|  |
| --- |
| **№ 9 практическая работа.** Алгоритмы на графах. |

**Вопросы:**

* 1. Дайте определение понятиям, связанными с графами;
	2. Как создавать структуры данных, описывающие графы;
	3. Как обходить граф по уровням и в глубину;
	4. Как находить минимальное остовное дерево в связном графе;
	5. Как находить кратчайший путь между двумя вершинами связного графа;
	6. Как находить компоненты двусвязности связного графа.
	7. Опишите алгоритм последовательного поиска
	8. Анализ наихудшего случая и среднего случая алгоритма последовательного поиска
	9. Опишите алгоритм двоичного поиска
	10. Анализ наихудшего случая и среднего случая алгоритма двоичного поиска
	11. Опишите алгоритм выборки

**ЗАДАНИЕ:**

1. Сгенерируйте полный взвешенный граф с 50 вершинами. Выполните алгоритм Дейкстры-Прима поиска минимального остовного дерева в этом графе, начиная поочередно с каждой из его вершин, и подсчитайте количество найденных таким образом минимальных остовных деревьев. Повторите эту процедуру четырежды, устанавливая максимальное значение случайного веса ребра равным 10, 25, 50 и 100. Сведите результаты экспериментов в отчет и дайте им объяснение. Если для каждого из значений максимума Вы сгенерируете несколько тестовых графов, то результаты будут более обоснованы.

2. Сгенерируйте полный взвешенный граф с 50 вершинами. Выполните алгоритм Крускала поиска минимального остовного дерева в этом графе. Встретив два ребра с одинаковым весом, выбирайте добавляемое ребро случайным образом. Сгенерируйте для графа 10 минимальных деревьев и найдите среди них попарно различные. Поскольку выбор весов производится случайным образом, при наличии различных минимальных остовных деревьев они должны обнаружиться. Повторите эту процедуру четырежды, устанавливая максимальное значение случайного веса ребра равным 10, 25, 50 и 100. Сведите результаты экспериментов в отчет и дайте им объяснение. 'Если для каждого из значений максимума Вы сгенерируете несколько тестовых графов, то результаты будут более обоснованы.

3. Сгенерируйте полный взвешенный граф с 50 вершинами. Для каждой пары вершин (А и В) проверьте, совпадает ли генерируемый кратчайший путь из А в В с генерируемым кратчайшим путем из В в А и подсчитайте число различных пар. Повторите эту процедуру четырежды, устанавливая максимальное значение случайного веса ребра равным 10, 25, 50 и 100. Сведите результаты экспериментов в отчет и дайте им объяснение. Если для каждого из значений максимума Вы сгенерируете несколько тестовых графов, то результаты будут более обоснованы.

4. Напишите программу, которая будет генерировать полный взвешенный граф. Выполните алгоритм Дейкстры-Прима и Крускала поиска минимального остовного дерева в этом графе. Установите счетчики, подсчитывающие число посещений каждым алгоритмом каждого ребра. Другими словами, подсчитайте число обращений к матрице примыканий. Повторите эту процедуру четырежды, устанавливая максимальное значение случайного веса ребра равным 10, 25, 50 и 100. Сведите результаты экспериментов в отчет и дайте им объяснение. Если для каждого из значений максимума Вы сгенерируете несколько случайных тестовых графов различных размеров, то результаты будут более обоснованы.

5. Нарисуйте дерево решений для алгоритма двоичного поиска в списке из 12 элементов. Внутренние узлы дерева должны быть помечены проверяемыми элементами, левый непосредственный потомок внутреннего узла должен указывать, что происходит, когда целевое значение меньше проверяемого элемента, правый — когда оно больше.

6. Напишите алгоритм поиска по общему дереву. При выполнении двоичного поиска на большом наборе данных число необходимых сравнений может оставаться довольно большим.Так, например, один поиск в телефонном справочнике большого города вполне может потребовать 25 сравнений. Для улучшения этого показателя используются деревья общего вида, в которых у узла может быть больше двух непосредственных потомков. В узле общего дерева записывается несколько значений ключа, и непосредственные потомки узла задают поддеревья, в которых содержатся (а) элементы, меньшие всех значений ключа в узле, (б) элементы, большие первого значения, но меньшие остальных значений ключа в узле, (в) элементы, большие первых двух значений ключа в узле, но меньшие остальных его значений, и т.д.