

**Паспорт
расчетно-графического задания (работы)**

по дисциплине «Алгоритмы и структуры данных», 5 семестр

1. Методика оценки

Задание на РГЗ направлено на достижение знаний, умений и компетенций:

- ОПК3: Готовность применять основы информатики и программирования к проектированию, конструированию и тестированию программных продуктов.
- ПК.22 Способность создавать программные интерфейсы.
- з3. Знать принципы и методологии структурного и объектно-ориентированного подходов к разработке программного обеспечения.

При выполнении расчетно-графического задания (работы) студенты должны спроектировать и реализовать библиотеку классов для решения различных задач на графах. Задание к РГЗ содержит три основных пункта:

1. Реализация универсального класса для программирования графов различной природы (неориентированных, ориентированных, взвешенных) и ассоциированных с ним вспомогательных классов
2. Реализация класса для решения заданной вариантом задачи 1 на ориентированном или неориентированном графе.
3. Реализация класса для решения заданной вариантом задачи 2 на ориентированном или неориентированном взвешенном графе.

1.1 Основные этапы выполнения и структурные части РГЗ:

1.1.1 Реализация класса «Универсальный граф» и ассоциированных вспомогательных классов

Реализация и отладка класса «Граф» и вспомогательных классов выполняется с использованием технологии объектно-ориентированного программирования. Граф реализуется в виде класса «Простой граф» и ассоциированных классов «Дескриптор вершины графа», «Дескриптор ребра графа», «Итератор вершин графа», «Итератор ребер графа», «Итератор исходящих ребер вершины».

- Объект «Простой граф» реализуется в виде шаблонного класса.

Параметрами шаблона являются типы «Дескриптор вершины графа» и «Дескриптор ребра графа».

- Объект «Дескриптор вершины графа» реализуется в виде шаблонного класса. Параметрами шаблона являются тип, задающий имя вершины, и тип данных, связанных с вершиной.
- Объект «Дескриптор ребра графа» реализуется в виде шаблонного класса. Параметрами шаблона являются тип «Дескриптор вершины графа», тип, задающий вес ребра, и тип данных, связанных с ребром.
- Ассоциированные классы («Итератор вершин графа», «Итератор ребер графа», «Итератор исходящих ребер вершины») реализовать, как вложенные в класс «Простой граф».
- Для реализации различных форм представления графа используются вспомогательные классы «L – граф» и «M – граф». Рекомендуется объединить эти классы в иерархию с использованием наследования и полиморфизма.
- Объект «Простой граф» содержит вектор дескрипторов вершин, вставленных в граф.
- Объект «Простой граф» содержит структуру графа в виде списков смежности (L-граф) или матрицы смежности (M-граф). Элементы списков смежности или матрицы смежности содержат дескрипторы ребер, вставленных в граф.
- Для согласования систем идентификации вершин в прикладной программе (имена вершин) и в объекте «Простой граф» (адреса дескрипторов вершин) рекомендуется использовать объект «Отображение имени на адрес дескриптора». Объект реализуется в виде шаблонного класса. Параметрами шаблона являются тип имени вершины графа и указатель на «Дескриптор вершины графа». Такой объект обеспечивает прямой доступ по имени вершины к адресу дескриптора вершины. Объект - отображение можно реализовать на базе ассоциативной структуры (сбалансированное дерево, хеш-таблица), объекта библиотеки STL - `map` или `ash_map`, где ключом является имя вершины и т.п.

1.1.2 Реализация классов для решения задачи 1 и/или задачи2

Для реализации задач 1 - 2, заданных в варианте задания, разрабатываются объекты – клиенты для объекта «Простой граф». Объект задачи связан с объектом графа, для которого решается задача, с помощью указателя на объект «Простой граф».

- Объект задачи для решения задачи использует интерфейс объекта «Простой

граф» и интерфейс ассоциированных с графом объектов («Дескриптор вершины графа», «Дескриптор ребра графа», «Итератор вершин графа», «Итератор ребер графа», «Итератор исходящих ребер вершины»).

- Объект задачи содержит все структуры, необходимые для решения задачи, сохранения результатов решения задачи.
- Решение задачи происходит в момент создания объекта задачи, в момент вызова методов Set (g) (связывание объекта задачи с другим графом g), вызова метода Restart () (повторное решение задачи для графа).
- Метод Result () формирует результат решения задачи для клиентской программы в удобной форме, в соответствии с целью решаемой задачи.

1.2 Критерии оценки

1. Работа считается не выполненной, если не выполнены все 3 пункта задания на РГЗ или выполнен первый пункт задания с существенными отступлениями от требований, сформулированных в разделе 1.1.1. Оценка составляет 0 баллов.

2. Работа считается выполненной на пороговом уровне, если выполнен пункт 1 задания с соблюдением требований к реализации, сформулированных в разделе 1.1.1. Оценка составляет 10 баллов.

3. Работа считается выполненной на базовом уровне, если выполнены пункт 1 и один из пунктов 2 или 3 задания. При реализации пунктов должны быть соблюдены требования, задания с соблюдением требований к реализации, сформулированные в разделах 1.1.1 и 1.1.2. Оценка составляет 20 баллов.

4. Работа считается выполненной на продвинутом уровне, если выполнены все три пункта задания к РГЗ. При этом соблюдены все требования к реализации, сформулированные в разделах 1.1.1 и 1.1.2. Оценка составляет 30 баллов.

1.3 Шкала оценки

В общей оценке по дисциплине баллы за РГЗ(Р) учитываются в соответствии с правилами балльно-рейтинговой системы, приведенными в рабочей программе дисциплины.

Для аттестации студентов используется балльно-рейтинговая система. Рейтинг студента по дисциплине определяется как сумма баллов за работу в семестре (текущий рейтинг) и баллов, полученных в результате промежуточной аттестации (дифференцированный зачет).

Максимальная сумма баллов за 5 семестр составляет 100 (текущий рейтинг - 80

баллов, дифференцированный зачет - 20 баллов). Минимальная сумма баллов за 5 семестр - 50 баллов.

В таблице 1 приведено максимальное количество баллов, которое может набрать студент по разным видам учебной деятельности в течение 5 семестра и диапазоны баллов, соответствующие минимальному и максимальному количествам баллов. Максимальная сумма баллов за 5 семестр составляет 100 (текущий рейтинг – 80 баллов, зачет – 20).

Таблица 1 Сроки и диапазоны баллов для всех видов учебной работы

№п/п	Вид учебной работы (учебной деятельности)	Максимальное количество баллов	Диапазоны баллов	Срок представления и защиты (неделя семестра)
Пятый семестр:				
1	Лабораторная работа № 1	6	4-8	7
2	Лабораторная работа № 2	8	6 - 12	11
3	Лабораторная работа № 3	12	7 - 14	15
4	Лабораторная работа № 4	14	8- 16	17
5	Расчетно-графическое задание	30	10 - 30	17
Итого по текущему рейтингу:		80	40 - 80	
6	Дифференцированный зачет	20	10-20	
Итого за пятый семестр:		100	50-72 (удовл.) 73-87 (хор.) 88-100 (отл.)	

В результате текущей и промежуточной аттестации итоговые баллы переводятся в оценку системы ECTS и пятибальную шкалу согласно таблице 2:

Таблица 2 Сроки и диапазоны баллов для всех видов учебной работы

90 - 100	A+ (99 - 100)	отлично
	A (93 - 98)	
	A- (90 - 92)	
80 - 89	B+(87 - 89)	хорошо
	B (83 - 86)	
	B-(80 - 82)	
70 - 79	C+(77- 79)	
	C (73 -76)	
	C-(70 - 72)	
60 - 69	D+(67 - 69)	удовлетворительно
	D (63 -66)	
	D-(60 - 62)	
50 - 59	E (50 -59)	
25 - 49	FX	не удовлетворительно
0 - 24	F	

Правила текущей аттестации РГЗ:

В течение 5-го семестра обучения (до 17 учебной недели включительно) необходимо представить и защитить расчетно-графическое задание (РГЗ) согласно варианту.

Максимальное количество баллов - 30 выставляется в случае, если успешно выполнены пункты 1- 3 задания к РГЗ при условии своевременной (до конца 17 учебной недели включительно) и успешной защиты.

Количество баллов 20 выставляется в случае, если успешно выполнены пункты 1, и один из пунктов 2-3 к РГЗ при условии своевременной и успешной защиты.

Минимальное количество баллов 10 выставляется в случае, если успешно выполнен пункт 1 задания к РГЗ при условии своевременной и успешной защиты.

В случае пересдачи или представления и защиты РГЗ с опозданием от учебного графика происходит потеря баллов (опоздание на 1 неделю - потеря 1 балла).

2. Примерный перечень тем РГЗ(Р)

2.1 Пункт 1 задания

Реализация универсального класса для программирования графов различной природы (неориентированных, ориентированных, взвешенных) и ассоциированных с ним вспомогательных классов.

2.1.1 Разработать АТД «Простой граф».

Интерфейс АТД «Простой граф» включает операции:

Конструктор () по умолчанию: создает пустой L - граф с нулевым числом вершин и ребер,

Конструктор (V, D, F) создает граф с V вершинами, без ребер, типа D (ориентированный / неориентированный), формы представления F (L- граф/М-граф),

Конструктор (V, E, D, F) создает граф с V вершинами, с E случайными ребрами, типа

D (ориентированный / неориентированный), формы представления F (L- граф/М-граф),

Конструктор (G) - конструктор копирования создает объект – копию графа G,

Деструктор () уничтожает внутренние структуры графа,

V () - возвращает число вершин в графе,

E () - возвращает число ребер в графе,

Directed () - возвращает тип графа (ориентированный / неориентированный)

Dense () - возвращает форму представления графа (L- граф / M- граф),

K () - возвращает коэффициент насыщенности графа,

ToListGraph () преобразует граф к L- графу,

ToMatrixGraph () преобразует граф к M- графу,

InsertV () добавляет вершину к графу и возвращает адрес дескриптора вновь созданной вершины,

DeleteV (v) - удаляет вершину из графа, заданную адресом дескриптора **v**,

InsertE(v1, v2) - добавляет ребро (**v1, v2**) к графу, соединяющую вершины, заданные адресами дескрипторов **v1** и **v2**, и возвращает адрес дескриптора вновь созданного ребра - **e**,

DeleteE (v1, v2) удаляет ребро, соединяющее вершины, заданные адресами дескрипторов **v1** и **v2**,

GetEdge (v1, v2) возвращает адрес дескриптора ребра соединяющего вершины, заданные дескрипторами **v1** и **v2**,

2.1.2 Разработать ассоциированные с графом типы:

АТД «Дескриптор вершины графа»

Дескриптор вершины содержит поля:

name – имя вершины,

data – данные, связанные с вершиной,

index – индекс вершины в структуре графа или -1,

Интерфейс АТД «Дескриптор вершины графа» включает операции:

Конструктор (): поле **name** не определено, поле **data** не определено,

Конструктор (name, data): **name** - имя вершины, **data** - данные, связанные с вершиной,

GetName () - возвращает имя вершины,

GetData () - возвращает данные, связанные с вершиной,

SetName (name) – задает имя вершины,

SetData (data) – записывает данные **data** в дескриптор вершины.

АТД «Дескриптор ребра графа»

Дескриптор ребра содержит поля:

v1 - дескриптор вершины, из которой исходит ребро,

v2 - дескриптор вершины, в которую входит ребро,

w - вес ребра,

data - данные, связанные с ребром,

Интерфейс АТД «Дескриптор ребра графа» включает операции:

Конструктор (v1, v2): *v1* - дескриптор вершины, из которой исходит ребро, *v2* - дескриптор вершины, в которую входит ребро,

Конструктор (v1, v2, w): *v1* - дескриптор вершины, из которой исходит ребро, *v2* - дескриптор вершины, в которую входит ребро, *w* - вес ребра,

Конструктор (v1, v2, w, data): *v1* - дескриптор вершины, из которой исходит ребро, *v2* - дескриптор вершины, в которую входит ребро, *w* - вес ребра, *data* - данные, связанные с ребром

v1() - возвращает дескриптор вершины, из которой исходит ребро,

v2() - возвращает дескриптор вершины, в которую входит ребро,

GetW () - возвращает вес ребра,

SetW (w) - изменение веса ребра,

GetData () - возвращает данные, связанные с ребром,

SetData (data) - изменение данных, связанных с ребром.

АТД «Итератор вершин графа»

Интерфейс АТД «Итератор вершин графа» включает операции:

Конструктор () - создает итератор вершин графа,

beg () - возвращает итератор, установленный на первую вершину графа,

end () - возвращает итератор, соответствующий окончанию переходов итератора,

operator ++ - переход к следующей вершине графа,

operator * - возвращает дескриптор вершины графа, на которую указывает итератор.

АТД «Итератор ребер графа»

Интерфейс АТД «Итератор ребер графа» включает операции:

Конструктор () - создает итератор ребер графа,

beg () - возвращает итератор, установленный на первое ребро графа,

end () - возвращает итератор, соответствующий окончанию переходов итератора,

operator ++ - переход к следующему ребру графа,

operator * - возвращает дескриптор ребра графа, на которое указывает итератор.

АТД «Итератор исходящих ребер вершины»

Интерфейс АТД «Итератор исходящих ребер вершины» включает операции:

Конструктор (v) - создает итератор исходящих ребер графа для вершины, заданной дескриптором *v*,

beg () - возвращает итератор, установленный на первое исходящее ребро вершины,
end () - возвращает итератор, соответствующий окончанию переходов итератора,
operator ++ - переход к следующему исходящему ребру,
operator * - возвращает дескриптор исходящего ребра вершины, на которое указывает итератор.

2.2 Пункт 2 задания

Спроектировать и реализовать шаблонный класс для АТД «Задача 1» в соответствии с вариантом и использовать для решения задачи на неориентированном или ориентированном графе.

Интерфейс АТД «Задача 1» включает операции:

Конструктор (*g*) - создает объект задачи 1, ассоциированный с графом *g*, и выполняет решение задачи для графа *g*,

Конструктор (*T*) - конструктор копирования создает копию объекта – задачи *T*,

Деструктор () - уничтожает внутренние структуры объекта задачи,

Set (*g*) – связывает объект задачи с графом *g* и выполняет решение задачи 1 для графа *g*,

Restart () – повторно выполняет решение задачи 1 для графа *g*,

Result () – возвращает результат решения задачи 1

Варианты задачи 1:

- 1) формирования двудольного неориентированного графа с раскраской в два цвета,
- 2) определение кратчайших по числу ребер путей между всеми парами вершин орграфа,
- 3) определение гамильтонова цикла в орграфе,
- 4) определение диаметра связного неориентированного графа,
- 5) определение последовательности вершин в результате прямой топологической сортировки ациклического орграфа DAG (сортировка - на основе очереди истоков),
- 6) построение остовного дерева орграфа в пределах заданной высоты за счёт добавления минимального числа рёбер,
- 7) определение последовательности вершин на основе топологической сортировки ациклического орграфа DAG на основе поиска в глубину,
- 8) поиск всех простых циклов, включающих заданную вершину орграфа,
- 9) формирование реберно-связного неориентированного графа,
- 10) поиск простого цикла заданной длины, включающего заданную вершину,

- 11) построение редуцированного графа сильносвязных компонент,
- 12) классификация всех рёбер относительно заданной вершины орграфа,
- 13) определение вершин, отстоящих на расстоянии d от заданной вершины (d – число рёбер),
- 14) формирование двусвязного неориентированного графа,
- 15) определение эйлера цикла в неориентированном графе,
- 16) формирование списка ребер неориентированного графа в порядке двухпроходного эйлера цикла.

2.3 Пункт 3 задания

Спроектировать и реализовать шаблонный класс для АТД «Задача 2» в соответствии с вариантом и использовать для решения задачи на взвешенном графе.

Интерфейс АТД «Задача 2» включает операции:

Конструктор (g) - создает объект задачи 3, ассоциированный с графом g , и выполняет решение задачи для графа g ,

Конструктор (T) - конструктор копирования создает копию объекта – задачи T ,

Деструктор () - уничтожает внутренние структуры объекта задачи,

Set (g) – связывает объект задачи с графом g и выполняет решение задачи 3 для графа g ,

Restart () – повторно выполняет решение задачи 3 для графа g ,

Result() – возвращает результат решения задачи 3

Варианты алгоритмов:

- 1) определение диаметра и пути, соответствующего диаметру, на основе алгоритма Флойда,
- 2) приведение отрицательной весовой функции орграфа к положительной методом Джонсона,
- 3) определение входящих в заданную вершину кратчайших путей на основе алгоритма Дейкстры,
- 4) определение центра взвешенного орграфа на основе алгоритма Беллмана – Форда.
- 5) разбиение вершин взвешенного неориентированного графа на кластеры, объединяющие вершины ребрами с длиной, большей d . Для разбиения использовать алгоритм Крускала,
- 6) определение периферии взвешенного орграфа на основе алгоритма Дейкстры,

- 7) определение списка вершин, которые лежат в пределах заданного расстояния d от заданной вершины взвешенного орграфа с отрицательной весовой функции на основе алгоритма Беллмана-Форда,
- 8) определение первого по величине (по количеству вершин) кластера в неориентированном взвешенном графе, вершины которого объединены ребрами с длиной, большей d . Для определения кластера использовать алгоритм Прима,
- 9) определение периферии взвешенного орграфа на основе алгоритма Флойда,
- 10) нахождения ребра (ребер), устранение которых вызывает максимальное возрастание кратчайшего пути из вершины u в вершину v во взвешенном орграфе на основе алгоритма Дейкстры,
- 11) определения кратчайших путей для всех пар вершин взвешенного орграфа на основе алгоритма Беллмана-Форда,
- 12) построение минимального остовного дерева относительно заданной взвешенного неориентированного графа методом Прима,
- 13) определение радиуса и соответствующего радиусу пути взвешенного орграфа на основе алгоритма Дейкстры,
- 14) определение центра взвешенного орграфа на основе алгоритма Флойда,
- 15) определение эксцентриситета заданной вершины взвешенного орграфа с отрицательной весовой функции на основе алгоритма Беллмана-Форда,
- 16) формирование остовного дерева взвешенного неориентированного графа с суммарным весом ребер, ближайшим к заданному весу d . Для формирования дерева использовать алгоритм Крускала,