Государственное образовательное учреждение
"Приднестровский государственный университет им. Т.Г. Шевченко"

Физико-технический институт
Факультет информатики и вычислительной техники
Кафедра программного обеспечения вычислительной техники

УТВЕРЖДАЮ

Заведующий кафедрой ПОВТ


С.Г. Федорченко

«28» августа 2023 г.

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ

по дисциплине

Б1.О.22 «АЛГОРИТМЫ И СТРУКТУРЫ ДАННЫХ»

Направление подготовки

2.09.03.02 Информационные системы и технологии

Профиль подготовки
Безопасность информационных систем

Квалификация (степень) выпускника:	**бакалавр**
Форма обучения:	**очная, заочная**
Год набора:	**2021 г.**

Разработал:
к.п.н., доцент кафедры ПОВТ

/А.В.Кирсанова

«28» августа 2023 г.

Тирасполь, 2023

Паспорт фонда оценочных средств по учебной дисциплине

1. **В результате изучения дисциплины: Б1.О.22 «Алгоритмы и структуры данных» у обучающегося должны быть сформированы следующие компетенции:**

Категория (группа) компетенций	Код и наименование	Код и наименование индикатора достижения универсальной компетенции
Общепрофессиональные компетенции выпускников и индикаторы их достижения		
	ОПК-1. Способен применять естественнонаучные и общеинженерные знания, методы математического анализа и моделирования, теоретического и экспериментального исследования в профессиональной деятельности	ИД-1$_{ОПК-1}$ Знать основы высшей математики, физики, вычислительной техники и программирования ИД-2$_{ОПК-1}$ Уметь решать стандартные профессиональные задачи с применением естественнонаучных и общеинженерных знаний, методов математического анализа и моделирования ИД-3$_{ОПК-1}$ Иметь навыки: теоретического и экспериментального исследования объектов профессиональной деятельности.
	ОПК-6. Способен разрабатывать алгоритмы и программы, пригодные для практического применения в области информационных систем и технологий	ИД-1$_{ОПК-6}$ Знать: методы алгоритмизации, языки и технологии программирования, пригодные для практического применения в области информационных систем и технологий. ИД-2$_{ОПК-6}$ Уметь: применять методы алгоритмизации, языки и технологии программирования при решении профессиональных задач в области информационных систем и технологий. ИД-3$_{ОПК-6}$ Иметь навыки: программирования. отладки и тестирования прототипов программно-технических комплексов задач

2. Программа оценивания контролируемой компетенции:

Текущая аттестация	Контролируемые модули, разделы (темы) дисциплины их название	Код контролируемой компетенции (или ее части)	Наименование оценочного средства
РУБЕЖНЫЙ КОНТРОЛЬ	*Раздел 1.* Анализ сложности и эффективности алгоритмов.	ОПК-1, ОПК-6	КР1, ЛБ1-ЛБ8
	Раздел 2. Алгоритмы поиска и сортировки.	ОПК-1, ОПК-6	КР1, ЛБ2-ЛБ4
	Раздел 3. Хеширование.		КР1, ЛБ6
РУБЕЖНАЯ АТТЕСТАЦИЯ	*Раздел 4.* Алгоритмы на графах.	ОПК-1, ОПК-6	КР2, ЛБ5
	Раздел 5. Алгоритмы на деревьях.		КР2, ЛБ7
	Раздел 6. NP-полные и трудно решаемые задачи.	ОПК-1, ОПК-6	КР2, ЛБ8
Промежуточная аттестация		ОПК-1, ОПК-6	Экзамен

3. Показатели и критерии оценивания компетенции по этапам формирования, описание шкал оценивания

Этапы оценивания компетенции	Показатели достижения заданного уровня освоения компетенции	Критерии оценивания результатов обучения			
		2	**3**	**4**	**5**
Первый этап	**Знать** ОПК-1	Не знает основы вычислительной техники и программирования	Знает основы вычислительной техники и программирования, но допускает существенные ошибки	Знает основы вычислительной техники и программирования, но допускает неточности	Знает основы вычислительной техники и программирования
Второй этап	**Уметь** ОПК-1	Не умеет решать стандартные профессиональные задачи с применением естественнонау	Умеет решать стандартные профессиональные задачи с применением естественнонаучных и	Умеет решать стандартные профессиональные задачи с применением естественнонаучных и	умеет решать стандартные профессиональные задачи с применением естественнонаучных и

		чных и общеинженерных знаний, методов математического анализа и моделирования	общеинженерных знаний, методов математического анализа и моделирования, но допускает существенные ошибки	общеинженерных знаний, методов математического анализа и моделирования, но допускает неточности	общеинженерных знаний, методов математического анализа и моделирования
Третий этап	**Владеть** ОПК-1	Не владеет навыками программирования. отладки и тестирования прототипов программно-технических комплексов задач	Владеет навыками программирования. отладки и тестирования прототипов программно-технических комплексов задач, но допускает существенные ошибки	Владеет навыками программирования. отладки и тестирования прототипов программно-технических комплексов задач, но допускает неточности	Владеет навыками программирования. отладки и тестирования прототипов программно-технических комплексов задач
Первый этап	**Знать** ОПК-6	Не знает методы алгоритмизации, языки и технологии программирования, пригодные для практического применения в области информационных систем и технологий	Знает методы алгоритмизации, языки и технологии программирования, пригодные для практического применения в области информационных систем и технологий, но допускает существенные ошибки	Знает методы алгоритмизации, языки и технологии программирования, пригодные для практического применения в области информационных систем и технологий, но допускает неточности	Знает методы алгоритмизации, языки и технологии программирования, пригодные для практического применения в области информационных систем и технологий
Второй этап	**Уметь** ОПК-6	Не умеет применять методы алгоритмизации, языки и технологии программирования при решении профессиональных задач в области	Умеет применять методы алгоритмизации, языки и технологии программирования при решении профессиональных задач в области	Умеет применять методы алгоритмизации, языки и технологии программирования при решении профессиональных задач в области	Умеет применять методы алгоритмизации, языки и технологии программирования при решении профессиональных задач в области информационных систем и технологий

		информационных систем и технологий	информационных систем и технологий, но допускает существенные ошибки	информационных систем и технологий, но допускает неточности	
Третий этап	**Владеть** ОПК-6	Не имеет навыки программирования, отладки и тестирования прототипов программно-технических комплексов задач	Имеет навыки программирования, отладки и тестирования прототипов программно-технических комплексов задач, но допускает существенные ошибки	Имеет навыки программирования, отладки и тестирования прототипов программно-технических комплексов задач, но допускает неточности	Имеет навыки программирования, отладки и тестирования прототипов программно-технических комплексов задач

4. Шкала оценивания

Согласно Положению «О порядке организации аттестации в ИТИ ПГУ им. Т.Г. Шевченко, итоговая оценка представляет собой сумму баллов, полученных студентом по итогу освоения дисциплины (модуля):

Оценка в традиционной шкале	Оценка в 100-балльной шкале	Буквенные эквиваленты оценок в шкале ЗЕ (% успешно аттестованных)
5 (отлично)	88–100	A (отлично) – 88-100 баллов
4 (хорошо)	70–87	B (очень хорошо) – 80-87баллов
		C (хорошо) – 70-79 баллов
3 (удовлетворительно)	50–69	D(удовлетворительно) – 60-69 баллов
		E(посредственно) – 50-59 баллов
2 (неудовлетворительно)	0–49	Fx– неудовлетворительно, с возможной пересдачей – 21-49 баллов
		F– неудовлетворительно, с повторным изучением дисциплины – 0-20 баллов

Расшифровка уровня знаний, соответствующего полученным баллам, дается в таблице, указанной ниже

A	“Отлично” - теоретическое содержание курса освоено полностью, без пробелов, необходимые практические навыки работы с освоенным материалом сформированы, все предусмотренные программой обучения учебные задания выполнены, качество их выполнения оценено числом баллов, близким к максимальному.
B	“Очень хорошо” - теоретическое содержание курса освоено полностью, без пробелов, необходимые практические навыки работы с освоенным материалом в основном сформированы, все предусмотренные программой обучения учебные задания выполнены, качество выполнения большинства из них оценено числом

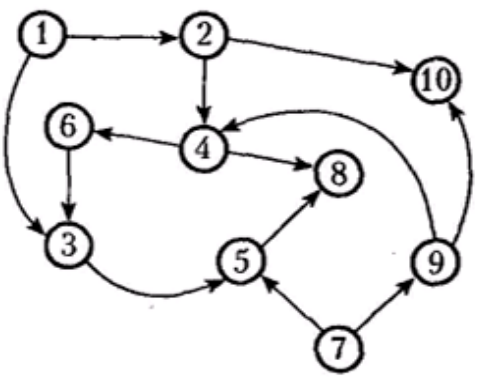
	баллов, близким к максимальному.
C	“Хорошо” - теоретическое содержание курса освоено полностью, без пробелов, некоторые практические навыки работы с освоенным материалом сформированы недостаточно, все предусмотренные программой обучения учебные задания выполнены, качество выполнения ни одного из них не оценено минимальным числом баллов, некоторые виды заданий выполнены с ошибками.
D	“Удовлетворительно” - теоретическое содержание курса освоено частично, но пробелы не носят существенного характера, необходимые практические навыки работы с освоенным материалом в основном сформированы, большинство предусмотренных программой обучения учебных заданий выполнено, некоторые из выполненных заданий, возможно, содержат ошибки.
E	“Посредственно” - теоретическое содержание курса освоено частично, некоторые практические навыки работы не сформированы, многие предусмотренные программой обучения учебные задания не выполнены, либо качество выполнения некоторых из них оценено числом баллов, близким к минимальному.
FX	“Условно неудовлетворительно” - теоретическое содержание курса освоено частично, необходимые практические навыки работы не сформированы, большинство предусмотренных программой обучения учебных заданий не выполнено, либо качество их выполнения оценено числом баллов, близким к минимальному; при дополнительной самостоятельной работе над материалом курса возможно повышение качества выполнения учебных заданий.
F	“Безусловно неудовлетворительно” - теоретическое содержание курса не освоено, необходимые практические навыки работы не сформированы, все выполненные учебные задания содержат грубые ошибки, дополнительная самостоятельная работа над материалом курса не приведет к какому-либо значимому повышению качества выполнения учебных заданий.

***5.* Типовые контрольные задания или иные материалы, необходимые для оценки знаний, умений, навыков и (или) опыта деятельности, характеризующих этапы формирования компетенций при изучении учебной дисциплины в процессе освоения образовательной программы**

5.1 Типовой вариант задания на контрольную работу 1.

1. Понятие сложности алгоритмов. Анализ сложности алгоритмов. Классы входных данных. Примеры. Асимптотические обозначения скоростей роста (с примерами).

2. Алгоритмы сортировки: основные понятия, параметры оценки. Теорема о высоте любого разрешающего дерева, сортирующего *n* элементов. Изложить суть сортировок: вставками, шейкер-сортировки, внешнего прямого слияния. Время работы всех упомянутых сортировок.



3. Выполните топологическую сортировку графа, используя метод «расстановки будильников».

5.2 Типовой вариант задания на контрольную работу 2

1. Определение понятий: хеширование, хеш-функция, хеш-таблица, ключ, хеш-значение, коллизия. Требования к хеш-функции.
2. Построение бинарного дерева поиска (БДП). Описание алгоритмов вставки, удаления, поиска заданного элемента и поиска следующего по величине элемента в БДП. Код поиска заданного элемента.
3. Числовые алгоритмы: быстрые алгоритмы сложения, умножения, деления, операций над натуральными числами по модулю, возведение в степень, определения простоты числа, поиск НОД.
4. NP-задачи. Метод сведения задачи к другой задаче. Пример сведения любой NP-задачи к задаче CIRCUIT-SAT.

5.3 Типовой вариант задания на лабораторную работу 1. Анализ алгоритмов.

1) Напишите программу, подсчитывающую количество прописных букв в текстовом файле. Сколько сравнений требуется этому алгоритму? Каково максимальное возможное значение числа операций увеличения счетчика? Минимальное такое число? (Выразите ответ через число N символов во входном файле.)

2) В файле записан некоторый набор чисел, однако мы не знаем, сколько их. Напишите программу для подсчета среднего значения чисел в файле. Какого типа операции делает Ваш алгоритм? Сколько операций каждого типа он делает?

3) Напишите программу, не использующую сложных условий, которая по трем введенным целым числам определяет, различны ли они все между собой. Сколько сравнений в среднем делает Ваш алгоритм? Не забудьте исследовать все классы входных данных.

4) Напишите программу, которая получает на входе три целых числа, и находит наибольшее из них. Каковы возможные классы входных данных? На каком из них Ваш алгоритм делает наибольшее число сравнений? На каком меньше всего? (Если разницы между наилучшим и наихудшим классами нет, перепишите свой алгоритм с простыми сравнениями так, чтобы он не использовал временных переменных, и чтобы в наилучшем случае он работал быстрее, чем в наихудшем).

5) Напишите программу для поиска второго по величине элемента в списке из N значений. Сколько сравнений делает Ваш алгоритм в наихудшем случае?

6) Напишите программу подсчета медианы трех целых чисел. Какой случай для алгоритма является наилучшим? Наихудшим? Средним? (Если наилучший и наихудший случаи совпадают, то перепишите Ваш алгоритм с простыми условиями, не пользуясь временными переменными, так, чтобы наилучший случай был лучше наихудшего.)

7) Напишите программу, проверяющую, верно ли, что данные четыре целых числа попарно различны. Какой из классов данных обеспечивает наилучший случай для Вашего алгоритма? Наихудший? Средний? (Если наилучший и наихудший случаи совпадают, то перепишите Ваш алгоритм с простыми условиями, не пользуясь временными переменными, так, чтобы наилучший случай был лучше наихудшего.)

5.4. Типовой вариант задания на лабораторную работу 2. Генерация псевдослучайных чисел.

1. Разберите предложенные алгоритмы генерации случайных чисел. Определите интервал повторения случайных чисел.
2. Напишите на основе алгоритма последовательного поиска функцию и протестируйте

ее.
3. С помощью одного из методов сгенерируйте список случайных чисел в промежутке от 1 до N и проводите поиск в этом списке; здесь N может быть любым числом от 100 до 1000. В Вашей программе должен быть глобальный счетчик, который следует установить в нуль в начале программы и увеличивать на 1 непосредственно перед очередным сравнением.
4. Затем главная программа должна вызывать функцию последовательного поиска для каждого элемента от 1 до N. Подсчитанное после этого общее число сравнений следует разделить на N; результатом будет среднее число сравнений при последовательном поиске.
5. Повторите шаг 3 для двоичного поиска в упорядоченном списке.
6. Повторите шаг 3 для выборки из списка заданного К-го минимального элемента.
7. Составьте отчет, в котором сравните результаты третьего, пятого и шестого шагов с предсказаниями анализа.

5.5. Типовой вариант задания на лабораторную работу 3. Сортировка массивов и последовательностей

Сгенерировать случайные данные не менее 50 и записать в массив или файл (в зависимости от вида сортировки – внешняя или внутренняя). Отсортировать полученную последовательность:

Вариант 1: методом вставки. Сколько сравнений выполнит алгоритм? Определите зависимость сложности от N.

Вариант 2: методом выбора. Сколько сравнений выполнит алгоритм? Определите зависимость сложности от N.

Вариант 3: методом классического пузырька. Сколько сравнений выполнит алгоритм? Определите зависимость сложности от N.

Вариант 4: методом шейкер сортировки. Сколько сравнений выполнит алгоритм? Определите зависимость сложности от N.

Вариант 5: методом Шелла. Сколько сравнений выполнит алгоритм? Определите зависимость сложности от N.

Вариант 6: методом быстрой сортировки. Сколько сравнений выполнит алгоритм? Определите зависимость сложности от N.

Вариант 7: методом пирамидальной сортировки. Сколько сравнений выполнит алгоритм? Определите зависимость сложности от N.

5.6. Типовой вариант задания на лабораторную работу 4. Алгоритмы поиска и выборки целевого элемента.

1. Разберите предложенные алгоритмы генерации случайных чисел. Определите интервал повторения случайных чисел.
2. Напишите на основе алгоритма последовательного поиска функцию и протестируйте ее.
3. С помощью одного из методов сгенерируйте список случайных чисел в промежутке от 1 до N и проводите поиск в этом списке; здесь N может быть любым числом от 100 до 1000. В Вашей программе должен быть глобальный счетчик, который следует установить в нуль в начале программы и увеличивать на 1 непосредственно перед очередным сравнением.
4. Затем главная программа должна вызывать функцию последовательного поиска для каждого элемента от 1 до N. Подсчитанное после этого общее число сравнений следует разделить на N; результатом будет среднее число сравнений при последовательном поиске.
5. Повторите шаг 3 для двоичного поиска в упорядоченном списке.
6. Повторите шаг 3 для выборки из списка заданного К-го минимального элемента.

7. Составьте отчет, в котором сравните результаты третьего, пятого и шестого шагов с предсказаниями анализа.

5.7. Типовой вариант задания на лабораторную работу 5. Алгоритмы на графах

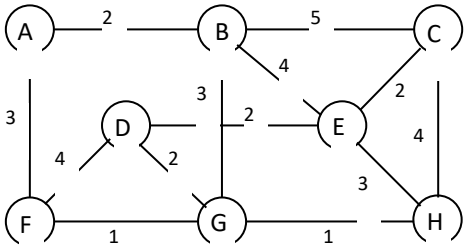
Задание 1

Выполните алгоритм поиска кратчайшего пути из вершины A во все остальные вершины в заданном графе. Подсчитайте количество просмотренных ребер (если обращений несколько, то это нужно учитывать).

G={(a,b,c,d,e,f,g,h),

{a,b}, {b,c}, {c,h}, {h,g}, {g,b}, {g,f}, {f,a}, {f,d}, {g,d}, {d,e}, {b,e},{e,c}, {e,h}}, веса

2, 5, 4, 1, 3, 1, 3, 4, 2, 2, 4, 2, 3



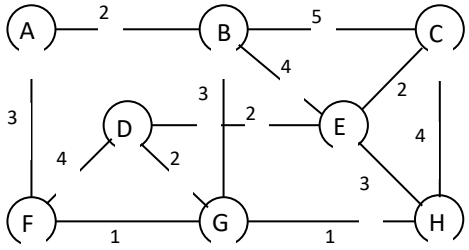
Задание 2

Найдите минимальное остовное дерево с помощью алгоритма Дейкстры-Прима в заданном графе, начиная с вершины A. Выполните анализ алгоритма, подсчитав, сколько раз рассматривается каждое ребро при добавлении вершин к кайме, при обновлении списка ребер ведущих в кайму, и при перенесении вершины из каймы в МОД.

G={(a,b,c,d,e,f,g,h),

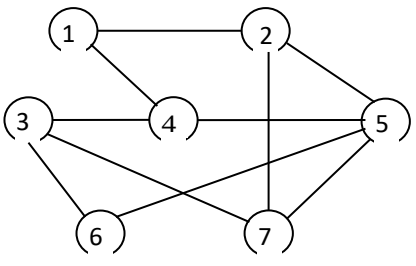
{a,b}, {b,c}, {c,h}, {h,g}, {g,b}, {g,f}, {f,a}, {f,d}, {g,d}, {d,e}, {b,e},{e,c}, {e,h}}, стоимости

2, 5, 4, 1, 3, 1, 3, 4, 2, 2, 4, 2, 3



Задание 3

Напишите программу, выполняющую обход заданного графа в глубину. Граф представлен с помощью матрицы смежности. Укажите порядок посещения вершин при обходе в глубину начиная с вершины 1. G={(1,2,3,4,5,6,7), {1,2}, {1,4}, {2,5}, {5,7}, {2,7}, {5,4}, {5,6}, {7,3}, {6,3}, {4,3}}.



5.8. Типовой вариант задания на лабораторную работу 6. Хеширование.

Исследовать качество указанной хеш-функции и представить график возникновения коллизий. В методе открытой адресации предусмотреть возврат на начало

таблицы со смещением. Размер хеш-таблицы не более 37. Подсчитать количество коллизий при построении хеш-таблицы.

1. Построение хэш-таблицы по методу цепочек. Хеш-функция построена по методу деления. Ключ 1 – числовой.
2. Построение хэш-таблицы по методу открытой адресации путем линейного перебора. Хеш-функция построена по методу деления. Ключ 1 – числовой.
3. Построение хэш-таблицы по методу открытой адресации путем квадратичного перебора. Хеш-функция построена по методу середины квадрата. Ключ 1 – числовой.
4. Построение хэш-таблицы по методу открытой адресации путем двойного хеширования. Хеш-функция построена по методу деления. Ключ 1 – числовой.
5. Построение хэш-таблицы по методу цепочек. Хеш-функция построена по мультипликативному методу. Ключ 1 – числовой.
6. Построение хэш-таблицы по методу открытой адресации путем линейного перебора. Хеш-функция построена по мультипликативному методу. Ключ 1 – числовой.
7. Построение хэш-таблицы по методу открытой адресации путем квадратичного перебора. Хеш-функция построена по мультипликативному методу. Ключ 1 – числовой.

5.9. Типовой вариант задания на лабораторную работу 7. Алгоритмы на бинарных деревьях

1. Построение бинарного дерева на списочной структуре данных и сортировка последовательности по убыванию прохождением бинарного дерева.
2. Построение бинарного дерева на списочной структуре данных и обход дерева в прямом порядке.
3. Построение бинарного дерева на массиве и обход дерева в прямом порядке.
4. Построение бинарного дерева на списочной структуре данных и обход дерева в обратном порядке.
5. Построение бинарного дерева на массиве и обход дерева в обратном порядке.
6. Построение бинарного дерева и сортировка методом турнира с выбыванием.
7. Сжатие текстовой информации методом кодирования Хафмана и ее декодирование. Сравнение результатов работы полученного алгоритма с работой архиватора WinRar.
8. Разбор и вычисление арифметических выражений с помощью построения бинарного дерева.
9. Разбор выражения и вычисление логической функции на всех наборах входов с помощью построения бинарного дерева.

5.10. Типовой вариант задания на лабораторную работу 8. NP-задачи.

Задача «3-Выполнимость (задача 3-ВЫП)». Требуется определить, существует ли набор значений переменных, при котором КНФ равна 1.

Задача о вершинном покрытии графа (задача ВП). Дан неориентированный граф G = (V, E) с множеством вершин V и множеством ребер E. Дано натуральное число K. Вершинным покрытием графа называется подмножество вершин V¢ Í V, такое, что любое ребро графа инцидентно хотя бы одной вершине изV¢. Требуется определить, существует ли вершинное покрытие графа Gмощности K.

Задача о гамильтоновом цикле. Дан граф G = (V, E). Существует ли цикл, ровно один раз проходящий через каждую вершину? В случае ориентированного графа можно поставить задачу об ориентированном гамильтоновом цикле. Она также NP-полна.

Задача о коммивояжере. Дано числоn, матрица расстояний $\{a_{ij}\}$ и число K. Существует ли маршрут коммивояжера (т.е. гамильтонов цикл) с длиной, не превышающей K? (К этой задаче легко сводится задача о гамильтоновом цикле).

Задача о самом длинном пути. Дан граф G = (V, E), заданы номера двух вершин A и B, а также число K. Существует ли путь из A в B с длиной *не менее* K?

Задача о разбиении. Данынпредметов с весами p_i. Можно ли разбить множество предметов на два непересекающихся подмножества с одинаковым весом?

Задача о рюкзаке. Имеетсяnнаименований товара, заданы объемы единиц каждого товара b_i и их стоимость c_i,i = 1…n. Имеется рюкзак объема B. Можно ли подобрать набор товаров, вмещающийся в рюкзак и имеющий при этом стоимость не менее заданного значения K?

5.11 Типовой тест промежуточной аттестации

1. Структура данных представляет собой
   a) набор правил и ограничений, определяющих связи между отдельными элементами и группами данных
   b) набор правил и ограничений, определяющих связи между отдельными элементами данных
   c) набор правил и ограничений, определяющих связи между отдельными группами данных
   d) некоторую иерархию данных

2. Линейный список, в котором доступен только последний элемент, называется
   a) стеком
   b) очередью
   c) деком
   d) массивом
   e) кольцом

3. Структура данных работа с элементами которой организована по принципу FIFO (первый пришел - первый ушел) это –
   а) Стек
   б) Дек
   в) Очередь
   г) Список

4. Линейный последовательный список, в котором включение исключение элементов возможно с обоих концов, называется
   a) стеком
   b) очередью
   c) деком
   d) кольцевой очередью

5. В чём особенности очереди ?
   a) открыта с обеих сторон ;
   b) открыта с одной стороны на вставку и удаление;
   c) доступен любой элемент.

6. В чём сосбенности стека ?
   a) открыт с обеих сторон на вставку и удаление;
   b) доступен любой элемент;
   c) открыт с одной стороны на вставку и удаление.

7. Какую дисциплину обслуживания принято называть FIFO?
   а) стек;

b) очередь;
c) дек.

8. Какая операция читает верхний элемент стека без удаления?
a) pop;
b) push;
b) stackpop.

9. Каково правило выборки элемента из стека?
a)первый элемент;
b)последний элемент;
c)любой элемент.

10. Как освободить память от удаленного из списка элемента?
a) p=getnode;
b) ptr(p)=nil;
c) freenode(p);
d) p=lst.

11.Как создать новый элемент списка с информационным полем D?
a)p=getnode;
b)p=getnode; info(p)=D;
c)p=getnode; ptr(D)=lst.

12. Как создать пустой элемент с указателем p?
a) p=getnode;
b) info(p);
c) freenode(p);
d) ptr(p)=lst.

13. Сколько указателей используется в односвязных списках?
a) 1
b) 2;
c) сколько угодно.

14.В чём отличительная особенность динамических объектов?
a)порождаются непосредственно перед выполнением программы;
b)возникают уже в процессе выполнения программы;
c)задаются в процессе выполнения программы.

15. При удалении элемента из кольцевого списка…
a)список разрывается;
b)в списке образуется дыра;
c)список становится короче на один элемент .

16.Для чего используется указатель в кольцевых списках
a)для ссылки на следующий элемент;
b)для запоминания номера сегмента расположения элемента;
c)для ссылки на предыдущий элемент ;
d)для расположения элемента в списке памяти.

17. Чем отличается кольцевой список от линейного?

a)в кольцевом списке последний элемент является одновременно и первым;
b)в кольцевом списке указатель последнего элемента пустой;
c)в кольцевых списках последнего элемента нет ;
d)в кольцевом списке указатель последнего элемента не пустой.

18. Сколько указателей используется в односвязном кольцевом списке ?
a)1;
b)2;
c)сколько угодно.

19. В каких направлениях можно перемещаться в кольцевом двунаправленном списке?
a)в обоих;
b) влево;
c) вправо.

20. С помощью какой структуры данных наиболее рационально реализовать очередь?
a)стек;
b)список;
c)дек.

21. В памяти ЭВМ бинарное дерево удобно представлять в виде:
a) связанных линейных списков;
b)массивов;
c)связанных нелинейных списков.

22. Элемент t, на который нет ссылок:
a)корнем;
b)промежуточным;
c)терминальным (лист).

23. Дерево называется полным бинарным, если степень исходов вершин равна:
a)2 или 0;
b)2;
c)M или 0;
d)M.

24.Даны три условия окончания просеивания при сортировке прямым включением. Найдите среди них лишнее.
a)найден элемент a(i) с ключом, меньшим чем ключ у x;
b)найден элемент a(i) с ключом, большим чем ключ у x;
c)достигнут левый конец готовой последовательности.

25. Какой из критериев эффективности сортировки определяется формулой M=0,01*n*n+10*n ?
a)число сравнений;
b)время, затраченное на написание программы;
c)количество перемещений;
d)время, затраченное на сортировку.

26. Как называется сортировка, происходящая в оперативной памяти?
a)сортировка таблицы адресов;
b)полная сортировка;
c)сортировка прямым включением;

d)внутренняя сортировка;
внешняя сортировка.

27. Как можно сократить затраты машинного времени при сортировке большого объёма данных?
a)производить сортировку в таблице адресов ключей;
b)производить сортировку на более мощном компьютере;
c)разбить данные на более мелкие порции и сортировать их.

28. Существуют следующие методы сортировки. Найдите ошибку.
a)строгие;
b)улучшенные;
c)динамические

29. Метод сортировки называется устойчивым, если в процессе сортировки…
a)относительное расположенние элементов безразлично;
b)относительное расположение элементов с равными ключами не меняется
c)относительное расположение элементов с равными ключами изменяется;
d)относительное расположение элементов не определено.

30. Улучшенные методы имеют значительное преимущество:
a)при большом количестве сортируемых элементов;
b)когда массив обратно упорядочен;
c)при малых количествах сортируемых элементов;
d)во всех случаях.

31. Что из перечисленных ниже понятий является одним из типов сортировки?
a)внутренняя сортировка;
b)сортировка по убыванию;
c)сортировка данных;
d)сортировка по возрастанию.

32. Сколько сравнений требует улучшенный алгоритм сортировки?
a)n*log(n);
b)en;
c)n*n/4.

33. Сколько сравнений и перестановок элементов требуется в пузырьковой сортировке?
a)n*lon(n);
b)(n*n)/4;
c)(n*n-n)/2.

34. Сколько дополнительных переменных нужно в пузырьковой сортировке помимо массива, содержащего элементы ?
a)0 (не нужно);
b)всего 1 элемент;
c)n переменных (ровно столько, сколько элементов в массиве).

35. Как рассортировать массив быстрее, пользуясь пузырьковым методом?
a)одинаково;
b)по возрастанию элементов;
c)по убыванию элементов.

36. В чём заключается идея метода QuickSort?
a)выбор 1,2,…n – го элемента для сравнения с остальными;
b)разделение ключей по отношению к выбранному ;
c)обмен местами между соседними элементами.

37. Массив сортируется “пузырьковым” методом. За сколько проходов по массиву самый “лёгкий” элемент в массиве окажется вверху ?
a)за 1 проход;
b)за n-1 проходов;
c)за n проходов, где n – число элементов массива.

38. При обходе дерева слева направо получаем последовательность…
a)отсортированную по убыванию;
b)неотсортированную ;
c)отсортированную по возрастанию.

39. При обходе дерева слева направо его элемент заносится в массив…
a)при втором заходе в элемент ;
b)при первом заходе в элемент;
c)при третьем заходе в элемент.

40. Где эффективен линейный поиск ?
a)в списке;
b)в массиве;
c)в массиве и в списке.

41. Какой поиск эффективнее ?
a)линейный;
b)бинарный;
c)без разницы.

42. В чём суть бинарного поиска ?
a)нахожденние элемента массива x путём деления массива пополам каждый раз, пока элемент не найден;
b)нахождение элемента x путём обхода массива;
c)нахождение элемента массива x путём деления массива.

43. Как расположены элементы в массиве бинарного поиск?
a)по возрастанию;
b)хаотично;
c)по убыванию.

44. В чём суть линейного поиск?
a)производится последовательный просмотр от начала до конца и обратно через 2 элемента;
b)производится последовательный просмотр элементов от середины таблицы;
c)производится последовательный просмотр каждого элемента.

45. Где наиболее эффективен метод транспозиции?
a)в массивах и в списках;
b)только в массивах;
c)только в списках.

46. В чём суть метода транспозиции?
a)перестановка местами соседних элементов;

b)нахождение одинаковых элементов;
c)перестановка найденного элемента на одну позицию в сторону начала списка.

47. Что такое уникальный ключ?
a)если разность значений двух данных равна ключу;
b)если сумма значений двух данных равна ключу;
c)если в таблице есть только одно данное с таким ключом.

48. В чём состоит назначение поиска?
a) среди массива данных найти те данные, которые соответствуют заданному аргументу;
b) определить, что данных в массиве нет;
c) с помощью данных найти аргумент.

49. Элемент дерева, который не ссылается на другие, называется
a) корнем
b) листом
c) узлом
d) промежуточным

50. Элемент дерева, на который не ссылаются другие, называется
a) корнем
b) листом
c) узлом
d) промежуточным

51. Элемент дерева, который имеет предка и потомков, называется
a) корнем
b) листом
c) узлом
d) промежуточным

52. Высотой дерева называется
a) максимальное количество узлов
b) максимальное количество связей
c) максимальное количество листьев
d) максимальная длина пути от корня до листа

53. Степенью дерева называется
a) максимальная степень всех узлов
b) максимальное количество уровней его узлов
c) максимальное количество узлов
d) максимальное количество связей
e) максимальное количество листьев

54. Как определяется длина пути дерева
a) как сумма длин путей всех его узлов
b) как количество ребер от узла до вершины
c) как количество ребер от листа до вершины
d) как максимальное количество ребер
e) как максимальное количество листьев
f) как длина самого длинного пути от ближнего узла до какого-либо листа

55. Дерево называется бинарным, если
a) количество узлов может быть либо пустым, либо состоять из корня с двумя другими бинарными поддеревьями
b) каждый узел имеет не менее двух предков
c) от корня до листа не более двух уровней
d) от корня до листа не менее двух уровней

56. Бинарное дерево можно представить
a) с помощью указателей
b) с помощью массивов
c) с помощью индексов
d) правильного ответа нет

57. Какой метод поиска представлен в следующем фрагменте REPEAT I:=I+1 UNTIL (A[I]=X) OR (I=N);
a) последовательный
b) двоичный
c) восходящий
d) нисходящий
e) смешанный

58. Какой метод поиска представлен в следующем фрагменте
REPEAT K:=(I+J)DIV 2; IF X>A[K] THEN I=K+1 ELSE J:=K-1;
UNTIL (A[K]=X) OR (I>J);
a) последовательный
b) бинарный
c) восходящий
d) нисходящий
e) смешанный

59. Реализация поиска в линейном списке выглядит следующим образом
a) WHILE (P<>NIL) AND (P^.KEY<>X) DO P:=P^.NEXT
b) WHILE (P<>NIL) DO P:=P^.NEXT
c) WHILE AND (P^.KEY<>X) DO P:=P^.NEXT
d) WHILE (P<>NIL) AND (P^.KEY<>X) P:=P^.NEXT
e) WHILE (P<>NIL P^.KEY<>X) DO P:=P^.NEXT

60. Как называются предки узла, имеющие уровень на единицу меньше уровня самого узла
a) детьми
b) родителями
c) братьями

61. В графах общая идея поиска в глубину состоит в следующем:
a) Поиск начинаем с некоторой фиксированной вершины v_0. Затем выбираем произвольную вершину u, смежную с v_0, и повторяем просмотр от u. Предположим, что находимся в некоторой вершине v. Если существует ещё не просмотренная вершина u, u-v, то рассматриваем её, затем продолжаем поиск с нее. Если не просмотренной вершины, смежной с v, не существует, то возвращаемся в вершину, из которой попали в v, и продолжаем поиск (если $v=v_0$, то поиск закончен);
b) Поиск начинаем с некоторой фиксированной вершины v_0. Затем выбираем произвольную вершину u, смежную с v_0, и повторяем просмотр от u.

Предположим, что находимся в некоторой вершине v. Если существует ещё не просмотренная вершина u, u-v, то рассматриваем её, затем продолжаем поиск с нее. Если не просмотренной вершины, смежной с v, не существует, то возвращаемся в вершину, из которой попали в v, и продолжаем поиск (если v=u, то поиск закончен);

c) Поиск начинаем с некоторой фиксированной вершины v_0. Затем выбираем произвольную вершину u, смежную с v_0, и повторяем просмотр от u. Предположим, что находимся в некоторой вершине v. Если существует ещё не просмотренная вершина u, то рассматриваем её, затем продолжаем поиск с нее. Если не просмотренной вершины, смежной с v, не существует, то возвращаемся в вершину, из которой попали в v, и продолжаем поиск (если $v=v_0$, то поиск закончен).

62. Стандартным способом устранения рекурсии при поиске в глубину является использование:

a) массива;
b) очереди;
c) стека;
d) циклического списка.

63. При поиске в ширину используется:

a) массив;
b) очередь;
c) стек;
e) циклический список.

64. В последовательном файле доступ к информации может быть

a) только последовательным
b) как последовательным, так и произвольным
c) произвольным
d) прямым

65. Граф – это

a) Нелинейная структура данных, реализующая отношение «многие ко многим»;
b) Линейная структура данных, реализующая отношение «многие ко многим»;
c) Нелинейная структура данных, реализующая отношение «многие к одному»;
d) Нелинейная структура данных, реализующая отношение «один ко многим»;
e) Линейная структура данных, реализующая отношение «один ко многим».

66. Узлам (или вершинам) графа можно сопоставить:

a) отношения между объектами;
b) объекты;
c) связи
d) типы отношений
e) множества

67. Рёбрам графа можно сопоставить:

a) связи
b) типы отношений
c) множества
d) объекты;
e) отношения между объектами;

68. Граф, содержащий только ребра, называется.
a) ориентированным
b) неориентированным
c) простым
d) смешанным

69. Граф, содержащий только дуги, называется.
a) ориентированным
b) неориентированным
c) простым
d) смешанным

70. Граф, содержащий дуги и ребра, называется.
a) ориентированным
b) неориентированным
c) простым
d) смешанным

71. Есть несколько способов представления графа в ЭВМ. Какой из способов приведенных ниже не относится к ним.
a) матрица инциденций;
b) матрица смежности;
c) список ребер;
d) массив инцидентности.

72. Если последовательность вершин v_0, v_1, …v_p определяет путь в графе G, то его длина определяется:
a) $\sum_{i=1}^{P} a(v_{i-1}, v_i)$;
b) $\sum_{i=1}^{P} a(v_{i+1}, v_i)$;
c) $\sum_{i=2}^{P} a(v_{i-1}, v_i)$;
d) $\sum_{i=0}^{P} a(v_{i-1}, v_i)$.

73. Каким образом осуществляется алгоритм нахождения кратчайшего пути от вершины s до вершины t
a) нахождение пути от вершины s до всех вершин графа
b) нахождение пути от вершины s до заданной вершины графа
c) нахождение кратчайших путей от вершины s до всех вершин графа
d) нахождение кратчайшего пути от вершины s до вершины t графа
e) нахождение всех путей от каждой вершины до всех вершин графа

74. Суть алгоритма Дейкстры - нахождения кратчайшего пути от вершины s до вершины t заключается
a) вычислении верхних ограничений d[v] в матрице весов дуг a[u,v] для u, v
b) вычислении верхних ограничений d[v]
c) вычислении верхних ограничений в матрице весов дуг a[u,v]
d) вычислении нижних ограничений d[v] в матрице весов дуг a[u,v] для u, v

75. Улучшение d[v] в алгоритме Форда- Беллмана производится по формуле
a) D[v]:=D[u]+a[u,v]
b) D[v]:=D[u]-a[u,v]
c) D[v]:=a[u,v]
d) D[v]:=D[u]

76. Строка представляет собой

a) конечную линейно-упорядоченную последовательность простых данных символьного типа
b) конечную последовательность простых данных символьного типа
c) конечную последовательность простых данных
d) последовательность данных символьного типа

77. Граф, содержащий только ребра, называется

a) ориентированным
b) неориентированным
c) простым
d) связным

78. Граф, содержащий только дуги, называется

a) ориентированным
b) неориентированным
c) простым
d) связным

79. Граф, содержащий ребра и дуги, называется

a) неориентированным
b) простым
c) смешанным
d) связным

80. Путь (цикл), который содержит все ребра графа только один раз, называется

a) Эйлеровым
b) Гамильтоновым
c) декартовым
d) замкнутым

5.12 Вопросы к экзамену

1. Концепция типа данных: основные принципы концепции. Абстрактные типы данных.
2. Структуры с индексированием. Поиск и выборка в структурах с индексированием. Анализ сложности. Усечение структур с индексированием.
3. Понятие сложности алгоритмов. Анализ сложности алгоритмов. Классы входных данных. Примеры. Асимптотические обозначения скоростей роста (с примерами).
4. Метод «разделяй и властвуй». Задача об умножении двух n-битовых чисел. Оценка сложности. Оценка сложности данной задачи.
5. Основная теорема. Правила вычисления времени выполнения программ: сумм, произведений, условных операторов.
6. Алгоритмы сортировки: основные понятия, параметры оценки. Примеры видов внутренней и внешней сортировки. Теорема о высоте любого разрешающего дерева, сортирующего n элементов.
7. Алгоритмы сортировки: вставками, выбором, пузырьком (классическим). Изложить суть, время работы, пример.
8. Алгоритмы сортировки: шейкер-сортировка; корневая; Шелла. Изложить суть, время работы, пример.
9. Алгоритмы сортировки: прямого слияния; естественного слияния; многофазной сортировки. Изложить суть, время работы, пример и псевдокод сортировок.

10. Алгоритмы сортировки: слиянием; быстрая, пирамидальная. Изложить суть, время работы, пример.
11. Алгоритмы проверки ацикличности графа; проверка связности графа; обхода графа в глубину. Изложить суть, время работы, пример.
12. Алгоритмы поиска МОД Дейкстры-Прима, определение максимального количества различных МОД данного графа. Изложить суть, время работы, пример.
13. Алгоритм топологической сортировки графа. Изложить суть, время работы, пример.
14. Муравьиный алгоритм. Изложить суть, время работы, пример.
15. Алгоритмы нахождение кратчайших путей в невзвешенном графе, во взвешенном графе с неотрицательными весами. Изложить суть, время работы, пример.
16. Алгоритм определение компонент сильной связности графа. Изложить суть, время работы, пример.
17. Определение понятий: хеширование, хеш-функция, хеш-таблица, ключ, хеш-значение, коллизия. Требования к хеш-функции.
18. Классификация методов создания хеш-функции: описание каждого метода, код, требования к размеру хеш-таблицы и другим параметрам при реализации каждого метода.
19. Классификация методов разрешения коллизий при хешировании, описание, пример и анализ каждого метода. Псевдокоды вставки, удаления и поиска элементов при хешировании методом открытой адресации и методом цепочек.
20. Переполнение хеш-таблицы, рехеширование. Оценка качества хеш-функции.
21. Хеширование по вторичным ключам.
22. Деревья: определение, виды бинарных деревьев, способы представление деревьев в памяти. Построение бинарного дерева (код).
23. Способы обхода бинарных деревьев: описание и пример всех обходов. Исходный код обходов.
24. Построение бинарного дерева поиска (БДП). Вставка, удаление, поиск заданного элемента и поиск следующего по величине элемента в БДП. Описание и код.
25. Сортировка с прохождением бинарного дерева: описание, пример, код.
26. АВЛ-деревья: определение, свойства, применение. Одинарный и двойной повороты. Алгоритмы АВЛ вставки и удаления. Оценка сбалансированности и производительности АВЛ-деревьев. Оценка сбалансированных деревьев. Оценка производительности АВЛ-деревьев.
27. Сжатие информации с помощью деревьев. Построение префиксного кода.
28. Определение NP-класса и класса P. Определение задачи поиска. Примеры NP-задач и соответствующих им P-задач. Определение NP-полной задачи. Примеры NP-полных задач.
29. Типичные NP-задачи: постановка задачи о коммивояжере, об упаковке рюкзака, о раскраске графа, о планировании работ, о раскладке по ящикам.
30. Жадные приближенные алгоритмы решения NP-задач. Приближения в задачах о коммивояжере, об упаковке рюкзака, о раскраске графа, о планировании работ, о раскладке по ящикам.