МИНОБРНАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
Научно-образовательный центр «Высшая ИТ школа»

УТВЕРЖДАЮ
Исполнительный директор НОЦ ВИТШ

Т.С.Кетова

Фонд оценочных средств
по дисциплине
Б1.О.13 Программирование (основы) 1

Направление подготовки
09.03.04 Программная инженерия

Квалификация
Программный инженер

Год приема
2024

СОГЛАСОВАНО:

Руководитель ОП
О.А.Змеев

Председатель УМК
Д.О. Змеев

Томск-2024

1. Компетенции и результаты обучения, формируемые в результате освоения дисциплины «Б1.О.13 Программирование (основы) 1»

Компетенция	**Планируемые результаты обучения**, характеризующие этапы формирования компетенций	**Вид оценочного средства** (тесты, задания, кейсы, вопросы и др.)	**Критерии оценивания результатов обучения**			
			Неудовлетворительно (уровень освоения не сформирован)	Удовлетворительно (пороговый уровень освоения)	Хорошо (базовый уровень освоения)	Отлично (повышенный уровень освоения)
БК-1 Способен применять общие и специализированные компьютерные программы при решении задач профессиональной деятельности	Знает: правила и принципы применения общих и специализированных компьютерных программ для решения задач профессиональной деятельности Умеет: применять современные IT-технологии для сбора, анализа и представления информации; использовать в профессиональной деятельности общие и специализированные компьютерные программы	*Практические задание*	Студент не способен эффективно использовать какие-либо профессиональные инструменты разработчика. Студент с трудом пишет программный код для алгоритмов с использованием базовых инструментов и испытывает трудности в понимании	Студент может использовать некоторые базовые инструменты разработчика под руководством и при поддержке. Студент может писать программный код для простых алгоритмов, но может столкнуться с трудностями, когда дело доходит до более сложных	Студент способен использовать большинство профессиональных инструментов разработчика с легкостью и минимальным руководством. Студент может писать программный код для широкого спектра алгоритмов, включая сложные алгоритмы. Студент демонстрирует последовательное улучшение своих навыков и	Студент обладает высоким уровнем владения всеми профессиональными инструментами разработчика. Студент может легко и эффективно писать программный код для сложных алгоритмов. Студент демонстрирует исключительные навыки и знания и может оказывать руководство и поддержку другим.

			использования профессиональных инструментов. Студент не демонстрирует никакого улучшения в своих навыках, несмотря на руководство и поддержку.	алгоритмов. Студент демонстрирует некоторое улучшение в своих навыках, но нуждается в большем количестве практики, чтобы стать опытным.	становится все более опытным.	
БК-4 Способен разрабатывать алгоритмы для решения вычислительных задач и объяснять, как программы реализуют алгоритмы с точки зрения обработки инструкций, выполнения программы и	Знает: общую теорию вычислений на вычислительной технике, трудоемкость и ресурсоемкость алгоритмов, механизмы хранения и обработки данных в форме переменных Умеет: Декомпозировать сложные вычислительные задачи на более простые; реализовывать алгоритмы в разных стилях написания и языках	*Практические задание*	Студент не в состоянии составить базовую структуру алгоритма для решения практических задач. Учащийся испытывает трудности с пониманием проблемы и определением соответствую	Студент способен составить базовую структуру алгоритма для решения некоторых простых практических задач с руководством и поддержкой. Студент может понять проблему и определить	Студент способен составить базовую структуру алгоритма для решения большинства практических задач с легкостью и минимальным руководством. Студент может понять проблему и определить соответствующие шаги для ее решения, даже для	Студент обладает большим опытом в составлении базовой структуры алгоритма для решения любой практической задачи. Студент может быстро и эффективно понять проблему и определить соответствующие шаги для ее решения, даже для сложных задач. Студент обладает высокими навыками написания программного кода алгоритма для решения

запущенных процессов	программирования; искать дефекты в алгоритмах и их устранять; оптимизировать реализацию алгоритмов		щих шагов для ее решения. Студент не в состоянии написать программный код алгоритма для решения практических задач. Учащийся испытывает трудности с переводом алгоритма в код или выявлением и исправлением ошибок в своем коде. Студент не способен оптимизировать программный код или структуру алгоритма, если он не соответствует требованиям, описанным в условии	соответствующие шаги для ее решения, но может столкнуться с более сложными проблемами. Студент способен написать программный код алгоритма для решения некоторых простых практических задач с руководством и поддержкой. Студент может перевести алгоритм в код и выявить и исправить ошибки в своем коде, но может столкнуться с трудностями, когда дело доходит до более сложных задач. Студент способен оптимизироват	сложных задач. Студент способен написать программный код алгоритма для решения большинства практических задач с легкостью и минимальным руководством. Студент может перевести алгоритм в код и выявить и исправить ошибки в своем коде, даже для сложных задач. Студент способен оптимизировать программный код или структуру алгоритма для удовлетворения большинства требований с легкостью и минимальным руководством. Студент может быстро и эффективно определить области, которые нуждаются в	любой практической задачи. Студент может быстро и эффективно перевести алгоритм в код, а также выявить и исправить ошибки в своем коде даже для сложных задач. Студент обладает большим опытом в оптимизации программного кода или структуры алгоритма для удовлетворения любых требований. Студент может быстро и эффективно определить области, которые нуждаются в оптимизации, и внести улучшения даже для самых сложных задач. Студент демонстрирует исключительные навыки и знания и может оказывать руководство и поддержку другим.

			практической задачи. Студент не имеет удовлетворительного представления о требованиях или не в состоянии определить области, которые нуждаются в оптимизации. Студент не демонстрирует никакого улучшения в своих навыках, несмотря на руководство и поддержку	ь программный код или структуру алгоритма для удовлетворения некоторых простых требований с помощью руководства и поддержки. Студент может определить области, которые нуждаются в оптимизации, и внести некоторые улучшения, но может испытывать трудности с более сложными задачами. Студент демонстрирует некоторое улучшение в своих навыках, но нуждается в большем количестве практики, чтобы стать опытным.	оптимизации, и внести улучшения даже для сложных задач. Студент демонстрирует последовательное улучшение своих навыков и становится все более опытным.	

2. Типовые контрольные задания или иные материалы, необходимые для оценки образовательных результатов обучения

2.1. Типовые задания для проведения текущего контроля успеваемости по дисциплине «Б1.О.13 Программирование (основы) 1»

Типовое задание на Контрольную по 1ому модулю предмета

Разработчик Вася очень не любит длинные названия переменных. Для Васи длинное название - это такое, длина которого состоит больше, чем из десяти символов.

Вася придумал для себя алгоритм замены таких длинных названий на более короткие записывается первая и последняя буква слова, а между ними — количество букв между первой и последней буквой.

Таким образом, переменная “synchrophasotron” преобразуется в переменную “s14n”.

Позже Вася обнаружил недостаток своего алгоритма - существуют переменные сокращенные названия которых совпадают - он назвал это явление колизей. Для переменных с коллизией он придумал следующий способ решения: в конце сокращенного названия переменной добавляется номер коллизии для неё.

Так, для переменные “synchrophasotron” и “synchrocyclotron” будут выглядеть следующим образом: “s14n” и “s14n1”

Ваша задача написать алгоритм, который заменить все большие названия переменных на сокращенные

Формат входных данных

В первой строке содержится целое число n ($1 \le n \le 100$).

В каждой из последующих n строк содержится по одному слову. *Все слова состоят из малых латинских букв и имеют длину от 1 до 100 символов.*

Формат выходных данных

Выведите n строк. В i строке должен находиться результат замены i-го слова из входных данных.

Входные данные	Выходные данные
3 code synchrophasotron	code s14n s14n1

synchrocyclotron	
1 hedgehog	hedgehog
4 word localization internationalization pneumonoultramicroscopicsilicovolcanoconiosis	word l10n i18n p43s

Типовое задание на Контрольную по 2ому модулю предмета

Студент Хитса в качестве дипломной работы решил взять очень трудоемкий алгоритм. Суммарно у него есть n дней, чтобы запустить особую программу и представить полученные результаты. Но есть одна проблема: программе нужно работать x дней, чтобы подсчитать результаты. Но, к счастью, программу можно оптимизировать. Если потратить y (целое неотрицательное) дней на оптимизацию, то программа станет работать за $\frac{x}{y+1}$дней. Программу можно оптимизировать сколько угодно дней, но запустить можно только один раз (т.е. нельзя оптимизировать программу первый раз, потом запустить, потом оптимизировать ещё раз. Можно только вначале оптимизировать, а потом запустить). В целом остаётся главный вопрос: успеет ли студент Хитса представить результаты программы за n дней.

Входные данные
В первой строке задано единственное целое число T ($1 \le T \le 50$) — количество наборов входных данных. В следующих T строках заданы сами наборы – по одному в строке. Каждая строка содержит два целых числа n и x (оба до 10^9) — количество дней перед дедлайном и количество дней, за которое срабатывает программа.

Выходные данные
Выведите T ответов, YES если студент успеет, EPIC FAIL если нет

3 1 1 4 5 5 11	YES YES EPIC FAIL

Типовое задание на Контрольную по 3ому модулю предмета

Вам дано двоичное дерево (слева потомки меньше узла, справа больше узла), в котором все элементы уникальны по значению.
Ваша задача - придумать, как вывести путь от узла со значением X до корня дерева.

На вход даётся значение N - количество узлов дерева. Далее в каждой из N строк для вершины с порядковым номером *i* вводится два числа: *vi* и *pi* - значение вершины и порядковый номер родителя вершины (у корня дерева номер родителя равен нулю).
Затем вводится число K - количество вершин, от которых нужно будет построить путь до корня дерева. В каждой из K последующих строк содержится число m - значение узла. Гарантируется, что узел со значением m присутствует в дереве.

В ответ выведите путь (цепочку порядковых номеров вершин) от каждой из K вершин до корня дерева. Каждый путь должен отделяться новой строкой.

При решении задачи обязательно необходимо использовать структуру (-ы)

Пример

Входные данные	Выходные данные
9 64 0 29 1 87 1 24 2 68 3 100 3 27 4 65 5 76 5 3 100 76 64	6 3 1 9 5 3 1 1

Типовое задание на Контрольную по 4ому модулю предмета

На одном из предметов в Выдающейся Информативной Топовой Школе студентам необходимо сдавать задачи. Всего задач 9, но за раз можно сдать лишь 4 различные задачи. Конечно же, студенты последнего набора понимают важность предмета и ни за что не списывают. И всё же им хочется сформировать очередь на сдачу таким образом, чтобы максимально похожие решения были сданы в разном порядке. Поэтому они договорились, что проводиться сдача будет согласно следующим правилам:

1. Можно увеличить номер первой сдаваемой задачи на 1, если он не равен 9, и если новый номер задачи не встречается в последовательности сдаваемых студентом задач. Если встречается, то номер можно увеличить до ближайшего числа, которое не входит в последовательность.
2. Можно уменьшить номер последней сдаваемой задачи на 1, если он не равен 1 и если новый номер задачи не присутствует в последовательности сдаваемых студентом задач. Если встречается, то номер можно уменьшить до ближайшего числа, которое не входит в последовательность.
3. Можно циклически сдвинуть все номера на один вправо.
4. Можно циклически сдвинуть все номера на один влево.

Например, применяя эти правила к порядку задач 1235, можно получить числа 4235, 1236, 5123 и 2351 соответственно.
Точные правила сдачи студенты пока не придумали, но пока их интересует вопрос, как получить из одного порядка сдачи задач другой за минимальное количество операций.

Формат входного файла
Во входном файле содержится два различных четырехзначных числа, каждое из которых не содержит нулей

Формат выходного файла
Программа должна вывести последовательность четырехзначных чисел, не содержащих нулей. Последовательность должна начинаться первым из данных чисел и заканчиваться вторым из данных чисел, каждое последующее число в последовательности должно быть получено из предыдущего числа применением одного из правил. Количество чисел в последовательности должно быть минимально возможным.

Input	Output
9876	9876

5987	9875 5987

2.2. Типовые задания для проведения промежуточной аттестации по дисциплине «Б1.О.14 Программирование (основы) 1»

Вам даны n пар прямоугольников. Каждая пара прямоугольников находиться на отдельной плоскости и взаимодействия между парами не происходит…
Вам необходимо для каждой пары подсчитать площадь пересечения прямоугольников и отсортировать все пары в порядке этой площади

Формат входных данных
n
далее n строк по 8 чисел, 2 координаты левого верхнего угла первого прямоугольник, 2 координаты правого нижнего угла первого прямоугольника, 2 координаты левого верхнего угла второго прямоугольник, 2 координаты правого нижнего угла второго прямоугольника

Формат выходных данных
Порядковые номера пар прямоугольников в отсортированном виде

В волшебном мире есть волшебники, которые умеют делать волшебные сумки разной вместимости, которые умеют вмещать любой объём предметов в них, с самым главным условием, чтобы суммарно эти предметы весили не более **S** килограмм(но каждую сумку можно зачаровать на отдельный вес и нельзя помещать одну такую сумку в другую).
Но поскольку никто не знает как волшебники это делают, то волшебники жадно установили на такие сумки ужасные цены, если сумка может вмещать не более S килограммов, то цена за такую сумку будет $S^{log_{10} S}$ золотых монет… Помимо этого, поскольку волшебникам было лень поддерживать чары на сумки разных размеров правильно, то они добавили предупреждение в руководство пользователя: если предмет весит меньше чем $Log_{10} S$, то он растворяется в сумке(и предмета больше не существует)
Эпический герой решил купить для себя такую сумку, но у него возник вопрос… Очевидно, что чем больше он возьмет сумку, тем дороже она ему обойдется, поэтому он решил каждому своему предмету(которые еще и весят по разному) подсчитать полезность. И наш эпический герой будет считать, что покупка сумки была выгодна только если максимальная сумма полезности всех предметов, которые есть у героя и которые он сможет в неё положить(без уничтожения), больше чем стоимость этой сумки. Дальше герой решил найти максимальный размер выгодной сумки, но не смог… Теперь ваша задача найти максимально большую выгодную сумку

Формат входных данных
n - количество предметов, которые есть у героя, далее для каждого предмета пара чисел полезность и вес этого предмета

Формат выходных данных
Искомый размер максимально выгодной сумки

5 1 1 2 2 3 3 4 4 5 5	10

P.S. Гарантируется что сумма полезности всех предметов не превысит 100 000 000
Суммарный вес предметов ограничен 10^18
Ваша задача написать алгоритм для поиска в глубину, который в качестве основного представления графа используя список рёбер
Прим.

1) Вам нельзя преобразовывать граф в матрицу смежности или список смежности. Т.Е. можно использовать или список рёбер или его производные или альтернативное представление списка рёбер
2) Суммарная эффективность алгоритма должна быть примерно такой же как и поиск в глубину выполненный на матрице смежности (допускается небольшое/ухудшение производительности), хотя при эффективной реализации будет быстрее