Лабораторная работа № 1

Тема. Реализация проектов. Упорядочение исходного списка работ сетевого графика
Сроки выполнения работы: 1-2 недели семестра

Задание.

1. Изучить параграфы 2.1 – 2.4, обратить внимание на правила построения сетевых графиков (п. 2.3)

   (см. в Илиас: Лекция 1, Лекция 2)

2. Ознакомиться с тестовым примером в Приложении к данному заданию.
3. Для разработки алгоритма и отладки программы разработать свои тестовые примеры.
4. Разработать алгоритм[1] частичного упорядочения работ сетевого графика по заданному списку не упорядоченных взаимосвязанных работ некоторого проекта с учетом требований, изложенных в Приложении, и с учетом минимизации времени работы алгоритма на больших массивах. Выявить начальную и конечную вершины графа. Выявить все полные пути от начальной вершины до конечной.

   При разработке алгоритма и программы не должны использоваться стандартные библиотеки работы с графами и программы расчета сетевых графиков.

5. Привести описание алгоритма.
6. Составить и отладить программу на выбранном языке программирования, которая позволяет упорядочить заданный список работ и выводит на экран монитора:
   - исходный список работ;
   - частично упорядоченный (с шифрами событий и с индексами) список работ;
   - список всех полных путей СГ – от начальной вершины до конечной.
7. Оформить отчет. Защитить работу.

[1] В данной лабораторной работе приведен тестовый пример, в котором сетевой график (граф) имеет циклы. Пример приведен для наглядности и более глубокой иллюстрации материала по теме работы. **При разработке алгоритма и программы в лабораторной работе наличие циклов не учитывать. Учесть только наличие петель**.

Приложение

Тестовый пример с пояснениями

Табл. П.1 – входная информация: исходный список работ

A: шифр предшествующего события	**B**: шифр последующего события	τ_{AB} - продолжительность соответствующей работы*, связывающей события с шифрами A и B (дни)
30	11	13
2	8	6
100	31	13
31	30	6
8	3	8
100	2	8
3	3	10
2	10	9
31	35	7
100	35	9
3	1	5
15	40	7
2	31	4
30	1	5
8	30	3
3	16	8
40	11	4
35	30	9
8	30	4
8	3	8
11	20	17
30	3	4
31	8	10
35	40	10
30	40	8
35	11	6
10	3	3
17	10	0 (ε)
11	1	6
1	31	20
1	20	13

---

* В данном алгоритме этот параметр (вес дуги) роли не играет. При обращении к программе он передается в «таблице» выходной информации.

Из списка таблицы П.1 формируем таблицу П.2, отмечая (или удаляя) строки таблицы П.1, уже перенесенные в таблицу П.2 (чтобы не анализировать их при очередном цикле просмотра списка). В таблице П.2 получим список, который упорядочен с учетом логических связей дуг графа. Такой список работ назовем частично упорядоченным.

В данном примере табл. П.2 разбита на 2 табл. П2.1 и П.2.2.

При анализе логических связей параллельно формируем список путей в графе для выдачи информации пользователю.

Первый шаг: нахождение начальной вершины графа (см. табл. П.2.1). На втором шаге: выявление логических связей дуг графа (см. табл. П.2.2). Цель построения таблицы: предоставить пользователю информацию для анализа полноты списка работ. После анализа и возможной корректировки списка взаимосвязанных работ строится СГ и вычисляются его параметры.

Табл. П.2.1 – промежуточная информация

A	B	τ_{AB}	Примечание	Анализ
100	31	13	«100» - начальное событие, в него не входит ни одна работа (дуга), его шифра нет в столбце B	Начальное событие, как и конечное, в СГ может быть **только одно**. Решение принимает пользователь. В результате: либо удаляем одно из событий, либо вводим фиктивную вершину графа. Пусть принято решение – ввести фиктивную вершину с шифром «-100»
100	2	8		
100	35	9		
17	10	0 (ε)	«17» - начальное событие	
-100	17	0		Ввели 2 фиктивных работы (дуги), выходящих из фиктивной вершины . Появилось новое начальное событие: с шифром «-100». Помещаем эти две строки таблицы в ее начало.
-100	100	0		
15	40	7	«15» - начальное событие	Пусть принято решение – удалить событие 15 (это ошибка в исходных данных)

Табл. П.2.2 – промежуточная информация

A	B	τ_{AB}	Примечание	Анализ
-100	17	0	«-100» - начальное событие	
-100	100	0		
17	10	0 (ε)	«17» – событие, связанное с «-100» Отрезок пути: -100; 17; 10;	Анализ отрезка пути: цикла нет, петель нет
100	31	13		
100	2	8		
100	35	9		
10	3	3	Отрезок пути: -100; 17; 10; 3	
31	30	6	Продолжаем строить отрезки путей	
31	35	7		
31	8	10		
2	8	6		
2	10	9	«10» - уже было «обработано»	
2	31	4	«31» - уже было «обработано»	

Табл. П.2.2 – продолжение

35	30	9		
35	40	10		
35	11	6		
3	3	10	Выдача сообщения: «Ошибка. Дуга: 3 -3 удалена»	Решение принимается автоматически
3	1	5		
3	16	8		
30	11	13		
30	1	5		
30	3	4		
30	40	8		
8	3	8		
8	30	3	Выдача сообщения: «Ошибка. Работа : 8 -30 дублируется, но имеет два веса: 3 и 4. Выбрать нужное»	Удалить одну из дуг (строк таблицы), указанной пользователем. Сохранить введенное значение веса дуги.
8	30	4		
8	3	8	Работа 8 -3 введена 2 раза.	Одну из строк таблицы удалить. Решение принимается автоматически
40	11	4		
11	20	17		
11	1	6		
1	31	20	В формируемом списке путей есть путь: «31»-«30»-«1». Поэтому, выдача сообщения: «Ошибка. Найден цикл с дугой «1» - «31». Есть пути: 1-31-30-1 и 31-30-1». Отредактируйте исходную информацию	Go To → корректировка входной информации. Пусть работа «1» - «31» удалена: цикла в СГ не стало.
1	20	13		
Начальный список пуст. **Поиск конечной вершины СГ** (ее нет в столбце А)			Сообщение: «Вершины графа (события) с шифрами 16 и 20 конечные. Ввести фиктивную конечную вершину? да; нет»	Если да, то добавить фиктивную вершину и дуги к ней с нулевыми весами. Если нет, удалить одну из вершин.
			Выдача сообщения: «Какую конечную вершину СГ и входящие в нее работы удалить: «16» или «20»?»	Если удалить вершину «16», сетевой график будет построен верно: будет иметь одну конечную вершину : с шифром «20». Если удалить вершину «20», то окажется вновь две конечных вершины: «1» и «16» (повторить анализ в цикле)

Результат, выводимый на экран для анализа информации пользователем:

- Табл. П.1
- Табл. П.2
- список полных путей: от начального события до конечного.

Табл. П.2

A	B	τ_{AB}
-100	17	0
-100	100	0
17	10	0
100	31	13
100	2	8
100	35	9
10	3	3
31	30	6
31	35	7
31	8	10
2	8	6
2	10	9
2	31	4
35	30	9
35	40	10
35	11	6
3	1	5
30	11	13
30	1	5
30	3	4
30	40	8
8	3	8
8	30	3
40	11	4
11	20	17
11	1	6
1	20	13

Полные пути в СГ:

-100 17 10 3 1 20

-100 100 2 10 3 1 20

-100 100 2 8 3 1 20

….

-100 100 35 11 20

(здесь приведены лишь примеры путей из полного списка путей данного СГ)

Примечание. Для наглядности на нижеследующем рисунке приведен граф, построенный по информации таблицы П.2

(в лаб. работе его строить и выводить на экран **не нужно**, в программе аналогичная процедура также должна **отсутствовать**):