Федеральное государственное автономное образовательное учреждение высшего образования "Национальный исследовательский университет "Высшая школа экономики"

Факультет компьютерных наук
Департамент программной инженерии

Рабочая программа дисциплины

Алгоритмы и структуры данных (часть 1 | модули 1-2)

для образовательной программы «Программная инженерия»
направления подготовки 09.03.04 «Программная инженерия»
уровень - бакалавр

Разработчик программы
Дегтярев К.Ю., к.т.н., доцент, kdegtiarev@hse.ru

Одобрена на заседании департамента программной инженерии « ___ » _____ 2018 г.

Руководитель департамента Авдошин С.М. _____

Рекомендована Академическим советом образовательной программы

«___»_____ 2018 г., № протокола _____

Утверждена « ___ » _____ 2018 г.

Академический руководитель образовательной программы

Шилов В.В. _____

Москва, 2018

1 Область применения и нормативные ссылки

Настоящая программа учебной дисциплины «Алгоритмы и структуры данных» (часть 1, модули 1-2) устанавливает минимальные требования к знаниям и умениям студента и определяет содержание и виды учебных занятий и отчетности.

Программа предназначена для преподавателей, ведущих данную дисциплину (лекции и практические занятия), учебных ассистентов и студентов направления 09.03.04 «Программная инженерия» подготовки бакалавра, изучающих дисциплину «Алгоритмы и структуры данных» в первом семестре (модули 1 и 2) 2018-2019 учебного года.

Программа дисциплины разработана в соответствии с:

- федеральным государственным образовательным стандартом (ОС) высшего образования Федерального государственного автономного образовательного учреждения высшего профессионального образования «Национальный исследовательский университет «Высшая школа экономики» по направлению 09.03.04 «Программная инженерия» - бакалавриат (действует с 2017 г.,), разработанным в соответствии с Федеральным законом №273-ФЗ «Об образовании в Российской Федерации»; http://www.hse.ru/standards/standard,
- образовательной программой (базовым учебным планом, составленным на основе ОС ПИ) направления 09.03.04 «Программная инженерия» подготовки бакалавра;
- рабочим учебным планом (для 2-го года обучения; 2018-2019 учебный год) по направлению 09.03.04 «Программная инженерия» подготовки бакалавра, утвержденным 28.02.2018 (http://asav.hse.ru/plans.html?login=web&password=web).

2 Цели освоения дисциплины

Цели освоения дисциплины «*Алгоритмы и структуры данных*» (**часть 1**, модули 1-2):

- формирование у студентов профессиональных компетенций, связанных с использованием теоретических знаний в области структур данных и теории алгоритмов, пониманием концепции абстрактных типов данных (АТД) и подходов к их реализации на языке C++ на основе принципов объектно-ориентированного построения программ, оценки влияния выбора структур данных и/или алгоритмов на производительность (быстродействие/эффективность) программ;
- получение практических навыков решения задач с использованием разных структур данных (напр., линейных списков, стеков, очередей, хэш-таблиц и пр.), используя концепции абстракции данных и модульного программирования;
- развитие умений, основанных на полученных теоретических знаниях, позволяющих на творческом и репродуктивном уровне предлагать и применять эффективные подходы к решению (алгоритмизации) поставленных задач с использованием данных простой и сложной структуры;
- получение студентами навыков *самостоятельной (исследовательской) работы*, предполагающей изучение специфических особенностей работы со структурами данных в рамках разработки подходов (алгоритмов) к решению поставленной задачи, вопросов управления памятью в C++ и использования компонентов стандартной библиотеки шаблонов (STL).

3 Компетенции обучающегося, формируемые в результате освоения дисциплины

В результате освоения дисциплины студент должен:

- знать базовые абстрактные типы (структуры) данных (*контейнеры*), понимать их особенности, применимые операции и методы реализации АТД на языке C++; контейнеры, итераторы и алгоритмы как основные компоненты STL.
- уметь разрабатывать C++ программы, реализующие заданный алгоритм и использующий определенные структуры данных.
- иметь навыки (приобрести опыт) и владеть основами процедурного и объектно-ориентированного программирования на языке C++, работы с шаблонами функций и классов, использования STL как реализованную на языке C++ коллекцию обобщенных (*generic*) структур данных (*контейнеров*) и алгоритмов, методами оценки трудоемкости (сложности) алгоритмов, подходами к измерению времени в программных реализациях алгоритмов решения поставленных задач.

В результате изучения дисциплины студент осваивает (и развивает) следующие профессиональные компетенции:

1. Научно-исследовательская деятельность (*общие*):

— участие в проведении исследований (экспериментов, наблюдений и количественных измерений), связанных с объектами профессиональной деятельности (программными протами, проектами, …, методами и инструментами программной инженерии) в соответствии с утвержденными заданиями и методиками (ПК-1, ПК-3).

Формы и методы обучения, способствующие формированию и развитию компетенции: лекции и практические занятия, консультации и самостоятельная работа студентов (изучение теоретического материала и практическая реализация программ, предлагаемых в качестве основных и дополнительных заданий, подготовка отчетов);

2. Проектная деятельность (*отдельные*):

— участие в проектировании компонентов программного продукта в объеме, достаточном для их конструирования в рамках поставленного задания;

— создание компонент программного обеспечения (кодирование, отладка, модульное и интеграционное тестирование);

— выполнение измерений / рефакторинг кода в соответствии с планом (ИК-9, ИК-10, ИК-11).

Формы и методы обучения, способствующие формированию и развитию компетенции: лекции и практические занятия, консультации и самостоятельная работа студентов (изучение теоретического материала и практическая реализация программ); выполнение домашних заданий; подготовка отчета о выполненной работе (анализ поставленной задачи, предлагаемые варианты решения, особенности реализации и пр.) и программ на языке C++ (кодирование, отладка и тестирование, рефакторинг);

3. Технологическая деятельность:

— освоение и применение средств проектирования, разработки и тестирования программного обеспечения;

— использование типовых методов для контроля, оценки и обеспечения качества программной продукции (ИК-16).

4. Производственная деятельность (*отдельные*):

— участие в процессах разработки программного обеспечения (ИК-17).

Формы и методы обучения, способствующие формированию и развитию компетенции: лекции и практические занятия, консультации и самостоятельная работа студентов (изучение теоретического материала и практическая реализация программ); выполнение домашних заданий; подготовка отчета

о выполненной работе (анализ поставленной задачи, предлагаемые варианты решения, особенности реализации и пр.) и программ на языке C++ (кодирование, отладка и тестирование, рефакторинг).

4 Место дисциплины в структуре образовательной программы

Настоящая дисциплина относится к блоку профессионального цикла (Major) дисциплин, базовой части (**Б.ПЦ.Б**) рабочего учебного плана направления 09.03.04 «Программная инженерия» подготовки бакалавра (ОП «Программная инженерия», 2018-2019 учебный год, факультет компьютерных наук) – см. http://asav.hse.ru/plans.html?login=web&password=web.

В соответствии с рабочим учебным планом (РУП) на 2018-2019 учебный год, дисциплина *«Алгоритмы и структуры данных»* (часть 1) предлагается студентам 2-го курса бакалавриата в 1-м и 2-м модулях учебного года. Количество аудиторных часов – **56** (28 + 28; на 7 и 8 учебных недель модулей 1 и 2, соответственно); эти часы делятся между лекциями и практическими занятиями следующим образом: 16Л+12Пр (модуль 1) и 14Л+14Пр (модуль 2). Общее количество часов, отведенных студентам для самостоятельной работы – **82** (половина от того количества часов, которое предусмотрено РУП для двух частей 4-х модульной дисциплины «Алгоритмы и структуры данных» в 2018-2019 учебном году). Текущий контроль (первое полугодие, модули 1 и 2) – контрольные работы, проводимые в течение каждого из 2-х модулей. Итоговый контроль – **экзамен** в конце 2-го модуля. Общее число часов (часть 1 дисциплины) равняется **138**.

Изучение дисциплины базируется на знаниях, полученных студентами при освоении учебных дисциплин «Дискретная математика» и «Программирование» первого года обучения, основ программирования в части базовых алгоритмических конструкций. Таким образом, «Алгоритмы и структуры данных» (часть 1) можно рассматривать в качестве одной из важных составляющих цепочки предлагаемых в рамках БУП направления 09.03.04 дисциплин, связанных с теоретической информатикой, основами информационных технологий и программирования (первых двух лет обучения).

Дисциплина является основой для последующего изучения таких дисциплин как «Проектирование и архитектура программных систем», «Обеспечение качества и тестирование» (3-й курс), отдельных научно-исследовательских семинаров (3-й и 4-й курсы | цикл **Б.ПД** ‘НИС’), а также выполнения курсовых, проектных и выпускных работ, предусмотренных БУП по направлению подготовки 09.03.04.

●●● **Примечание**: *Как мы знаем, алгоритмы решают возникающие повсеместно вычислительные задачи; естественно, мы всегда требуем от алгоритма получения правильного решения поставленной задачи. Разработка алгоритма является весьма трудоемкой задачей, требующей одновременного мониторинга соответствия предлагаемых **действий** сформулированным **целям** (верхний уровень представления проблемы) и отдельных мелких шагов (нижний уровень представления проблемы), образующих в совокупности этапы действий, которые и приводят в конечном итоге к решению задачи. Для описания последовательности шагов алгоритма обычно используются три формы его описания, а именно:*

- ***вербальная** (словесная) **форма**, которая может грешить неточностью и/или нечеткостью формулировок, серьезно влияющих на понимание представляемых действий,*
- ***псевдокод** (форма представления с «облегченным» синтаксисом, использующий смесь конструкций обычного (повседневного) языка и языка программирования) – как образно отмечено Стивеном Скиеной (**Steven Sol Skiena**, Teaching Professor of Computer Science at Stony Brook University), псевдокод – это «язык программирования, который никогда не выдает сообщений о синтаксических ошибках»,*
- ***язык программирования** (описание алгоритма по жестким синтаксическим правилам выбранного языка программирования, позволяющим достичь точного описания реализуемых шагов).*

Последняя из перечисленных форм описания шагов АЛГОРИТМА представляется наиболее интересной и практически значимой, поскольку современная жизнь связана с повседневным использованием компьютеров (вычислительных устройств), на которых работают многочисленные и разнообразные реализующие алгоритмы программы. Компьютеры призваны существенно ускорить выполняемые вычисления, хранить и обрабатывать большие объемы информации (ДАННЫХ), которые представляют собой ***абстракцию*** *связанной с рассматриваемой проблемой реальности. Такие абстракции (по сути, модели предметной области) дают возможность описать данные, которыми оперируют алгоритмы.*

Для обработки данных нам следует определить типы (данных) и множество операций, которые выполняются над этими данными, при этом детали того, как реализованы эти операции либо неизвестны, либо скрываются. Таким образом, имеющийся набор (коллекция) данных, понимание специфики отношений между этими данными и набора применяемых к данным действий (операций) без уточнения деталей их реализации приводит к появлению важной концепции ***абстракции*** *(знания того,* ***ЧТО*** *можно сделать с соответствующим типом данных, но не* ***КАК*** *это делается с точки зрения «неуточненной» реализации (деталей)). В частности, в основе библиотеки языка C++ лежит обширная и достаточно сложная* ***стандартная библиотека шаблонов (STL)****, являющаяся «библиотекой универсальных компонентов для управления коллекциями данных с помощью эффективных алгоритмов»* (*Н.Джосаттис. Стандартная библиотека C++. Справочное руководство, 2-е изд., «И.Д.Вильямс», 2014*). *В языках программирования абстракция часто «проявляется посредством пользовательских типов данных; … для того, чтобы использовать абстрактный тип данных (****АТД****), необходимо понимать его интерфейс, в значительной степени игнорируя детали реализации»* (*C.Allison. C&C++ Code Capsules: A Guide for Practitioners, Prentice Hall PTR, 1998*). *В своей работе программисты сосредотачивают свое внимание на тех возможностях, которые предоставляет класс, а не на «деталях реализации класса, т.е. они следуют* ***принципу абстракции данных****: пользовательский код не должен иметь доступа к деталям реализации используемых классов»* (*W.J.Collins Data Strucutures and the Standard Template Library, McGraw-Hill, 2003*). *Классы в языке C++ обеспечивают «реализацию АТД (с достаточной степенью сокрытия информации), и любая часть программы, предусматривающая выполнение операции над АТД, сможет это сделать посредством вызова соответствующего метода класса»* (*M.A.Weiss. Data Structures and Algorithms Analysis in C++, 3rd ed., Pearson Educ., 2006*).

5 Приблизительный тематический план учебной дисциплины

(учитывая тот факт, что в 2018-2019 учебном году содержание части 1 дисциплины «Алгоритмы и структуры данных» претерпевает некоторые изменения и дается в обновленном формате, то возможны некоторые отклонения от заявленного в плане количества часов, отведенных под обсуждение/объяснение отдельных тем)

№	Название темы	Всего часов по дисциплине	Аудиторные часы		Самостоятельная работа
			Лекции	Практические занятия	
Модуль 1 (28 часов / 16 + 12)	01 сентября – 21 октября **2018 г.** (**зачетная неделя**: 22 – 28 октября)				
1	*Вводные замечания*. Что такое решение задачи? Понятие абстракции, абстрактного типа данных (АТД). Спецификация АТД. Типы данных, структуры данных и алгоритмы. Реализация АТД (пример вектора). Объектно-ориентированное программирование и C++. Классы, шаблоны. АТД «Стек», стек как пример контейнера, примеры реализации; системный стек, запись активации. Персистентный стек. Мультистек. Асимптотические оценки сложности. Амортизационный анализ	21	5	4	12
2	Структуры данных и стандартная библиотека шаблонов (STL) – *введение*. Контейнер `std::vector` (модель динамического массива). Динамическое выделение памяти. АТД "Список" (List). Свойства операций над АТД "Список". Связные списки (односвязные, двусвязные, *xor*-связные списки и Y-образные списки). Примеры. Многопотоковые связные списки. Связные списки с циклами (обсуждение алгоритмов). Персистентные структуры данных. Элементы управления памятью. Шаблонные классы `std::array` и `std::forward_list`. Комментарии.	21	5	4	12

3	АТД "Очередь" (Queue) и "Дек" (очередь с двумя концами / Deque). Особенности реализации дека (двухуровневая модель дека); дек на двух стеках. Шаблонный класс `std::deque` и адаптер `std::stack`. Приоритетная очередь. Двоичная куча (ДК); представление ДК в виде массива. Свойства кучи, поиск элемента.	24	6	4	14
• **Итого *(МОДУЛЬ 1)*:**		**66**	**16**	**12**	**38**
Модуль 2 (28 часов / 14 + 14)	29 октября – 19 декабря **2018 г.** (**зачетная неделя**: 20 – 31 декабря)				
4	Кольцевой (циклический буфер), библиотека `Boost`, контейнер `boost::circular_buffer` (кольцо). Задача Иосифа Флавия (Josephus problem). Упорядоченные множества. Двоичные деревья, представление двоичных деревьев. АТД "Дерево поиска" (Search tree), бинарное дерево поиска (BST), 2-3 деревья, 2-3-4 деревья, B-деревья, дерево отрезков; примеры и комментарии.	35	9	6	20
5	Хэширование; разрешение коллизий. Хэш-таблицы в стандартной библиотеке. Качество хэш-функции; заполнение хэш-таблицы значениями с использованием заданных хэш-функций и подходов; коалиционное хэширование. Метод Брента, хэширование на основе кодов Грея (Gray code chaining).	23	5	4	14
6	Дополнительные темы *(опция)*	14	-	4	10
• **Итого *(МОДУЛЬ 2)*:**		**72**	**14**	**14**	**44**
• **Итого (*МОДУЛИ 1 и 2*):**		**138**	**30**	**26**	**82**

6 Формы контроля знаний студентов

Тип контроля	Форма контроля	модули		Параметры
		#1	#2	
Текущий	контрольные работы (КР)	☑ контрольные работы, проводимые в течение модуля	☑ контрольные работы, проводимые в течение модуля	Подготовка программ в соответствии с предложенным заданием
Итоговый	Экзамен	☒	☑ после 20-го декабря 2018 г.	Теоретический (письменный) экзамен

6.1 Критерии оценки знаний и навыков

Оценки по всем формам контроля выставляются по 10-ти балльной шкале.

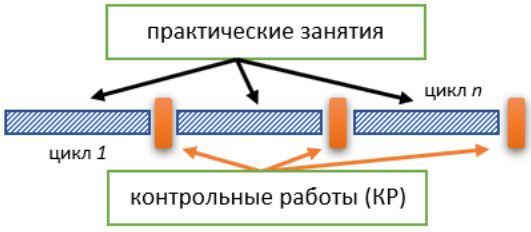
Текущий контроль предусматривает выполнение нескольких контрольных работ (КР) в течение модулей 1-2. Подготовленный студентом код (проект) представляется на проверку до истечения *установленного срока*. Детали, связанные с проведением таких контрольных работ, объясняются на практических занятиях 1-го модуля. Работы, представленные позже объявленного студентам срока, не проверяются – в этом случае, за соответствующую работу автоматически выставляется оценка «0» (ноль).

Итоговый контроль: экзамен в конце 2-го модуля (теоретические вопросы); экзамен проводится в письменной форме (альтернатива: в виде теста).

6.2 Порядок формирования оценок по дисциплине

По всем видам работ оценка выставляется по принятой в НИУ ВШЭ 10-балльной шкале.

В рамках первой части дисциплины (модули 1-2) предусматривается оценивание контрольных работ (**КР**) студентов. Необходимые объяснения, касающиеся деталей заданий, предъявляемых требований и времени проведения КР, даются преподавателем на практических занятиях. В оценке (результате проверки работ) учитываются оригинальность предложенного решения, правильность и полнота выполнения заданий, стиль оформления кода и пр. Соответствующие КР будут завершать «циклы» практических занятий (циклы могут иметь разную длину), т.е.



Задания, рассматриваемые на **практических занятиях**, студенты будут выполнять, по большей части, дома – для каждого такого задания (**ДЗ**) устанавливается определенный крайний срок, до истечения которого выполненные работы должны быть представлены на проверку. Пояснения относительно деталей оценивания и выделяемого для завершения каждого задания времени даются преподавателем на практических занятиях.

КР имеют разные веса ($w_i, i = \overline{1,n}$), которые распространяются и на задания практических занятий соответствующего «цикла». Если $O_{КР}^{(i)}$ – оценка, полученная студентом за КР i-го цикла, а

$O^{(i)}_{практ}$ – оценка, полученная за выполненные задания (ДЗ) практических занятий i-го цикла (последняя оценка получается, исходя из сложности отдельных заданий, и формула получения этой оценки объясняется студентам заранее), то ***результирующая (итоговая) оценка*** за практические занятия первой части дисциплины (модули 1-2) вычисляется по следующей формуле: $P_{ИТОГ} = \sum_{i=1}^{n} w_i \cdot \left(0,7 \cdot min(O^{(i)}_{практ}, O^{(i)}_{КР}) + 0,3 \cdot max(O^{(i)}_{практ}, O^{(i)}_{КР})\right)$, где n – число КР, проводимых в течение первых двух модулей. При выполнении работ студенты могут пользоваться своими компьютерами (ноутбуками) – это относится как к контрольным работам (КР), так и ко всем практическим занятиям в течение 1-го и 2-го модулей дисциплины.

Как было отмечено выше, все ***основные детали*** и ***правила организации работы*** в рамках практических занятий по данной дисциплине будут объяснены преподавателем, проводящим занятия в подгруппах. Посещаемость практических занятий ($O_{ПЗ}$) отмечается в рабочей ведомости преподавателя (печатный или электронный вариант) и учитывается в итоговой оценке (конец 2-го модуля). Эта оценка вычисляется как отношение $10 * present/total$, где $present$ – число посещенных занятий, $total$ – общее число практических занятий в течение 1-го и 2-го модулей.

Практические занятия планируется начать со 2-ой недели первого модуля, т.е. с ***10-го сентября 2018 г.*** (см. расписание учебных занятий - https://www.hse.ru/ba/se/timetable).

[Накопление] С учетом $O_{ПЗ}$, накопленная оценка студента вычисляется следующим образом:

$$O_{накопленная} = P_{ИТОГ} + 0,07 \cdot O_{ПЗ}$$

{Опция} Презентации **Pr** продолжительностью до 10÷12 минут по заранее согласованным темам, сделанные студентами во время лекционных часов, могут дополнительно дать до 0.3 балла к вычисляемому значению $O_{наколенная}$.

- Примечание: если в результате вычислений накопленная оценка $O_{накопленная}$ окажется больше 10, то она «урезается» до значения 10.0, которое и используется в последующих расчетах.

Для проставления в ведомость (конец 2-го модуля) накопленная оценка $O_{накопленная}$ округляется до ближайшего целого (по правилам НИУ ВШЭ) – в результате получается $O_{накоп[ROUND]}$. Студент может претендовать на получение оценки «автомат» (оценка *«отлично»*, т.е. 8, 9 или 10 по шкале, принятой в НИУ ВШЭ) по первой части дисциплины, исходя из накопленной оценки, при условии, что $round(P_{ИТОГ}) \geq 8$ и $O_{ПЗ} \geq 8$ (в этом случае, в ведомость студенту выставляется итоговая оценка в виде округленного накопления $O_{накоп[ROUND]}$).

[Результат] Итоговая оценка (конец второго модуля учебного года) вычисляется на основе полученной оценки накопления (с округлением) $O_{накоп[ROUND]}$ и оценки за экзамен $O_{экзамен}$ (целое число) в виде $O_{результат} = 0,6 \cdot O_{накоп[ROUND]} + 0,4 \cdot O_{экзамен}$. Результирующая оценка $O_{экзамен}$ первой части (модули 1 и 2) дисциплины – результат округления до ближайшего целого.

В случае получения неудовлетворительной оценки и повторной сдачи (т.е. пересдачи) экзамена формула вычисления оценки студента не меняется.

Таблица соответствия оценок по десятибалльной и пятибалльной системам

По десятибалльной шкале	По пятибалльной шкале
1 – неудовлетворительно **2** – очень плохо **3** – плохо	неудовлетворительно – **2**
4 – удовлетворительно **5** – весьма удовлетворительно	удовлетворительно – **3**
6 – хорошо **7** – очень хорошо	хорошо – **4**
8 – почти отлично **9** – отлично **10** – блестяще	отлично – **5**

7 Содержание дисциплины (см. стр. 6-7 программы)

(возможны расширения/сужения отдельных обсуждаемых тем)

7.1 Содержание лекций

Тема 1. *Вводные замечания*. Что такое решение задачи? Понятие абстракции, абстрактного типа данных (АТД). Спецификация АТД. Типы данных, структуры данных и алгоритмы. Реализация АТД (пример вектора). Объектно-ориентированное программирование и C++. Классы, шаблоны. АТД «Стек», стек как пример контейнера, примеры реализации; системный стек, запись активации. Персистентный стек. Мультистек. Асимптотические оценки сложности O-большое и Ω-большое. Амортизационный анализ

Литература (источники информации):

1. Weiss M.A. Data Structures and Algorithm Analysis in C++, 3rd ed., Pearson Education, 2006, 586 p. (4th edition of the book was released in 2014) — **русский перевод книги**: Уайс М.А. Организация структур данных и решение задач, пер. с англ., М.: Эком, 2009, 896 с. | M.A.Weiss Homepage - http://users.cis.fiu.edu/~weiss/
2. Collins W.J. Data Structures and the Standard Template Library, McGraw-Hill HE, 2003, 688 p.
3. Кубенский А.А. Структуры и алгоритмы обработки данных: объектно-ориентированный подход и реализация на C++, уч. пособие, СПб.: БХВ-Петербург, 2004, 465 с.
4. Ахо А. В. Хопкрофт Дж.Э., Ульман Дж. Д. Структуры данных и алгоритмы, уч. пособие, ООО «И.Д. Вильямс», 2010, 398 с.
5. Кормен Т., Лейзерсон Ч., Ривест Р., Штайн К. Алгоритмы: построение и анализ, 2-е изд., пер. с англ., СПб.: ООО «И.Д. Вильямс», 2011., 1296 с.
6. Бабенко М.А., Левин М.В. Введение в теорию алгоритмов и структур данных, 2-е изд. (испр.), М.: МЦНМО, 2014, 140 с.

Тема 2. Структуры данных и стандартная библиотека шаблонов (STL) – *введение*. Контейнер `std::vector` (модель динамического массива). Динамическое выделение памяти. АТД "Список" (List). Свойства операций над АТД "Список". Связные списки (односвязные, двусвязные, *xor*-связные списки и Y-образные списки). Примеры. Многопотоковые связные списки. Связные списки с циклами (обсуждение алгоритмов). Персистентные структуры данных. Элементы управления памятью. Шаблонные классы `std::array` и `std::forward_list`. Комментарии.

Литература (источники информации):

1. Weiss M.A. Data Structures and Algorithm Analysis in C++, 3rd ed., Pearson Education, 2006, 586 p. (4th edition of the book was released in 2014) — **русский перевод книги**: Уайс М.А. Организация структур данных и решение задач, пер. с англ., М.: Эком, 2009, 896 с. | M.A.Weiss Homepage - http://users.cis.fiu.edu/~weiss/
2. Кормен Т., Лейзерсон Ч., Ривест Р., Штайн К. Алгоритмы: построение и анализ, 2-е изд., пер. с англ., СПб.: ООО «И.Д. Вильямс», 2011., 1296 с.
3. Джосаттис Н.М. Стандартная библиотека C++. Справочное руководство, 2-е изд., пер. с англ., ООО «И.Д. Вильямс», 2014, 1136 с. — **английский оригинал книги**: Josuttis N.M. The C++ Standard Library: A Tutorial and Reference, 2nd ed., Addison-Wesley Professional, 2012, 1128 p.
4. Бабенко М.А., Левин М.В. Введение в теорию алгоритмов и структур данных, 2-е изд. (испр.), М.: МЦНМО, 2014, 140 с.
5. Прата С. Язык программирования C++. Лекции и упражнения, 6-е изд., пер. с англ., серия Landmark, ООО «И.Д. Вильямс», 2012, 1248 с.
6. Standard C++ Library Reference, http://www.cplusplus.com/reference/

Тема 3. АТД "Очередь" (Queue) и "Дек" (очередь с двумя концами / Deque). Особенности реализации дека (двухуровневая модель дека); дек на двух стеках. Шаблонный класс `std::deque` и адаптер `std::stack`. Приоритетная очередь. Двоичная куча (ДК); представление ДК в виде массива. Свойства кучи, поиск элемента.

Литература (источники информации):

1. Kieras D. Using C++11's Smart Pointers (Course EECS381 "Object-oriented and Advanced Programming), 2014, http://www.umich.edu/~eecs381/handouts/C++11_smart_ptrs.pdf
2. Липпман С.Б., Лажойе Ж., Му Б. Язык программирования C++. Базовый курс, пер. с англ., 5-е изд., ООО «И.Д.Вильямс», 2014, 1120 с.
3. Седжвик Р. Алгоритмы на C++. Анализ, структуры данных, поиск, сортировка, алгоритмы на графах, ООО «И.Д.Вильямс», 2011, 1056 с.
4. Weiss M.A. Data Structures and Algorithm Analysis in C++, 3rd ed., Pearson Education, 2006, 586 p. (4th edition of the book was released in 2014) — **русский перевод книги**: Уайс М.А. Организация структур данных и решение задач, пер. с англ., М.: Эком, 2009, 896 с. | M.A.Weiss Homepage - http://users.cis.fiu.edu/~weiss/
5. Гагарина Л.Г., Колдаев В.Д. Алгоритмы и структуры данных, М.: Финансы и статистика (Инфра-М), 2009, 303 с.
6. Кормен Т., Лейзерсон Ч., Ривест Р., Штайн К. Алгоритмы: построение и анализ, 2-е изд., пер. с англ., СПб.: ООО «И.Д. Вильямс», 2011., 1296 с.
7. Langsam Y., Augenstein M.J., Tenenbaum A.M. Data Strucutures Using C and C++, 2nd ed., Prentice-Hall, 1996
8. Скиена С. Алгоритмы. Руководство по разработке, 2-е изд., пер. с англ., СПб.: БХВ-Петер-бург, 2011, 720 с. — **английский оригинал книги**: Skiena S.S. The Algorithm Design Manual, 2nd ed., Springer Verlag, 2008, 728 p.

Тема 4. Кольцевой (циклический буфер), библиотека `Boost`, контейнер `boost::circular_buffer` (кольцо). Задача Иосифа Флавия (Josephus problem). Упорядоченные множества. Двоичные деревья, представление двоичных дере-вьев. АТД "Дерево поиска" (Search tree), бинарное дерево поиска (BST), 2-3 деревья, 2-3-4 деревья, B-деревья, дерево отрезков; примеры и комментарии.

Литература (источники информации):

1. Джосаттис Н.М. Стандартная библиотека C++. Справочное руководство, 2-е изд., пер. с англ., ООО «И.Д. Вильямс», 2014, 1136 с. — **английский оригинал книги**:

Josuttis N.M. The C++ Standard Library: A Tutorial and Reference, 2nd ed., Addison-Wesley Professional, 2012, 1128 p.
2. Weiss M.A. Data Structures and Algorithm Analysis in C++, 3rd ed., Pearson Education, 2006, 586 p. (4th edition of the book was released in 2014) — **русский перевод книги**: Уайс М.А. Организация структур данных и решение задач, пер. с англ., М.: Эком, 2009, 896 с. | M.A.Weiss Homepage - http://users.cis.fiu.edu/~weiss/
3. Гагарина Л.Г., Колдаев В.Д. Алгоритмы и структуры данных, М.: Финансы и статистика (Инфра-М), 2009, 303 с.
4. Collins W.J. Data Structures and the Standard Template Library, McGraw-Hill HE, 2003, 688 p.
5. Седжвик Р. Алгоритмы на C++. Анализ, структуры данных, поиск, сортировка, алгоритмы на графах, ООО «И.Д.Вильямс», 2011, 1056 с.
6. Goodrich M.T., Tamassia R., Mount D.M. Data Structures and Algorithms in C++, 2nd ed., Wiley&Sons, 2011, 735 p.
7. Бабенко М.А., Левин М.В. Введение в теорию алгоритмов и структур данных, 2-е изд. (испр.), М.: МЦНМО, 2014, 140 с.
8. Boost C++ Libraries - http://www.boost.org/
9. Обзор библиотеки Boost - https://www.youtube.com/watch?v=cEZgrioRkic

Тема 5. Хэширование; разрешение коллизий. Хэш-таблицы в стандартной библиотеке. Качество хэш-функции; заполнение хэш-таблицы значениями с использованием заданных хэш-функций и подходов; коалиционное хэширование. Метод Брента, хэширование на основе кодов Грея (Gray code chaining).

Литература (источники информации):

1. Weiss M.A. Data Structures and Algorithm Analysis in C++, 3rd ed., Pearson Education, 2006, 586 p. (4th edition of the book was released in 2014) — **русский перевод книги**: Уайс М.А. Организация структур данных и решение задач, пер. с англ., М.: Эком, 2009, 896 с. | M.A.Weiss Homepage - http://users.cis.fiu.edu/~weiss/
2. Кормен Т., Лейзерсон Ч., Ривест Р., Штайн К. Алгоритмы: построение и анализ, 2-е изд., пер. с англ., СПб.: ООО «И.Д. Вильямс», 2011., 1296 с. | Кормен Т.Х. Алгоритмы. Вводный курс, пер. с англ., ООО «И.Д.Вильямс», 2014, 208 с.
3. Гагарина Л.Г., Колдаев В.Д. Алгоритмы и структуры данных, М.: Финансы и статистика (Инфра-М), 2009, 303 с.
4. Седжвик Р. Алгоритмы на C++. Анализ, структуры данных, поиск, сортировка, алгоритмы на графах, ООО «И.Д.Вильямс», 2011, 1056 с.
5. Бабенко М.А., Левин М.В. Введение в теорию алгоритмов и структур данных, 2-е изд. (испр.), М.: МЦНМО, 2014, 140 с.
6. Джосаттис Н.М. Стандартная библиотека C++. Справочное руководство, 2-е изд., пер. с англ., ООО «И.Д. Вильямс», 2014, 1136 с. — **английский оригинал книги**: Josuttis N.M. The C++ Standard Library: A Tutorial and Reference, 2nd ed., Addison-Wesley Professional, 2012, 1128 p.
7. Goodrich M.T., Tamassia R., Mount D.M. Data Structures and Algorithms in C++, 2nd ed., Wiley&Sons, 2011, 735 p.
8. Chen W.-C., Vitter J.S. Analysis of Early-Insertion Standard Coalesced Hashing / SIAM Journal of Computation, 12-**4**, 1983, pp.667-681.
9. Loeb M., Tharp A.L. Gray Code Chaining: A High Performance Hashing Algorithm for Storage Applications / IEEE Int. Conference on Information Technology (ITNG'07), 2007.

7.2 Содержание практических занятий

Темы практических занятий в рамках дисциплины *«Алгоритмы и структуры данных»* (**часть 1**) определяются материалом, рассматриваемым на лекционных занятиях дисциплины и предлагаемым студентам для самостоятельного рассмотрения (изучения). В частности, на практических занятиях могут быть рассмотрены следующие варианты заданий (общее описание тем и направлений, которое может быть дополнено и/или изменено в процессе реализации дисциплины за счет модификации некоторых заданий, появления новых заданий и пр.):

(1) работа с одно- (*singly*) и двусвязными списками (*doubly linked lists*); двусвязные списки с ограничителями; процедуры *INSERT*, *DELETE*, *UNION* и др. применительно к динамическим структурам; поиск в отсортированном компактном списке,

(2) представление списка с помощью нескольких массивов, представление с помощью одного массива; выделение и освобождение памяти для элементов двусвязного списка, представленного с помощью нескольких массивов; простой менеджер памяти, основанный на использовании связных списков,

(3) реализация очереди с помощью двух стеков, анализ времени выполнения операций с элементами очереди; реализация стека с помощью двух очередей, анализ времени выполнения операций с элементами стека; работа со стеком: принятие решений (*backtracking*); обращение (*reverse*) содержимого стека и пр.; кольцевая очередь (*circular queue*),

(4) двоичные деревья поиска с одинаковыми ключами; задача планирования (запрашиваемого) времени посадки самолетов (отсортированный массив, отсортированный список, дерево двоичного поиска); оценка времени размещения нового элемента в структуре,

(5) качество хэш-функции; таблицы с прямой адресацией; хэш-таблицы; разрешение коллизий при помощи цепочек; хэширование с открытой адресацией; совершенное (идеальное) хэширование; заполнение хэш-таблицы значениями с использованием заданной хэш-функции и различных подходов (*probing*) и др.

8 Образовательные технологии

Работа на практических занятиях (ПР) предполагает *самостоятельную* разработку программ. Языком программной реализации алгоритмов является язык программирования C++ (при проведении практических занятий в компьютерных классах используется интегрированная среда разработки Microsoft Visual Studio). На лекционных занятиях минимально рассматриваются лишь те элементы языка и стандартной библиотеки, которые необходимы для представления материала лекций. Выполнение заданий, предлагаемых на практических занятиях, предполагает, в том числе, и *самостоятельное изучение* материала, связанного с самим языком программирования и его библиотеки. Ссылки на сопроводительные и рекомендуемые материалы размещаются на странице дисциплины в LMS.

9 Оценочные средства для текущего контроля и аттестации студента

9.1 Тематика заданий текущего контроля

9.1.1. Контрольная работа

Примерное задание (шаблон) описано в программе дисциплины *«Алгоритмы и структуры данных»* (**часть 1**). Там же приведено и краткое описание подхода к выставлению оценки за контрольную работу (*оставлено без изменений* — см. **стр.13** рабочей программы дисциплины «Алгоритмы и структуры данных» (**часть 1**) 2016-2017 учебного года), а именно:

1) Алгоритм реализован верно — **4 балла** (из 10 возможных),
2) Верно выполняется тестовый пример, показанный в описании задания — **5 баллов**,
3) Тестовый пример выполняется не на всех входных данных — (+ до 2 баллов) - **6-7 баллов**.
4) Тестовый пример выполняется на всех входных данных (+ до 2 баллов) - **8-9 баллов**.

9.2 Отдельные примерные базовые вопросы для оценки качества освоения дисциплины

1. Как бы вы определили понятие «структура данных»?
2. Как бы вы определили понятие «абстракция данных»? Что такое «абстракция» и в чем состоит ее важность (значимость). Приведите пример абстрактного типа данных. Сформулируйте и кратко прокомментируйте преимущества использования АТД,
3. Что собой представляет динамическая структура данных?
4. Определите АТД для символьных строк (АТД должен включать типичные функции, которые применяются к строкам; каждую функцию определите в терминах входных и выходных данных). Предложите две разные реализации строкового типа,
5. Что такое дек? Что такое стек? Объясните, чем отличается стек от очереди. Как выглядит пользовательский класс стека, очереди, очереди с приоритетом (STL)?
6. Реализуйте стек с использованием односвязного списка; операции `push` и `pop` должны выполняться за время O(1),
7. Какие из <*предложенных*> структур данных являются индексируемыми структурами?
8. Что такое связный список? Как можно представить разреженную матрицу (*sparse matrix*) с использованием связного списка? Как можно эффективно представить разреженную матрицу с использованием массива? Что можно сказать о сложности хранения (*space complexity*) такого подхода?
9. Предположим, что текущая конфигурация очереди имеет вид: `a b c d`. Сколько понадобится операций извлечения/размещения для получения конфигурации `d c b a`?
10. Как организовать поиск заданного элемента в односвязном списке (SLL)?
11. Какова временна́я сложность операций поиска, размещения и удаления в бинарном дереве поиска?
12. Что собой представляет дерево бинарного поиска (*binary search tree*)?
13. Сколько разных деревьев бинарного поиска можно создать из 4 разных ключей?
14. Какова специфика реализации контейнера ‘кольцо’ (`boost::circular_buffer`) из библиотеки `Boost`? Каковы преимущества такой реализации?
15. Результатом прямого (preorder) обхода дерева бинарного поиска является последовательность 30,20,10,15,25,23,39,35,42. Каков результат концевого (postorder) обхода того же дерева?
16. Какую работу выполняет хэш-функция?
17. Предположим, что задана функция `f(x)=x%7` и используется линейное исследова-ние (*linear probing*) для размещения ключей 37,38,72,48,98,11,56 в таблице с индексацией элементов от 0 до 6. На какой позиции в таблице будет размещен элемент 11?
18. Мы можем предположить, что эффективность хэширования, при котором коллизии разрешаются методом цепочек, можно повысить, если поддерживать списки в упорядоченном состоянии. Как такое изменение алгоритма повлияет на время выполнения успешного поиска, неуспешного поиска, вставки, удаления?
19. Что можно считать главным фактором преимущества B-деревьев перед деревьями бинарного поиска при индексировании отношений баз данных?
20. Покажите, что $0.1 \cdot n \cdot \log(n) - 3000 \cdot n + 5$ есть $O(n \cdot \log(n))$.

10 Учебно-методическое и информационное обеспечение дисциплины

10.1 Базовый учебник — *отсутствует* (обсуждение материала не следует какому-либо одному источнику)

10.2 Основная литература:

1. (на англ. языке) Weiss M.A. Data Structures and Algorithm Analysis in C++, 4th ed., Pearson Education, 2013, 654 p. — **русский перевод книги**: Уайс М.А. Организация структур данных и решение задач (3-е издание), пер. с англ., М.: Эком, 2009, 896 с. | M.A.Weiss Homepage - http://users.cis.fiu.edu/~weiss/
2. Джосаттис Н.М. Стандартная библиотека C++. Справочное руководство, 2-е изд., пер. с англ., ООО «И.Д. Вильямс», 2014, 1136 с. — **английский оригинал книги**: Josuttis N.M. The C++ Standard Library: A Tutorial and Reference, 2nd ed., Addison-Wesley Professional, 2012, 1128 p.
3. Кормен Т., Лейзерсон Ч., Ривест Р., Штайн К. Алгоритмы: построение и анализ, 2-е изд., пер. с англ., СПб.: ООО «И.Д. Вильямс», 2011., 1296 с. | Кормен Т.Х. Алгоритмы. Вводный курс, пер. с англ., ООО «И.Д.Вильямс», 2014, 208 с.
4. Седжвик Р. Алгоритмы на C++. Анализ, структуры данных, поиск, сортировка, алгоритмы на графах, ООО «И.Д.Вильямс», 2011, 1056 с. (2 книги в одной) | Sedgewick R. Algorithms in C++, Parts 1-4: Fundamentals, Data Structures, Sorting, Searching, 3rd ed., Addison-Wesley Professional, 1998, 752 p.
5. (на англ. языке) Sedgewick R., Wayne K. Algorithms, 4th ed., Addison-Wesley Prof., 2011, 990 p.
6. (на англ. языке) Goodrich M.T., Tamassia R., Mount D.M. Data Structures and Algorithms in C++, 2nd ed., Wiley&Sons, 2011, 735 p.

10.3 Дополнительная литература и источники

1. Скиена С. Алгоритмы. Руководство по разработке, 2-е изд., пер. с англ., СПб.: БХВ-Петербург, 2011, 720 с. — **английский оригинал книги**: Skiena S.S. The Algorithm Design Manual, 2nd ed., Springer Verlag, 2008, 728 p.
2. Макконелл Дж. Основы современных алгоритмов, 2-е доп. изд., пер. с англ., М.: Техносфера, 2006, 368 с.
3. Ахо А. В. Хопкрофт Дж.Э., Ульман Дж. Д. Структуры данных и алгоритмы, уч. пособие, ООО «И.Д. Вильямс», 2010, 398 с.
4. Гагарина Л.Г., Колдаев В.Д. Алгоритмы и структуры данных, М.: Финансы и статистика (Инфра-М), 2009, 303 с.
5. Бабенко М.А., Левин М.В. Введение в теорию алгоритмов и структур данных, 2-е изд. (испр.), М.: МЦНМО, 2014, 140 с.
6. Липпман С.Б., Лажойе Ж., Му Б. Язык программирования C++. Базовый курс, пер. с англ., 5-е изд., ООО «И.Д.Вильямс», 2014, 1120 с.
7. Кубенский А.А. Структуры и алгоритмы обработки данных: объектно-ориентированный подход и реализация на C++, уч. пособие, СПб.: БХВ-Петербург, 2004, 465 с.
8. (на англ. языке) Langsam Y., Augenstein M.J., Tenenbaum A.M. Data Strucutures Using C and C++, 2nd ed., Prentice-Hall, 1996.
9. (на англ. языке) Gilberg R.F., Forouzan B.A. Data Structures. A Pseudocode Approach with C, 2nd ed., Thomson-Course Technology, 2005, 736 с.
10. (на англ. языке) Collins W.J. Data Structures and the Standard Template Library, McGraw-Hill HE, 2003, 688 p.

11. (на англ. языке) Forouzan B.A., Mosharraf F. Foundations of Computer Science, 2nd ed., Thomson Learning (Cengage), 2008, 624 p.

10.4 Справочные материалы, словари, энциклопедии, инф. сайты и доп. книги

1. (на англ. языке) Collected Algorithms (ACM), htpp://www.acm.org/calgo
2. TutorialPoint: C++ STL Tutorial, http://www.tutorialspoint.com/cplusplus/cpp_stl_tutorial.htm
3. Контейнеры STL (MSDN), http://msdn.microsoft.com/ru-ru/library/1fe2x6kt.aspx
4. (на русск. языке) Справка по C++, http://ru.cppreference.com/w/ | (на англ. языке) C++ Reference, http://en.cppreference.com/w/
5. (на англ. языке) Standard C++ Library Reference, http://www.cplusplus.com/reference/
6. (на англ. языке) Shaffer C.A. Data Structures and Algorithm Analysis (C++ version/3.2), Virginia Tech, 2010-13, people.cs.vt.edu/shaffer/Book/C++3elatest.pdf
7. Using C++11's Smart Pointers (Course EECS381 "Object-oriented and Advanced Programming Handouts, UM), 2014, http://www.umich.edu/~eecs381/handouts/C++11_smart_ptrs.pdf
8. (на англ. языке) Meyers S. Effective STL: 50 Specific Ways to Improve Your Use of the Standard Template Library, Addison-Wesley Prof., 2001, 286 p.
9. (на англ. языке) Raykar M. Implementing a simple smart pointer in C++, CodeProject, 2006, http://www.codeproject.com/Articles/15351/Implementing-a-simple-smart-pointer-in-c

10.5 Программные средства

Практические занятия проводятся в компьютерных классах с выходом в Интернет и доступом к ресурсам электронной библиотеки (http://library.hse.ru/e-resources/e-resources.htm) НИУ ВШЭ. Предусматривается наличие у каждого студента рабочего места (на практических занятиях студенты могут пользоваться собственными ноутбуками). Практическая работа ориентирована на использование современных интегрированных инструментальных сред разработки.

●●● **Примечание**: для самостоятельной работы (установки на персональных компьютерах/ноутбуках) студенты могут использовать другие C++ IDE – например,
Code::Blocks 17.12 (www.codeblocks.org) – по состоянию на 27.08.2018,
Jetbrains CLion 2018.2 (https://www.jetbrains.com/clion/),
Orwell Dev-C++ 5.11 (http://sourceforge.net/projects/orwelldevcpp/),
CodeLite 12.0 (http://codelite.org) - по состоянию на 27.08.2018, Qt – Open Source (http://www.qt.io) и др.

Используемое программное обеспечение в компьютерных классах:

1. Microsoft Office Professional 2010-2013 (или, более поздние версии),
2. MS Visual Studio 2015 (или более поздние версии). Во всех подгруппах используется единая среда разработки в течение *всего учебного года*.

10.6 Дистанционная поддержка дисциплины

Дистанционная поддержка дисциплины обеспечивается использованием LMS. В разделе дисциплины «Алгоритмы и структуры данных 2018 уч. год Б 2 курс (код 103504)» (первая часть дисциплины АиСД, Бакалавриат, курс 2, 2018-2019 учебный год) размещаются дополнительные материалы, связанные с лекциями, практическими занятиями, материалы для самоподготовки, проекты. Дополнительно, могут использоваться и другие онлайн-инструменты коммуникации внутри команды (большой группы) – напр., Slack и др.

11 Материально-техническое обеспечение дисциплины

Проектор (лекционные и практические занятия), классы для практических занятий, оснащенные компьютерами с установленной инструментальной средой разработки MS Visual Studio 2015 (или более поздней версии).

12 Дополнительная информация (опция)

Студентам могут быть рекомендованы для самостоятельного изучения онлайн-курсы {$\mathbf{c_1}$} *«Algorithm Fundamentals»* (https://www.coursera.org/learn/algorithmic-toolbox) и {$\mathbf{c_2}$} *«Data Structures»* (https://www.coursera.org/learn/data-structures; два курса из специализации *«Data Structures and Algorithms»*), которые предлагаются Калифорнийским Университетом в Сан-Диего (University of California, San Diego) и Высшей Школой Экономики (НИУ ВШЭ). Успешное завершение курса (представление сертификата / Course Certificate с итоговой оценкой $g_{res}(\mathbf{c_i}) \geq 70$ (из 100 баллов)) дает возможность студенту получить дополнительные бонусы $\mathbf{B_{cert.}}$, учитываемые (добавляемые с весом 1) при вычислении накопленной оценки $O_{\text{накопленная}}$ (см. стр. 8 программы); оценка-бонус $\mathbf{B_{cert.}}$ определяется следующим образом:
(1) если завершен один курс ($\mathbf{c_1}$ *или* $\mathbf{c_2}$) с оценкой ≥ 70, тогда при $70 \leq g_{res}(\mathbf{c_i}) < 83$ – $\mathbf{B_{cert.}} = 0.2$; при $83 \leq g_{res}(\mathbf{c_i}) < 93$ – $\mathbf{B_{cert.}} = 0.25$; если же $93 \leq g_{res}(\mathbf{c_i}) \leq 100$, то $\mathbf{B_{cert.}} = 0.35$,
(2) если завершены оба курса ($\mathbf{c_1}$ *и* $\mathbf{c_2}$), и итоговая оценка обоих курсов ≥ 70, то $\mathbf{B_{cert.}}$ есть сумма $\mathbf{B_{cert.}}$ каждого курса (см. пункт 1), т.е. суммарный бонус может составить максимум 0.7 балла.

• *Примечание*: внимательно следите за датами (начала курса), предъявляемыми требованиями, имеющимся в распоряжении временем и пр. для принятия окончательного решения (решаетесь ли вы на это или нет) и планирования общей нагрузки. Данное предложение должно рассматриваться как опция, от которой можно отказаться, именно поэтому она оценивается дополнительными баллами.