	2		
	Итого	34		
2 Поиск	Подготовка к практическим занятиям, семинарам	2	ОПК-1, ПК-1	Зачет, Опрос на занятиях, Отчет по практическому занятию, Тест
	Самостоятельное изучение тем (вопросов) теоретической части курса	26		
	Итого	28		
Итого за семестр		62		
2 семестр				
3 Анализ эффективности алгоритмов	Выполнение контрольных работ	5	ОПК-1, ПК-1	Контрольная работа, Тест
	Самостоятельное изучение тем (вопросов) теоретической части курса	5		
	Итого	10		
4 Двоичные деревья	Проработка лекционного материала	2	ОПК-1, ПК-1	Зачет, Отчет по лабораторной работе, Тест
	Подготовка к лабораторным работам	2		
	Оформление отчетов по лабораторным работам	2		
	Итого	6		

Итого за семестр		16		
	Подготовка и сдача зачета	4		Зачет
Итого		82		

10. Курсовой проект / курсовая работа

Не предусмотрено РУП.

11. Рейтинговая система для оценки успеваемости обучающихся

Рейтинговая система не используется.

12. Учебно-методическое и информационное обеспечение дисциплины

12.1. Основная литература

1. Вирт, Н. Алгоритмы и структуры данных. Новая версия для Оберона [Электронный ресурс] [Электронный ресурс]: учебное пособие / Н. Вирт. — Электрон. дан. — Москва ДМК Пресс, 2010. — 272 с. - Режим доступа: https://e.lanbook.com/book/1261 (дата обращения: 25.07.2018).

2. Тюкачев, Н.А. C#. Алгоритмы и структуры данных [Электронный ресурс] [Электронный ресурс]: учебное пособие / Н.А. Тюкачев, В.Г. Хлебостроев. — Электрон. дан. — Санкт-Петербург Лань, 2018. — 232 с. - Режим доступа: https://e.lanbook.com/book/104961 (дата обращения: 25.07.2018).

12.2. Дополнительная литература

1. Информатика и программирование [Электронный ресурс]: Учебное пособие / Н. В. Пермякова - 2016. 188 с. - Режим доступа: https://edu.tusur.ru/publications/7678 (дата обращения: 25.07.2018).

2. Потопахин, В. Искусство алгоритмизации [Электронный ресурс] / В. Потопахин. — Электрон. дан. — Москва [Электронный ресурс]: ДМК Пресс, 2011. — 320 с. - Режим доступа: https://e.lanbook.com/book/1269 (дата обращения: 25.07.2018).

12.3. Учебно-методические пособия

12.3.1. Обязательные учебно-методические пособия

1. Алгоритмы и структуры данных [Электронный ресурс]: Методические указания к лабораторным работам, практическим занятиям и организации самостоятельной работы / Н. В. Пермякова - 2018. 26 с. - Режим доступа: https://edu.tusur.ru/publications/8390 (дата обращения: 25.07.2018).

12.3.2. Учебно-методические пособия для лиц с ограниченными возможностями здоровья и инвалидов

Учебно-методические материалы для самостоятельной и аудиторной работы обучающихся из числа лиц с ограниченными возможностями здоровья и инвалидов предоставляются в формах, адаптированных к ограничениям их здоровья и восприятия информации.

Для лиц с нарушениями зрения:

- в форме электронного документа;
- в печатной форме увеличенным шрифтом.

Для лиц с нарушениями слуха:

- в форме электронного документа;
- в печатной форме.

Для лиц с нарушениями опорно-двигательного аппарата:

- в форме электронного документа;
- в печатной форме.

12.4. Профессиональные базы данных и информационные справочные системы

1. Дополнительно к профессиональным базам данных рекомендуется использовать информационные, справочные и нормативные базы данных https://lib.tusur.ru/ru/resursy/bazy-dannyh

13. Материально-техническое обеспечение дисциплины и требуемое программное

обеспечение

13.1. Общие требования к материально-техническому и программному обеспечению дисциплины

13.1.1. Материально-техническое и программное обеспечение для лекционных занятий

Для проведения занятий лекционного типа, групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации используется учебная аудитория с количеством посадочных мест не менее 22-24, оборудованная доской и стандартной учебной мебелью. Имеются демонстрационное оборудование и учебно-наглядные пособия, обеспечивающие тематические иллюстрации по лекционным разделам дисциплины.

13.1.2. Материально-техническое и программное обеспечение для практических занятий

Лаборатория «Информатика и программирование»

учебная аудитория для проведения занятий практического типа, учебная аудитория для проведения занятий лабораторного типа, помещение для курсового проектирования (выполнения курсовых работ), помещение для самостоятельной работы

634034, Томская область, г. Томск, Вершинина улица, д. 74, 428 ауд.

Описание имеющегося оборудования:

- Персональный компьютер Intel Core 2 Duo E6550 2.3 ГГц, ОЗУ – 2 Гб, жесткий диск – 250 Гб (14 шт.);
- Меловая доска;
- Комплект специализированной учебной мебели;
- Рабочее место преподавателя.

Программное обеспечение:

- Dev-Cpp
- EclEmma, Eclipse Public License v1.0
- Eclipse Oxygen, Eclipce PLv2->GNU GPLv2
- GCC, GNU GPLv3
- Google Chrome
- Java SE Development Kit
- Lazarus
- LibreOffice
- Microsoft Visio 2010
- Microsoft Visual Studio 2015
- Microsoft Windows 7 Pro
- PDF-XChange Viewer
- Архиватор7z 16.04, GNU LGPL

Лаборатория «Муниципальная информатика»

учебная аудитория для проведения занятий практического типа, учебная аудитория для проведения занятий лабораторного типа, помещение для курсового проектирования (выполнения курсовых работ), помещение для самостоятельной работы

634034, Томская область, г. Томск, Вершинина улица, д. 74, 432б ауд.

Описание имеющегося оборудования:

- Персональный компьютер Intel Core i5-2320 3.0 ГГц, ОЗУ – 4 Гб, жесткий диск – 500 Гб (12 шт.);
- Магнитно-маркерная доска;
- Комплект специализированной учебной мебели;
- Рабочее место преподавателя.

Программное обеспечение:

- Dev-Cpp
- EclEmma, Eclipse Public License v1.0
- Eclipse Oxygen, Eclipce PLv2->GNU GPLv2

– GCC, GNU GPLv3
– Google Chrome
– IntelliJ Community, Apache 2.0 license
– Java SE Development Kit
– Lazarus
– LibreOffice
– Microsoft Visio 2010
– Microsoft Visual Studio 2015
– Microsoft Windows 10 Pro
– PDF-XChange Viewer
– Архиватор7z 16.04, GNU LGPL

Лаборатория «Распределенные вычислительные системы»

учебная аудитория для проведения занятий практического типа, учебная аудитория для проведения занятий лабораторного типа, помещение для курсового проектирования (выполнения курсовых работ), помещение для самостоятельной работы

634034, Томская область, г. Томск, Вершинина улица, д. 74, 432а ауд.

Описание имеющегося оборудования:

- Персональный компьютер Intel Core i5-3330 3.0 ГГц, ОЗУ – 4 Гб, жесткий диск – 500 Гб (12 шт.);
- Меловая доска;
- Комплект специализированной учебной мебели;
- Рабочее место преподавателя.

Программное обеспечение:

– Dev-Cpp
– EclEmma, Eclipse Public License v1.0
– Eclipse Oxygen, Eclipce PLv2->GNU GPLv2
– GCC, GNU GPLv3
– Google Chrome
– IntelliJ Community, Apache 2.0 license
– Java SE Development Kit
– Lazarus
– LibreOffice
– Microsoft Visio 2010
– Microsoft Visual Studio 2015
– Microsoft Windows 10 Pro
– PDF-XChange Viewer
– Архиватор7z 16.04, GNU LGPL

Лаборатория «Операционные системы и СУБД»

учебная аудитория для проведения занятий практического типа, учебная аудитория для проведения занятий лабораторного типа, помещение для курсового проектирования (выполнения курсовых работ), помещение для самостоятельной работы

634034, Томская область, г. Томск, Вершинина улица, д. 74, 430 ауд.

Описание имеющегося оборудования:

- Персональный компьютер Intel Core 2 Duo E6550 2.3 ГГц, ОЗУ – 2 Гб, жесткий диск – 250 Гб (12 шт.);
- Магнитно-маркерная доска;
- Комплект специализированной учебной мебели;
- Рабочее место преподавателя.

Программное обеспечение:

– Dev-Cpp
– EclEmma, Eclipse Public License v1.0

- Eclipse Oxygen, Eclipce PLv2->GNU GPLv2
- GCC, GNU GPLv3
- Google Chrome
- IntelliJ Community, Apache 2.0 license
- Java SE Development Kit
- Lazarus
- LibreOffice
- Microsoft Visio 2010
- Microsoft Visual Studio 2015
- Microsoft Windows 7 Pro
- PDF-XChange Viewer
- Архиватор7z 16.04, GNU LGPL

Лаборатория «Информатика и программирование»

учебная аудитория для проведения занятий практического типа, учебная аудитория для проведения занятий лабораторного типа, помещение для курсового проектирования (выполнения курсовых работ), помещение для самостоятельной работы

634034, Томская область, г. Томск, Вершинина улица, д. 74, 428 ауд.

Описание имеющегося оборудования:

- Персональный компьютер Intel Core 2 Duo E6550 2.3 ГГц, ОЗУ – 2 Гб, жесткий диск – 250 Гб (14 шт.);
- Меловая доска;
- Комплект специализированной учебной мебели;
- Рабочее место преподавателя.

Программное обеспечение:

- Dev-Cpp
- EclEmma, Eclipse Public License v1.0
- Eclipse Oxygen, Eclipce PLv2->GNU GPLv2
- GCC, GNU GPLv3
- Google Chrome
- IntelliJ Community, Apache 2.0 license
- Java SE Development Kit
- Lazarus
- LibreOffice
- Microsoft Visio 2010
- Microsoft Visual Studio 2015
- Microsoft Windows 7 Pro
- Архиватор7z 16.04, GNU LGPL

Лаборатория «Программная инженерия»

учебная аудитория для проведения занятий практического типа, учебная аудитория для проведения занятий лабораторного типа, помещение для курсового проектирования (выполнения курсовых работ), помещение для самостоятельной работы

634034, Томская область, г. Томск, Вершинина улица, д. 74, 409 ауд.

Описание имеющегося оборудования:

- Персональный компьютер Intel Core i3-6300 3.2 ГГц, ОЗУ – 8 Гб, жесткий диск – 500 Гб (10 шт.);
- Проектор Optoma Ex632.DLP;
- Экран для проектора Lumian Mas+Er;
- Магнитно-маркерная доска;
- Комплект специализированной учебной мебели;
- Рабочее место преподавателя.

Программное обеспечение:

- Dev-Cpp
- EclEmma, Eclipse Public License v1.0
- Eclipse Oxygen, Eclipce PLv2->GNU GPLv2
- GCC, GNU GPLv3
- Google Chrome
- Java SE Development Kit
- Lazarus
- LibreOffice
- MS Visual Studio 2015, MS Imagine Premium
- Microsoft Visio 2010
- Microsoft Visual Studio 2015
- Microsoft Windows 10
- PDF-XChange Viewer
- Архиватор7z 16.04, GNU LGPL

Лаборатория «Бизнес-информатика»

учебная аудитория для проведения занятий практического типа, учебная аудитория для проведения занятий лабораторного типа, помещение для курсового проектирования (выполнения курсовых работ), помещение для самостоятельной работы

634034, Томская область, г. Томск, Вершинина улица, д. 74, 407 ауд.

Описание имеющегося оборудования:

- Персональный компьютер Intel Core i5-2320 3.0 ГГц, ОЗУ – 4 Гб, жесткий диск – 500 Гб (12 шт.);
- Проектор Optoma Ex632.DLP;
- Экран для проектора Lumian Mas+Er;
- Магнитно-маркерная доска;
- Комплект специализированной учебной мебели;
- Рабочее место преподавателя.

Программное обеспечение:

- Dev-Cpp
- EclEmma, Eclipse Public License v1.0
- Eclipse Oxygen, Eclipce PLv2->GNU GPLv2
- GCC, GNU GPLv3
- Google Chrome
- Java SE Development Kit
- Lazarus
- LibreOffice
- Microsoft Visio 2010
- Microsoft Visual Studio 2015
- Microsoft Windows 10
- PDF-XChange Viewer
- Visio
- Visual Studio

13.1.3. Материально-техническое и программное обеспечение для лабораторных работ

Лаборатория «Муниципальная информатика»

учебная аудитория для проведения занятий практического типа, учебная аудитория для проведения занятий лабораторного типа, помещение для курсового проектирования (выполнения курсовых работ), помещение для самостоятельной работы

634034, Томская область, г. Томск, Вершинина улица, д. 74, 432б ауд.

Описание имеющегося оборудования:

- Персональный компьютер Intel Core i5-2320 3.0 ГГц, ОЗУ – 4 Гб, жесткий диск – 500 Гб

(12 шт.);

- Магнитно-маркерная доска;
- Комплект специализированной учебной мебели;
- Рабочее место преподавателя.

Программное обеспечение:

- Dev-Cpp
- EclEmma, Eclipse Public License v1.0
- Eclipse Oxygen, Eclipce PLv2->GNU GPLv2
- GCC, GNU GPLv3
- Google Chrome
- IntelliJ Community, Apache 2.0 license
- Java SE Development Kit
- Lazarus
- LibreOffice
- Microsoft Visio 2010
- Microsoft Visual Studio 2015
- Microsoft Windows 10 Pro
- PDF-XChange Viewer
- Архиватор7z 16.04, GNU LGPL

Лаборатория «Распределенные вычислительные системы»

учебная аудитория для проведения занятий практического типа, учебная аудитория для проведения занятий лабораторного типа, помещение для курсового проектирования (выполнения курсовых работ), помещение для самостоятельной работы

634034, Томская область, г. Томск, Вершинина улица, д. 74, 432а ауд.

Описание имеющегося оборудования:

- Персональный компьютер Intel Core i5-3330 3.0 ГГц, ОЗУ – 4 Гб, жесткий диск – 500 Гб (12 шт.);
- Меловая доска;
- Комплект специализированной учебной мебели;
- Рабочее место преподавателя.

Программное обеспечение:

- Dev-Cpp
- EclEmma, Eclipse Public License v1.0
- Eclipse Oxygen, Eclipce PLv2->GNU GPLv2
- GCC, GNU GPLv3
- Google Chrome
- IntelliJ Community, Apache 2.0 license
- Java SE Development Kit
- Lazarus
- LibreOffice
- Microsoft Visio 2010
- Microsoft Visual Studio 2015
- Microsoft Windows 10 Pro
- PDF-XChange Viewer
- Архиватор7z 16.04, GNU LGPL

Лаборатория «Операционные системы и СУБД»

учебная аудитория для проведения занятий практического типа, учебная аудитория для проведения занятий лабораторного типа, помещение для курсового проектирования (выполнения курсовых работ), помещение для самостоятельной работы

634034, Томская область, г. Томск, Вершинина улица, д. 74, 430 ауд.

Описание имеющегося оборудования:

- Персональный компьютер Intel Core 2 Duo E6550 2.3 ГГц, ОЗУ – 2 Гб, жесткий диск – 250 Гб (12 шт.);
- Магнитно-маркерная доска;
- Комплект специализированной учебной мебели;
- Рабочее место преподавателя.

Программное обеспечение:

- Dev-Cpp
- EclEmma, Eclipse Public License v1.0
- Eclipse Oxygen, Eclipce PLv2->GNU GPLv2
- GCC, GNU GPLv3
- Google Chrome
- IntelliJ Community, Apache 2.0 license
- Java SE Development Kit
- Lazarus
- LibreOffice
- Microsoft Visio 2010
- Microsoft Visual Studio 2015
- Microsoft Windows 7 Pro
- PDF-XChange Viewer
- Архиватор7z 16.04, GNU LGPL

Лаборатория «Программная инженерия»

учебная аудитория для проведения занятий практического типа, учебная аудитория для проведения занятий лабораторного типа, помещение для курсового проектирования (выполнения курсовых работ), помещение для самостоятельной работы

634034, Томская область, г. Томск, Вершинина улица, д. 74, 409 ауд.

Описание имеющегося оборудования:

- Персональный компьютер Intel Core i3-6300 3.2 ГГц, ОЗУ – 8 Гб, жесткий диск – 500 Гб (10 шт.);
- Проектор Optoma Ex632.DLP;
- Экран для проектора Lumian Mas+Er;
- Магнитно-маркерная доска;
- Комплект специализированной учебной мебели;
- Рабочее место преподавателя.

Программное обеспечение:

- Dev-Cpp
- EclEmma, Eclipse Public License v1.0
- Eclipse Oxygen, Eclipce PLv2->GNU GPLv2
- GCC, GNU GPLv3
- Google Chrome
- IntelliJ Community, Apache 2.0 license
- Java SE Development Kit
- Lazarus
- LibreOffice
- MS Visio 2010, MS Imagine Premium
- MS Visual Studio 2015, MS Imagine Premium
- Microsoft Visio 2010
- Microsoft Visual Studio 2015
- Microsoft Windows 10
- PDF-XChange Viewer
- Архиватор7z 16.04, GNU LGPL

Лаборатория «Бизнес-информатика»

учебная аудитория для проведения занятий практического типа, учебная аудитория для проведения занятий лабораторного типа, помещение для курсового проектирования (выполнения курсовых работ), помещение для самостоятельной работы

634034, Томская область, г. Томск, Вершинина улица, д. 74, 407 ауд.

Описание имеющегося оборудования:

- Персональный компьютер Intel Core i5-2320 3.0 ГГц, ОЗУ – 4 Гб, жесткий диск – 500 Гб (12 шт.);
- Проектор Optoma Ex632.DLP;
- Экран для проектора Lumian Mas+Er;
- Магнитно-маркерная доска;
- Комплект специализированной учебной мебели;
- Рабочее место преподавателя.

Программное обеспечение:

- Dev-Cpp
- EclEmma, Eclipse Public License v1.0
- Eclipse Oxygen, Eclipce PLv2->GNU GPLv2
- GCC, GNU GPLv3
- Google Chrome
- IntelliJ Community, Apache 2.0 license
- Java SE Development Kit
- Lazarus
- LibreOffice
- Microsoft Visio 2010
- Microsoft Visual Studio 2015
- Microsoft Windows 10
- PDF-XChange Viewer
- Visual Studio
- Архиватор7z 16.04, GNU LGPL

13.1.4. Материально-техническое и программное обеспечение для самостоятельной работы

Для самостоятельной работы используются учебные аудитории (компьютерные классы), расположенные по адресам:

- 634050, Томская область, г. Томск, Ленина проспект, д. 40, 233 ауд.;
- 634045, Томская область, г. Томск, ул. Красноармейская, д. 146, 201 ауд.;
- 634034, Томская область, г. Томск, Вершинина улица, д. 47, 126 ауд.;
- 634034, Томская область, г. Томск, Вершинина улица, д. 74, 207 ауд.

Состав оборудования:

- учебная мебель;
- компьютеры класса не ниже ПЭВМ INTEL Celeron D336 2.8ГГц. - 5 шт.;
- компьютеры подключены к сети «Интернет» и обеспечивают доступ в электронную информационно-образовательную среду университета.

Перечень программного обеспечения:

- Microsoft Windows;
- OpenOffice;
- Kaspersky Endpoint Security 10 для Windows;
- 7-Zip;
- Google Chrome.

13.2. Материально-техническое обеспечение дисциплины для лиц с ограниченными возможностями здоровья и инвалидов

Освоение дисциплины лицами с ограниченными возможностями здоровья и инвалидами осуществляется с использованием средств обучения общего и специального назначения.

При занятиях с обучающимися **с нарушениями слуха** предусмотрено использование звукоусиливающей аппаратуры, мультимедийных средств и других технических средств приема/передачи учебной информации в доступных формах, мобильной системы преподавания для обучающихся с инвалидностью, портативной индукционной системы. Учебная аудитория, в которой занимаются обучающиеся с нарушением слуха, оборудована компьютерной техникой, аудиотехникой, видеотехникой, электронной доской, мультимедийной системой.

При занятиях с обучающимися **с нарушениями зрениями** предусмотрено использование в лекционных и учебных аудиториях возможности просмотра удаленных объектов (например, текста на доске или слайда на экране) при помощи видеоувеличителей для комфортного просмотра.

При занятиях с обучающимися **с нарушениями опорно-двигательного аппарата** используются альтернативные устройства ввода информации и другие технические средства приема/передачи учебной информации в доступных формах, мобильной системы обучения для людей с инвалидностью.

14. Оценочные материалы и методические рекомендации по организации изучения дисциплины

14.1. Содержание оценочных материалов и методические рекомендации

Для оценки степени сформированности и уровня освоения закрепленных за дисциплиной компетенций используются оценочные материалы в составе:

14.1.1. Тестовые задания

ОПК – 1. Владением основными концепциями, принципами, теориями и фактами, связанными с информатикой.

Вопрос 1

Данные, обрабатываемые в алгоритме, могут быть представлены в виде массива. Сортировкой элементов массива будем называть алгоритм, который изменяет порядок элементов массива по следующему правилу: $a_i \leq a_{i+1}$. Какой из алгоритмов сортировки сравнивает два рядом стоящих элемента и меняет их местами, если первый элемент пары больше, чем второй?

сортировка вставками
сортировка выбором
сортировка бинарными вставками
сортировка обменом

Вопрос 2

Данные, обрабатываемые в алгоритме, могут быть представлены в виде массива. Сортировкой элементов массива будем называть алгоритм, который изменяет порядок элементов массива по следующему правилу: $a_i \leq a_{i+1}$. Какой из алгоритмов сортировки ищет минимальный элемент массива и меняет его местами с первым элементом, затем ищет минимальный элемент среди оставшихся, и меняет его местами со вторым элементом и так далее, пока не будет отсортирована вся последовательность?

сортировка вставками
сортировка выбором
сортировка бинарными вставками
сортировка обменом

Вопрос 3

Данные, обрабатываемые в алгоритме, могут быть представлены в виде массива. Сортировкой элементов массива будем называть алгоритм, который изменяет порядок элементов массива по следующему правилу: $a_i \leq a_{i+1}$.Какое из представленных утверждений верно?

сортировка вставками устойчива
сортировка обменом не устойчива

сортировка выбором устойчива
сортировка вставками – неестественная сортировка

Вопрос 4

Временная сложность алгоритма оценивается количеством выполняемых алгоритмом элементарных операций. Какова оценка временной сложности алгоритма, представленного ниже?

```
int n;
scanf(“%d”,&n);
for(float j= 0;j<=n;j+=0.25)
printf("%5.1f\n",j);
```

O(1)
O(n)
O(n2)
O(n ·log(n))

Вопрос 5

Временная сложность алгоритма оценивается количеством выполняемых алгоритмом элементарных операций. Какова оценка временной сложности алгоритма, представленного ниже?

```
int n, x[100];
scanf(“%d”,&n);
for(int j= 0;j<n;j++)
for(int i= 0;i<n;i++)
x[j]=i*j;
```

O(1)
O(n)
O(n2)
O(n ·log(n))

Вопрос 6

Данные, обрабатываемые в алгоритме, могут быть представлены в виде массива. Сортировкой элементов массива будем называть алгоритм, который изменяет порядок элементов массива по следующему правилу: $a_i \leq a_{i+1}$. Сортировка пирамидой (HeapSort) на первом этапе алгоритма строит пирамидально упорядоченный массив. В исходном массиве элементы располагались следующим образом: 1 6 2 4 1 7 9 3

Каким образом будут располагаться элементы массива после выполнения первого этапа HeapSort?

1 6 2 4 1 7 9 3
1 1 2 3 4 6 7 9
3 2 1 1 4 7 6 9
9 6 7 4 1 1 2 3

Вопрос 7

Данные, обрабатываемые в алгоритме, могут быть представлены в виде массива. Сортировкой элементов массива будем называть алгоритм, который изменяет порядок элементов массива по следующему правилу: $a_i \leq a_{i+1}$. Сортировка Шелла сравнивает элементы массива, отстоящие друг от друга на заданный интервал (шаг). В исходном массиве элементы располагались следующим образом:

2 5 7 8 1 9 2 4 6 1 5 4

Каким образом будут располагаться элементы массива после выполнения алгоритма с шагом 4?

1 1 2 4 2 5 5 4 6 9 7 8
1 1 2 2 4 4 5 5 6 7 8 9
1 2 1 4 2 5 4 5 6 9 7 8
1 5 7 8 2 9 2 4 6 1 5 4

Вопрос 8

Данные, обрабатываемые в алгоритме, могут быть представлены в виде массива. Сортировкой элементов массива будем называть алгоритм, который изменяет порядок элементов массива по следующему правилу: $a_i \leq a_{i+1}$. Сортировка расческой (combsort) сравнивает элементы массива, отстоящие друг от друга на заданный интервал (шаг). В исходном массиве элементы располагались следующим образом: 2 5 7 8 1 9 2 4 6 1 5 4

Каким образом будут располагаться элементы массива после выполнения алгоритма с шагом 4?

1 1 2 4 2 5 5 4 6 9 7 8
1 1 2 2 4 4 5 5 6 7 8 9
1 5 2 4 2 1 5 4 6 9 7 8
1 5 7 8 2 9 2 4 6 1 5 4

Вопрос 9

Данные, обрабатываемые в алгоритме, могут быть представлены в виде массива. Сортировкой элементов массива будем называть алгоритм, который изменяет порядок элементов массива по следующему правилу: $a_i \leq a_{i+1}$. Сортировка Хоара на каждой итерации алгоритма выбирает медианный элемент, а оставшиеся элементы массива переставляет следующим образом – все элементы, меньшие медианного записываются в левую часть массива, большие – в правую часть массива. В исходном массиве элементы располагались следующим образом:

2 5 7 8 1 9 2 4 6 1 5

Какое значение принимается в качестве медианного элемента, если реализована классическая версия алгоритма?

1
9
5
6

Вопрос 10

Данные, обрабатываемые в алгоритме, могут быть представлены в виде массива. Сортировкой элементов массива будем называть алгоритм, который изменяет порядок элементов массива по следующему правилу: $a_i \leq a_{i+1}$. Если элементы массива – сложные ключи, для которых не определена операция сравнения, то для сортировки таких массивов используют поразрядные сортировки. Сколько сравнений символов выполнит MSD сортировка на массиве?

Кран
Трап
Игра
Торт
Порт
Соль

6
11
10
8

Вопрос 11

Данные, обрабатываемые в алгоритме, могут быть представлены в виде массива. Сортировкой элементов массива будем называть алгоритм, который изменяет порядок элементов массива по следующему правилу: $a_i \leq a_{i+1}$. Если элементы массива – сложные ключи, для которых не определена операция сравнения, то для сортировки таких массивов используют поразрядные сортировки. Каким свойством должна обладать внутриразрядная сортировка при реализации LSD-сортировки?

устойчивость
естественность
топологичность
распределенность

Вопрос 12.

Данные, обрабатываемые в алгоритме, могут быть представлены в виде массива. Сортировкой элементов массива будем называть алгоритм, который изменяет порядок элементов массива по следующему правилу: $a_i \le a_{i+1}$. Если элементы массива – сложные ключи, для которых не определена операция сравнения, то для сортировки таких массивов используют поразрядные сортировки. Какая из перечисленных последовательностей представляет собой последовательность шагов выполнения двоичной быстрой сортировки, если сортировка применялась к массиву, элементы которого были расположены так, как показано ниже?

1010
0101
1101
0011
1011
0001

0101	0011	0001	
0011	0001	0011	
0001	0101	0101	
1010	1010	1010	
1101	1011	1011	
1011	1101	1101	
1010	0101	0001	0001
0101	1101	1010	0011
1101	0001	0011	0101
0011	1010	1011	1010
1011	0011	0101	1011
0001	1011	1101	1101
0001	0001		
0101	0011		
0011	0101		
1101	1010		
1011	1011		
1010	1101		
0011	0011	0001	0001
0001	0001	0011	0011
0101	0101	0101	0101
1011	1011	1011	1010
1101	1010	1010	1011
1010	1101	1101	1101

Вопрос 13

Данные, обрабатываемые в алгоритме, могут быть представлены в виде массива. Одной из задач информатики является задача поиска информации в массиве. Сколько сравнений выполнит алгоритм прямого поиска на массиве 1 1 2 2 2 3 3 3 4 4 5 6 6 6 7 7 8 8 8 9 9 9 если ищется первый элемент со значением 5?

10

22
11
1

Вопрос 14

Данные, обрабатываемые в алгоритме, могут быть представлены в виде массива. Одной из задач информатики является задача поиска информации в массиве. Сколько сравнений выполнит алгоритм бинарного поиска на массиве 1 1 2 2 2 3 3 3 4 4 5 6 6 6 7 7 8 8 8 9 9 9 если ищется первый элемент со значением 5? (Будем считать, что элементы массива нумеруются с единицы.)

2
3
4
1

Вопрос 15

Данные, обрабатываемые в алгоритме, могут быть представлены в виде массива. Одной из задач информатики является задача поиска информации в массиве. Сколько сравнений выполнит алгоритм интерполяционного поиска на массиве 1 1 2 2 2 3 3 3 4 4 5 6 6 6 7 7 8 8 8 9 9 9 если ищется первый элемент со значением 5? (Будем считать, что элементы массива нумеруются с единицы.)

2
1
3
5

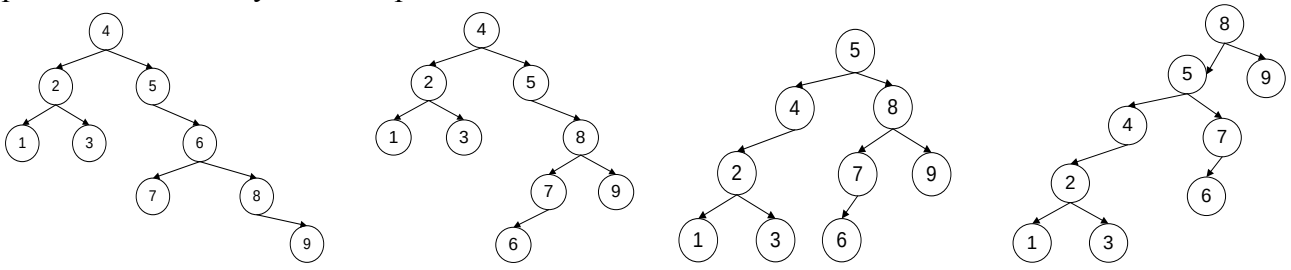
Вопрос 16

Данные, обрабатываемые в алгоритме, могут быть представлены в виде массива. Одной из задач информатики является задача поиска информации в массиве. Сколько сравнений выполнит алгоритм поиска, если элементы массива 4 2 5 1 6 8 7 3 9 представлены в виде BST-дерева, а ищется элемент со значением 3?

1
8
3
9

Вопрос 17

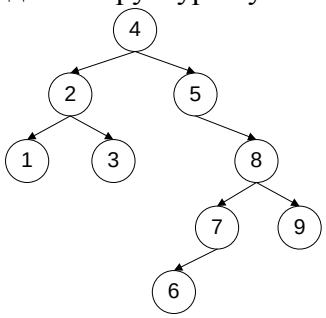
Одной из задач информатики является задача построения эффективных алгоритмов. Эффективность алгоритмов зависит и от способа представления данных, обрабатываемых в этих алгоритмах. Одним из способов представления данных является представление массивов в виде древовидных структур. Как будут располагаться элементы в BST-дереве, если элементы массива расположены следующим образом 4 2 5 1 6 8 7 3 9?



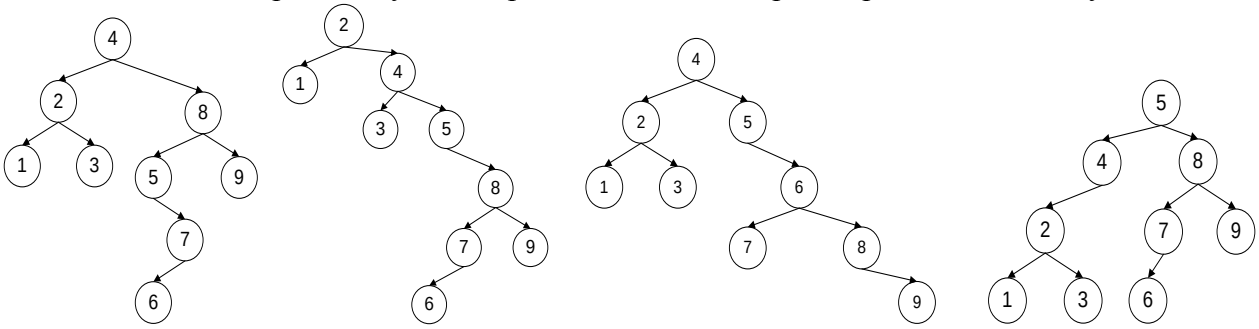
Вопрос 18

Одной из задач информатики является задача построения эффективных алгоритмов. Эффективность алгоритмов зависит и от способа представления данных, обрабатываемых в этих алгоритмах. Одним из способов представления данных является представление массивов в виде

древовидных структур. Пусть элементы массива расположены в BST-дереве следующим образом:



Какое BST-дерево получится при выполнении операции ротации влево в узле с ключом 4?



Вопрос 19

Задача поиска подстроки в строке относится к задаче алгоритмизации поиска. Алгоритм поиска подстроки, который описали Кнут, Моррис и Пратт разбит на два этапа, на первом этапе алгоритма строится таблица сдвигов подстроки. Какую таблицу сдвигов построит алгоритм для подстроки «АБРАКАДАБРА»?

{-1,0,0,0,0,0,1,1,1,2,3}
{-1,0,0,0,1,1,1,1,1,2,3}
{-1,0,0,0,1,0,1,0,1,2,3}
{-1,0,0,0,1,1,2,3,4,5,6}

Вопрос 20

Задача поиска подстроки в строке относится к задаче алгоритмизации поиска. Алгоритм поиска подстроки, который описал Боуэр-Мур разбит на два этапа и на первом этапе строится вспомогательный массив. Размерность этого массива равна размерности алфавита, которому принадлежат символы строки. Пусть подстрока поиска - «АБРАКАДАБРА». Какие значения массива соответствуют символам А, Б, Р, К и Д?

D[А] = 1 D[Б] = 2 D[Р] = 3 D[К] = 5 D[Д] = 7
D[А] = 10 D[Б] = 9 D[Р] = 8 D[К] = 5 D[Д] = 4
D[А] = 3 D[Б] = 2 D[Р] = 1 D[К] = 6 D[Д] = 4
D[А] = 5 D[Б] = 2 D[Р] = 2 D[К] = 1 D[Д] = 1

ПК-1.Готовностью применять основные методы и инструменты разработки программного обеспечения.

Вопрос 1

Следуя принципам метода нисходящего программирования, был разработан следующий алгоритм сортировки элементов массива X размерности n:

```
для  i  от 0 до n-1 нц
  f = 0
  для j от 0 до n-1-i нц
      если X[j]>X[j+1] то
            поменять местами
            X[j]и X[j+1]
            f = 1
```

```
  кц
  если f == 0 то закончить
            выполнение цикла
кц
```

Как называется этот алгоритм сортировки?

- сортировка обменом
- сортировка вставками
- сортировка выбором
- сортировка подсчетом

Вопрос 2

Следуя принципам метода модульного программирования, был разработан следующий алгоритм сортировки элементов массива X размерности n:

```
для  i  от 0 до n-1 нц
  k = i
  k = Min(X,i,n)
  если i != k то Change(X[i],X[k])
кц
```

Как называется этот алгоритм сортировки?

- сортировка обменом
- сортировка вставками
- сортировка выбором
- сортировка подсчетом

Вопрос 3

Следуя принципам структурного программирования, был разработан следующий алгоритм сортировки элементов массива X размерности n:

```
для  i  от 0 до n нц
  f = X[i]
  j = i-1
  пока f<X[j] and j>=0 нц
    X[j+1] =  X[j]
    j = j-1
  кц
  X[j+1] = f
кц
```

Как называется этот алгоритм сортировки?

- сортировка обменом
- сортировка вставками
- сортировка выбором
- сортировка подсчетом

Вопрос 4

Следуя принципам метода модульного программирования, был разработан следующий алгоритм сортировки элементов массива X размерности n:

```
m = Min(X,n)
m1 = Max(X,n)
k = m1-m+1
Определить массив P[k]
для  i  от 0 до k нц P[i] = 0 кц
для  i  от 0 до n нц
  P[X[i]-min] = P[X[i]-min] + 1
кц
k = 0
```

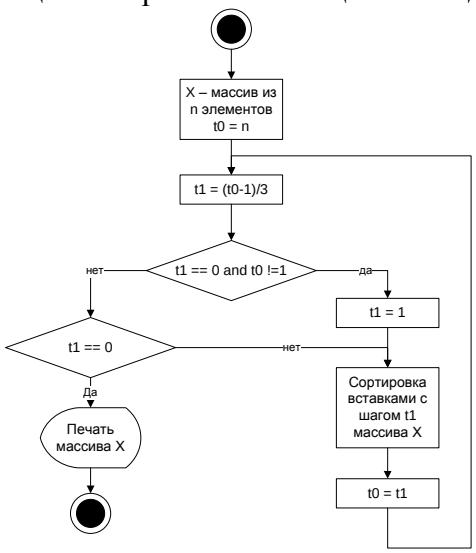
```
для  i  от 0 до k нц
  для  j от 0 до P[i] нц
       X[k]=i
       k = k+1
  кц
кц
```

Как называется этот алгоритм сортировки?

- сортировка обменом
- сортировка вставками
- сортировка выбором
- сортировка подсчетом

Вопрос 5

Метод нисходящего программирования требует пошаговой детализации алгоритма. Детализация алгоритма с помощью блок-диаграммы выполнена следующим образом:

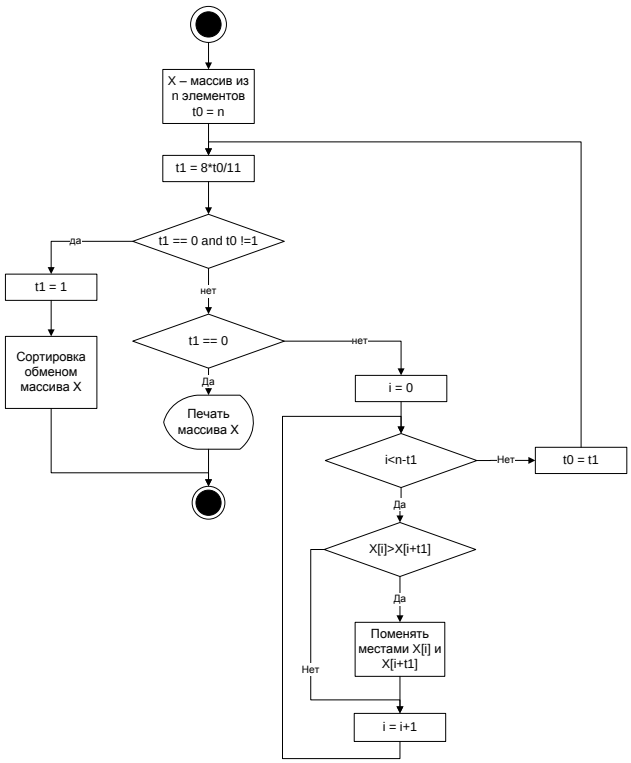


Сколько обменов элементов выполнит алгоритм с t1 = 4 на массиве
2 5 7 8 1 9 2 4 6 1 5 4?

- 13
- 14
- 15
- 16

Вопрос 6

Метод нисходящего программирования требует пошаговой детализации алгоритма. Детализация алгоритма с помощью блок-диаграммы выполнена следующим образом:

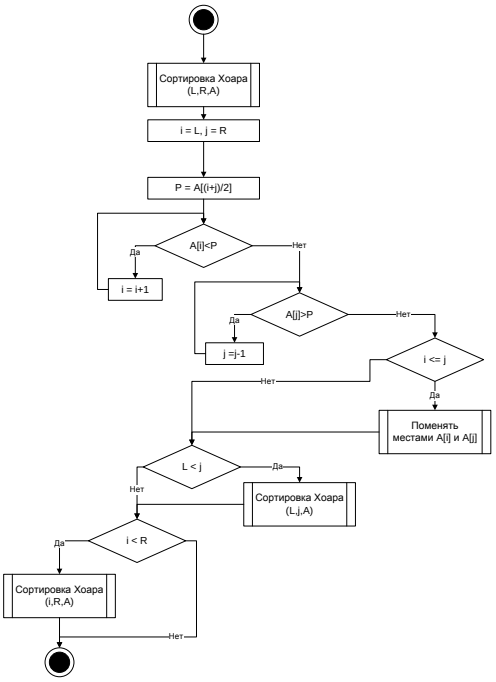


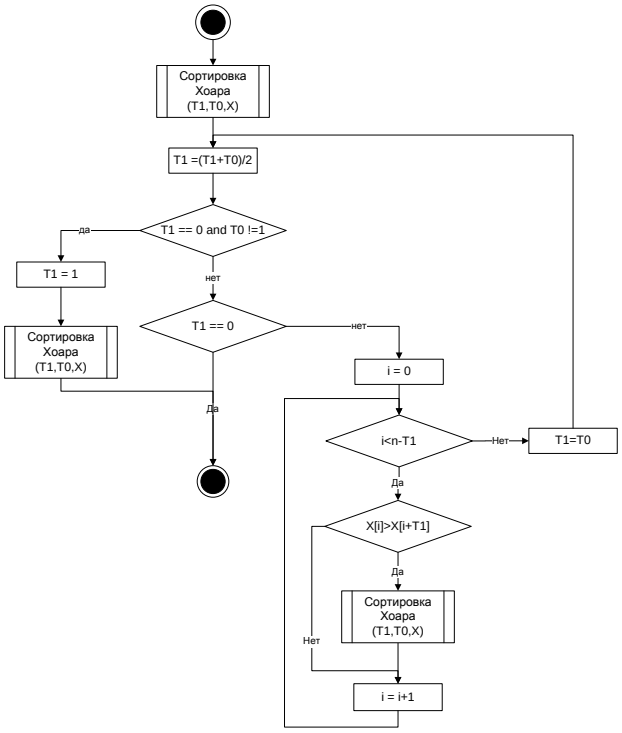
Сколько обменов элементов выполнит алгоритм с t1 = 4 на массиве
2 5 7 8 1 9 2 4 6 1 5 4?

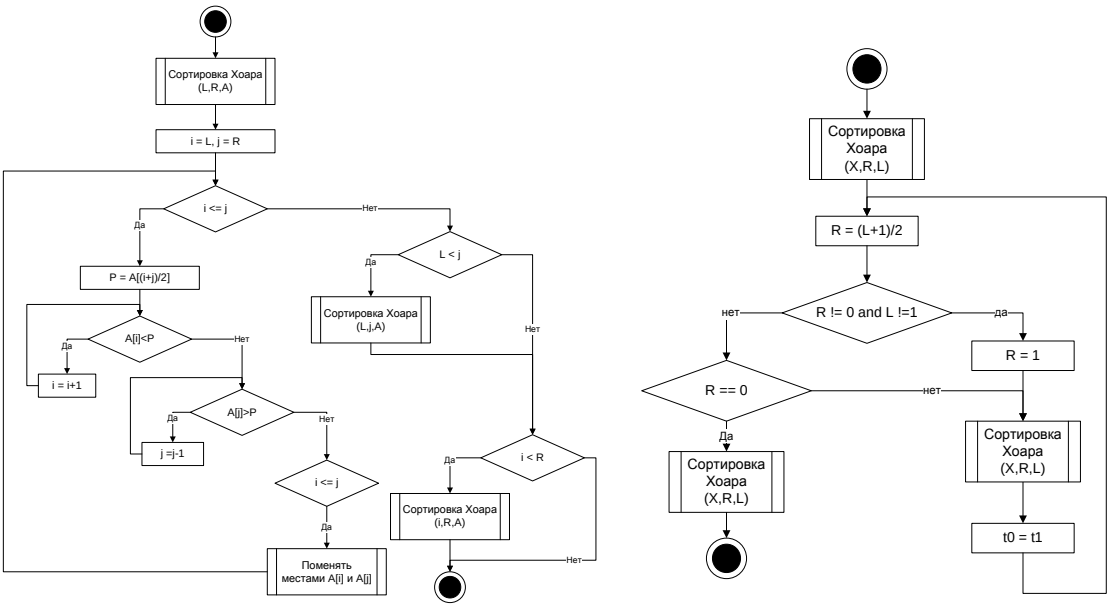
5
6
7
8

Вопрос 7

Детализация алгоритма, присущая процессу нисходящего программирования, может быть выполнена по методике Дамке. Какое графическое представление соответствует алгоритму сортировки Хоара.







Вопрос 8

Детализация алгоритма, которая проводится в процессе нисходящего программирования, может быть выполнена по методике Насси-Шнейдермана. Какое графическое представление соответствует алгоритму пирамидальной сортировки.

X[n] - исходный массив	
For i от 0 до n/2	
	Построить пирамидально упорядоченный массив с началом в элементе с индексом i
	i = i+1
L = n-1	
Пока L>=1	
	Поменять местами X[L] и X[0]
	Построить пирамидально упорядоченный массив с началом в элементе с индексом L
Распечатать упорядоченный массив	

X[n] - исходный массив	
For i от 0 до n-1	
	Построить пирамидально упорядоченный массив с началом в элементе с индексом i
	i = i+1
L = n	
Пока L>=1	
	Поменять местами X[L] и X[n-1]
	Построить пирамидально упорядоченный массив с началом в элементе с индексом L
Распечатать упорядоченный массив	

X[n] - исходный массив	
For i от n до n/2	
	Построить пирамидально упорядоченный массив с началом в элементе с индексом i
	i = i-1
L = n-1	
Пока L>=1	
	Поменять местами X[L] и X[0]
	Построить пирамидально упорядоченный массив с началом в элементе с индексом 0
	L = L-1
Распечатать упорядоченный массив	

X[n] - исходный массив	
For i от n/2 до 0	
	Построить пирамидально упорядоченный массив с началом в элементе с индексом i
	i = i-1
L = n-1	
Пока L>=1	
	Поменять местами X[L] и X[0]
	Построить пирамидально упорядоченный массив с началом в элементе с индексом 0
	L = L-1
Распечатать упорядоченный массив	

Вопрос 9

Согласно принципам структурного программирования, для разработки программ может

быть использована модульная декомпозиция решаемой задачи. Для реализации библиотеки функций для работы с очередью с приоритетами были спроектированы следующие модули:

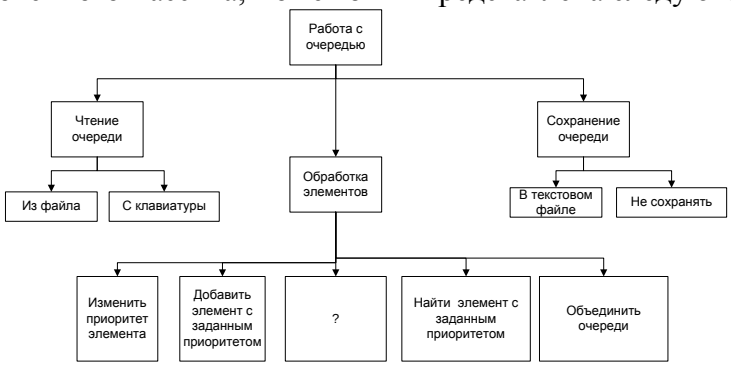
```
Изменить приоритет(
                Очередь A,
                Текущее значение приоритета a,
                Новое значение приоритета b);
Добавить элемент с заданным приоритетом(
                Очередь A,
                Значение приоритета p);
Найти элемент (
               Очередь A,
               Значение искомого приоритета p)
Объединить очереди(
                Очередь A,
                Очередь B)
Удалить элемент с максимальным приоритетом(
                Очередь B)
```

Элементы очереди хранятся в виде односвязного динамического неотсортированного списка. Какое из предложенных описаний функции `Добавить элемент с заданным приоритетом` является менее затратным с точки зрения времени выполнения функции?

функция добавляет элемент с заданным приоритетом в конец очереди
функция добавляет элемент с заданным приоритетом в начало очереди
функция добавляет элемент с заданным приоритетом в заданное место
функция добавляет элемент с заданным приоритетом после элемента с заданным значением

Вопрос 10

Согласно принципам структурного программирования, для разработки программ может быть использована иерархическая декомпозиция решаемой задачи. Иерархическая декомпозиция задачи написания библиотеки функций для работы с очередью, представленной в виде упорядоченного массива, может быть представлена следующим образом:



Какой модуль, реализующий одну из основных операций над очередью, отсутствует на схеме?

удалить элемент с максимальным приоритетом
поменять местами элементы очереди
добавить элемент в начало очереди
удалить элемент из конца очереди

Вопрос 11

Студенту дано задание разработать библиотеку функций для работы с очередью, элементы которой хранятся в виде неупорядоченного массива. Какая из реализаций функции добавления элемента в массив является верной и наиболее эффективной?

```
void Add1(int *x,int n, int a){
    x = (int*)realloc (x, (n+1)*sizeof(int));
    x[n] = a;
}
```

```
int *Add(int *x,int n, int a){
    x = (int*)realloc (x, (n+1)*sizeof(int));
    x[n] = a;
    return x;
}
```

```
int *Add2(int *x,int n, int a){
    int i;
    int* y = (int*)malloc ((n+1)*sizeof(int));
    for (i=0;i<n;i++)
      y[i+1] = x[i];
    y[0] = a;
    free(x);
    return y;
}
```

```
int *Add3(int *x,int n, int a){
    x[n] = a;
    return x;
}
```

Вопрос 12

Студенту дано задание разработать библиотеку функций для работы с очередью, элементы которой хранятся в виде упорядоченного массива. Какая из реализаций функции добавления элемента в массив является верной и наиболее эффективной?

```
int *Add2(int *x,int n, int a){
    int i,j;
    x = (int*)realloc (x, (n+1)*sizeof(int));
    x[n] = a;
    for(i=0;i<n-1;i++)
    for(j=0;j<n-1-i;j++)
       if(x[j]>x[j+1]) swap(&x[j],&x[j+1]);
    return x;
}
```

```
int *Add(int *x,int n, int a){
    x = (int*)realloc (x, (n+1)*sizeof(int));
    x[n] = a;
    return x;
}
```

```
int *Add1(int *x,int n, int a){
    int j;
    x = (int*)realloc (x, (n+1)*sizeof(int));
    j = n-1;
    while(j>=0&&a<x[j]){
        x[j+1]=x[j];
        j--;
    }
    x[j+1] = a;
    return x;
}
```

```
int *Add2(int *x,int n, int a){
    int i;
    int* y = (int*)malloc ((n+1)*sizeof(int));
    for (i=0;i<n;i++)
      y[i+1] = x[i];
    y[0] = a;
    free(x);
    return y;
}
```

Вопрос 13

Студенту дано задание разработать библиотеку функций для работы с очередью, элементы которой хранятся в виде упорядоченного однонаправленного списка. Какая из реализаций функции добавления элемента в массив является верной и наиболее эффективной?

```
Node *Add( Node *Top, int d ) {
   Node *newel = (Node*) malloc(sizeof(Node));
   newel->data = d;
   newel->next = Top;
return newel;  }
```

```
Node *Add1( Node *Top, int d ) {
    Node *newel = (Node*) malloc(sizeof(Node));
    newel->data = d;
    if(!Top) {
    newel->next = Top;
    return newel;}
    Node *cur = Top;
    while(cur->next) cur = cur->next;
    cur->next = newel;
    newel->next = NULL;
    return Top;
      }
```

```
Node * Add2(Node *Top, int k){
    Node *newel = (Node*) malloc(sizeof(Node));
    newel->data = k;
    if (!Top) {
        newel->next = Top;
        return newel;
    }
    Node *s = Top; Node* s1 = s->next;
    while (s1&&k!=s1->data){
     s = s1;
     s1 = s->next;
    }
    s -> next = newel;
    newel->next = s1;
    return Top;
}
```

```
Node * Add_S(Node *Top, int k){
    Node *newel = (Node*) malloc(sizeof(Node));
    newel->data = k;
    if (!Top||Top->data>=k) {
        newel->next = Top;
        return newel;
    }
    Node *s = Top; Node* s1 = s->next;
    while (s1&&k>s1->data){
     s = s1;
     s1 = s->next;
    }
    s -> next = newel;
    newel->next = s1;
    return Top;
}
```

Вопрос 14

Студенту дано задание разработать библиотеку функций для работы с очередью, элементы которой хранятся в виде неупорядоченного однонаправленного списка. Какая из реализаций функции добавления элемента в массив является верной и наиболее эффективной?

```
Node *Add( Node *Top, int d ) {
   Node *newel = (Node*) malloc(sizeof(Node));
   newel->data = d;
   newel->next = Top;
return newel;  }
```

```
Node *Add1( Node *Top, int d ) {
    Node *newel = (Node*) malloc(sizeof(Node));
    newel->data = d;
    if(!Top) {
    newel->next = Top;
    return newel;}
    Node *cur = Top;
    while(cur->next) cur = cur->next;
    cur->next = newel;
    newel->next = NULL;
    return Top;
      }
```

```
Node * Add2(Node *Top, int k){
    Node *newel = (Node*) malloc(sizeof(Node));
    newel->data = k;
    if (!Top) {
        newel->next = Top;
        return newel;
    }
    Node *s = Top; Node* s1 = s->next;
    while (s1&&k!=s1->data){
     s = s1;
     s1 = s->next;
    }
    s -> next = newel;
    newel->next = s1;
    return Top;
}
```

```
Node * Add_S(Node *Top, int k){
    Node *newel = (Node*) malloc(sizeof(Node));
    newel->data = k;
    if (!Top||Top->data>=k) {
        newel->next = Top;
        return newel;
    }
    Node *s = Top; Node* s1 = s->next;
    while (s1&&k>s1->data){
     s = s1;
     s1 = s->next;
    }
    s -> next = newel;
    newel->next = s1;
    return Top;
}
```

Вопрос 15

Реализация алгоритма бинарного поиска выполнена следующим образом:

```
int BinaryS(int* x, int n, int key){
    int a = 0, b= n-1,m;
    while(a<b){
        m = (a+b)/2;
        if (x[m]==key) return m;
        ...
    }
    return -1;
}
```

Какой фрагмент программы необходимо вставить вместо многоточия?

```
if (x[m]>key) a = m-1; else b = m+1;
```

```
if (x[m]>key) b = m-1; else a = m+1;
```

```
if (x[m]>key) b = m/2; else a = m/2+1;
```

```
if (x[m]>key) b = m; else a = a;
```

Вопрос 16

Реализация алгоритма интерполяционного поиска выполнена следующим образом:

```
int I_S(int* x, int n, int key){
    int a = 0, b= n-1,m;
    while(a<b){
        ...
        if (x[m]==key) return m;
        if (x[m]>key) b = m-1; else a = m+1;
    }
    return -1;
}
```

Какой фрагмент программы необходимо вставить вместо многоточия?

```
if (x[m]>key) b = m-1; else a = m+1;
```

```
if (x[m]>key) a = m-1; else b = m+1;
```

```
m = a + (key-x[a])/(x[a]-x[b])*(b-a);
```

```
m = b + (key-x[b])/(x[a]-x[b])*(b+a);
```

Вопрос 17

Какая из описанных структур данных может быть использована для программной реализации функций работы с BST-деревьями?

```
struct node{
      int data;
      struct node *next;}
--------------------------
struct node{
      int data;
      float prev
      struct node *next;
 }
--------------------------
  struct node{
      int data;
      int *next;}
--------------------------
```

```
struct node{
    int data;
    struct node *next;
   struct node *next1}
```

Вопрос 18

Какая из описанных структур данных может быть использована для программной реализации функций работы с динамическими однонаправленными списками?

```
struct node{
      int data;
      struct node *next;}
-------------------------
struct node{
      int data;
      float prev
      struct node *next;
 }
-------------------------

struct node{
      int data;
      int *next;}
-------------------------
struct node{
      int data;
      struct node *next;
     struct node *next1}
```

Вопрос 19

Какая реализация функции обхода дерева выведет значения ключей, хранящихся в дереве в порядке возрастания?

```
-------------------------
Просмотр (Node *Root)
   Если (Root - пуст) то return
   иначе
       печать (Root->data)
       Просмотр (Root->left)
       Просмотр(Root->right)
 конец
-------------------------
   Просмотр (Node *Root)
   Если (Root - пуст) то return
   иначе
       Просмотр (Root->left)
       печать (Root->data)
       Просмотр(Root->right)
 конец
-------------------------
   Просмотр (Node *Root)
   Если (Root - пуст) то return
   иначе
       Просмотр (Root->left)
       Просмотр(Root->right)
       печать (Root->data)
```

конец

Просмотр(Root *h, int n, int a, int b)
Перейти на позицию с координатами
((a+b)/2,n);
Если (h - пуст)
то
печать(h->data);
Просмотр(h->rigth,
n+1,
(a+b)/2,b);
Просмотр(h->left,
n+1,a,
(a+b)/2);

конец

Вопрос 20

Какая из предложенных функций реализует алгоритм прямого слияния?

```
int * merge1(int *x,int l,int m, int r){
   int i,j,k=0;
   int *y = (int*) malloc(sizeof(int)*(r-l+1));
   for(i=l;i<=m;i++)
   y[k++] = x[i];
   for(;i<=r;i++) {
   j = i-1;
   while(x[i]<y[j]&&j>=0){
            y[j+1] = y[j];j--;
      }
   y[j+1] = x[i];
}
   free(x);
   return y;
}
```

```
int * merge2(int *x,int l,int m, int r){
   int i=l,j = m+1,k=0;
   int *y = (int*) malloc(sizeof(int)*(r-l+1));
   while(i<=m&&j<=r){
    if(x[i]<x[j]) y[k++] = x[i],i++;
      else y[k++] = x[j],j++;
   }
   if(i<m) while(i<=m)y[k++] = x[i++];
   if(j<r) while(j<=r)y[k++] = x[j++];
   free(x);
   return y;
}
```

```
int * merge3(int *x,int l,int m, int r){
   int i,j,k=0;
   int *y = (int*) malloc(sizeof(int)*(r-l+1));
   for(i=m+1;i<=r;i++)
   y[k++] = x[i];
   for(i=l;i<=m;i++) {
   j = i-1;
   while(x[i]<y[j]&&j>=0){
            y[j+1] = y[j];j--;
      }
   y[j+1] = x[i];
}
   free(x);
   return y;
}
```

```
int * merge(int *x,int l,int m, int r){
   int i,j,k=0;
   int *y = (int*) malloc(sizeof(int)*(r-l+1));
   for(i=l;i<=m;i++)
   y[k++] = x[i];
   for(i=r;i>=m+1;i--)
   y[k++] = x[i];
   i=0;j=r-l;k=l;
   while(i<=j){
    if(y[i]<y[j]) x[k++] = y[i],i++;
      else x[k++] = y[j],j--;
   }
   free(y);
   return x;
}
```

14.1.2. Темы контрольных работ

Сортировка

14.1.3. Темы опросов на занятиях

1. Прямой поиск подстроки в строке.
2. Алгоритм Боуера-Мура.

3. Алгоритм Кнута, Морриса и Пратта.

14.1.4. Зачёт

Пример билета

Теоретическая часть

1. Запишите алгоритм сортировки обменом.

2. Продемонстрируйте работу сортировки Шелла на массиве
9 8 0 6 13 7 18 11 1 13 16 13 12 1 4 16 3 10 1 3 с шагом 3

3. Нарисуйте дерево разбиений, которое строит нисходящая сортировка слиянием при обработке массива из 25 элементов.

4. Продемонстрируйте алгоритм MSD сортировки на массиве:
40464
18401
61242
74330
75675
99500
12042
13384
25745
33713

5. Постройте таблицу сдвигов (КМП-алгоритм) для образа «тригонометрия»

6. Постройте дерево методом вставки в лист, если элементы в дерево добавлялись в следующем порядке 13 12 8 18 22 0 16 25 29 13 6 17 13 17 15 16. Продемонстрируйте ротацию влево в узле 18.

14.1.5. Вопросы для подготовки к практическим занятиям, семинарам

1. Расскажите алгоритм прямого поиска подстроки в строке.

2. Сколько сравнений выполнит алгоритм прямого поиска при поиске подстроки «пример» в тексте «рассматриваемый в учебном пособии при-мер может помочь в формировании представления о работе алгоритма»?

3. Запишите таблицу сдвигов, которая будет построена при реализа-ции алгоритма Боуера-Мура, выполняющего поиск подстроки в тексте, который составлен из символов русского алфавита, знаков препинания и пробелов.

4. Расскажите принцип работы алгоритма Кнута, Морриса и Пратта.

5. Постройте таблицу сдвигов алгоритма Кнута, Морриса и Пратта, которая будет организована для поиска подстроки «аквариумистика».

14.1.6. Темы лабораторных работ

Простые сортировки на месте

Двоичные деревья - 1

Двоичные деревья - 2

14.1.7. Методические рекомендации

Темы для самостоятельного изучения

1. Поразрядные сортировки.
2. Динамические структуры.
3. Двоичные деревья поиска (BST-деревья).
4. Поиск.

Вопросы, подлежащие изучению, и рекомендуемые источники перечислены в соответствующем разделе учебно-методического пособия.

Для подготовки к зачету, лабораторным работам, контрольным работам и практическим занятиям рекомендуется повторить соответствующие разделы учебно-методических

пособий.

14.2. Требования к оценочным материалам для лиц с ограниченными возможностями здоровья и инвалидов

Для лиц с ограниченными возможностями здоровья и инвалидов предусмотрены дополнительные оценочные материалы, перечень которых указан в таблице 14.

Таблица 14 – Дополнительные материалы оценивания для лиц с ограниченными возможностями здоровья и инвалидов

Категории обучающихся	Виды дополнительных оценочных материалов	Формы контроля и оценки результатов обучения
С нарушениями слуха	Тесты, письменные самостоятельные работы, вопросы к зачету, контрольные работы	Преимущественно письменная проверка
С нарушениями зрения	Собеседование по вопросам к зачету, опрос по терминам	Преимущественно устная проверка (индивидуально)
С нарушениями опорно-двигательного аппарата	Решение дистанционных тестов, контрольные работы, письменные самостоятельные работы, вопросы к зачету	Преимущественно дистанционными методами
С ограничениями по общемедицинским показаниям	Тесты, письменные самостоятельные работы, вопросы к зачету, контрольные работы, устные ответы	Преимущественно проверка методами исходя из состояния обучающегося на момент проверки

14.3. Методические рекомендации по оценочным материалам для лиц с ограниченными возможностями здоровья и инвалидов

Для лиц с ограниченными возможностями здоровья и инвалидов предусматривается доступная форма предоставления заданий оценочных средств, а именно:

- в печатной форме;
- в печатной форме с увеличенным шрифтом;
- в форме электронного документа;
- методом чтения ассистентом задания вслух;
- предоставление задания с использованием сурдоперевода.

Лицам с ограниченными возможностями здоровья и инвалидам увеличивается время на подготовку ответов на контрольные вопросы. Для таких обучающихся предусматривается доступная форма предоставления ответов на задания, а именно:

- письменно на бумаге;
- набор ответов на компьютере;
- набор ответов с использованием услуг ассистента;
- представление ответов устно.

Процедура оценивания результатов обучения лиц с ограниченными возможностями здоровья и инвалидов по дисциплине предусматривает предоставление информации в формах, адаптированных к ограничениям их здоровья и восприятия информации:

Для лиц с нарушениями зрения:

- в форме электронного документа;
- в печатной форме увеличенным шрифтом.

Для лиц с нарушениями слуха:

- в форме электронного документа;
- в печатной форме.

Для лиц с нарушениями опорно-двигательного аппарата:

- в форме электронного документа;
- в печатной форме.

При необходимости для лиц с ограниченными возможностями здоровья и инвалидов процедура оценивания результатов обучения может проводиться в несколько этапов.