

정렬 알고리즘 종합

Practice 4

시작하기 전에

- 프로그램을 짤때 원하는 어떤 언어 (C/C++, Java, R, python 등)도 사용 가능.
- 코드를 짤때 코드의 맨위에 자신의 학번 이름을 주석으로 적고 각 단계에 대하여 자세한 주석을 다시오.
- 코드를 돌려서 결과물을 제출하라는 문제는 결과물을 제출할때 화면 캡처를 사용할 것. 이는 자신의 코드를 돌렸을때 나온 결과임을 보이기 위함으로 사용하는 언어나 에디터 등등에 따라 다를 수 있으므로 방법은 알아서 제출할 것. 어떤 방식이든 자신의 코드를 돌려서 나온 결과라는 것만 보여주면 됨. 예를들어, 자바 이클립스를 사용하면 이때 자신의 코드의 윗부분(학번 이름과 앞에 코드 5줄정도 포함)이 아래 실행 결과와 같이 캡처되도록 하시오.
- **다음 실습전날 저녁 9시까지 eclass에 업로드하고 출력물을 제출하**시오.

문제 1 (25점)

- 최소 최대값 동시 찾기 문제
- 1~100000범위내에 있는1000개의 랜덤한 양의 정수를 생성한 후
 - 수업시간에 배운 최소값 찾기 Minimum() 함수와 이를 응용한 Maximum()함수를 이용하여 최소값과 최대값을 찾고 이를 출력하여라.
 - 수업시간에 배운 FindMinMax() 함수를 이용하여 동시에 최소값과 최대값을 찾고 이를 출력하여라.






주의) 각 언어에 있는 패키지나 기본 함수를 사용하면 안되고 직접 함수를 구현해서 사용해야 함.

문제 2 (25점)

- 피보나치 수열
- 피보나치 수열은 13세기의 이탈리아 수학자 Leonardo Pisano Fibonacci에 의해서 처음 소개되었다. 그는 한쌍의 토끼가 해를 지나면서 번식하는 토끼의 수를 계산하려고 하였다. 아래 피보나치 문제를 풀어보자.
- 가정: One pair of adult rabbits could create a new pair of rabbits in about the same time that it takes bunnies to grow into adults. Thus, in a given period, each pair of adult rabbits produces a new pair of baby rabbits, and all baby rabbits grow into adult rabbits.

문제 2

- Let F_n represent the number of rabbits in period n , then we can determine the value of F_n in terms of F_{n-1} and F_{n-2} , $F_n = F_{n-1} + F_{n-2}$, $F_1 = F_2 = 1$

Months	Rabbit population growth	Number of pairs of rabbits
0		1
1		1
2		2
3		3
4		5

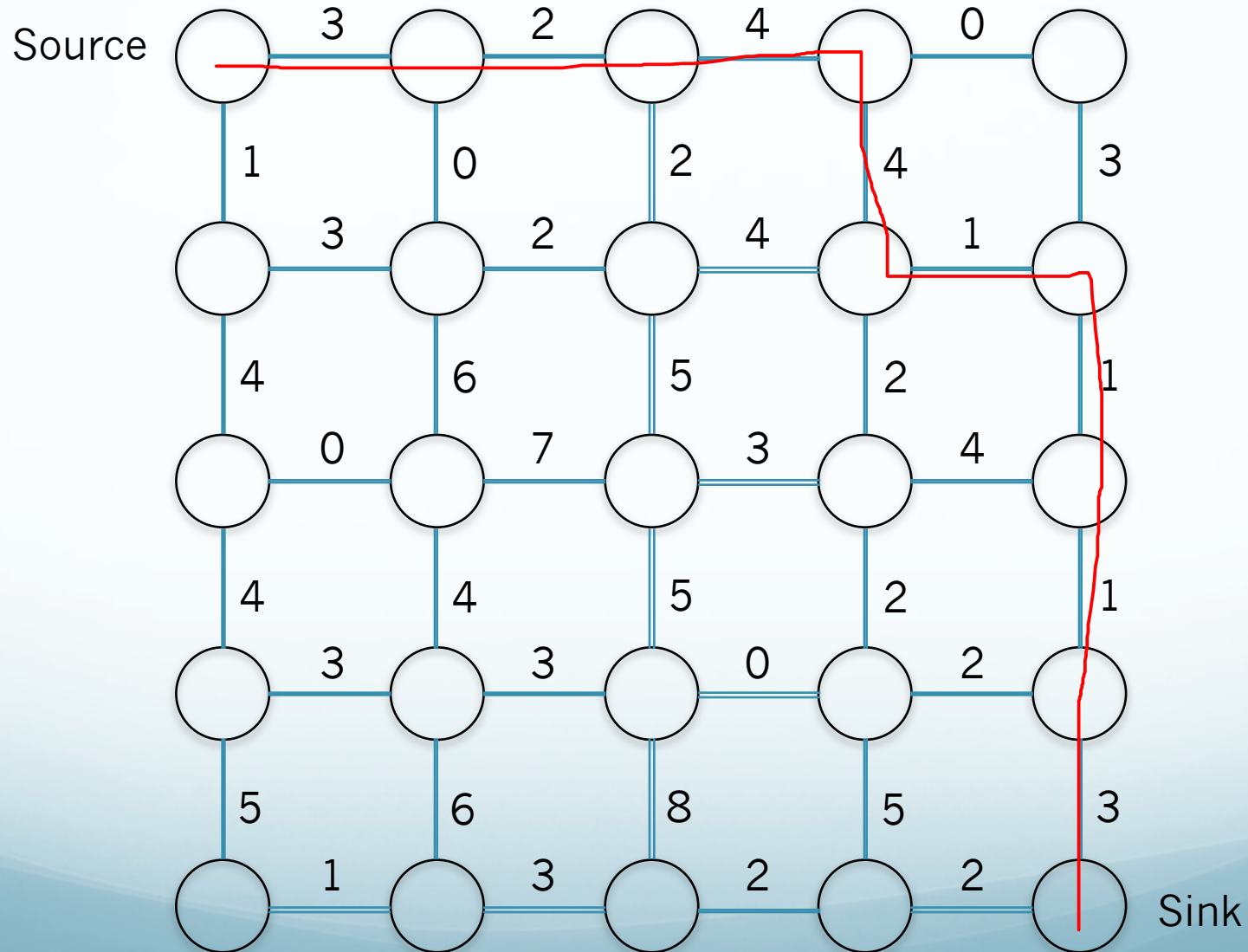
문제 2

- 피보나치 수열 함수를 구현해 보고 F_1 부터 F_{20} 까지를 출력해 보아라.

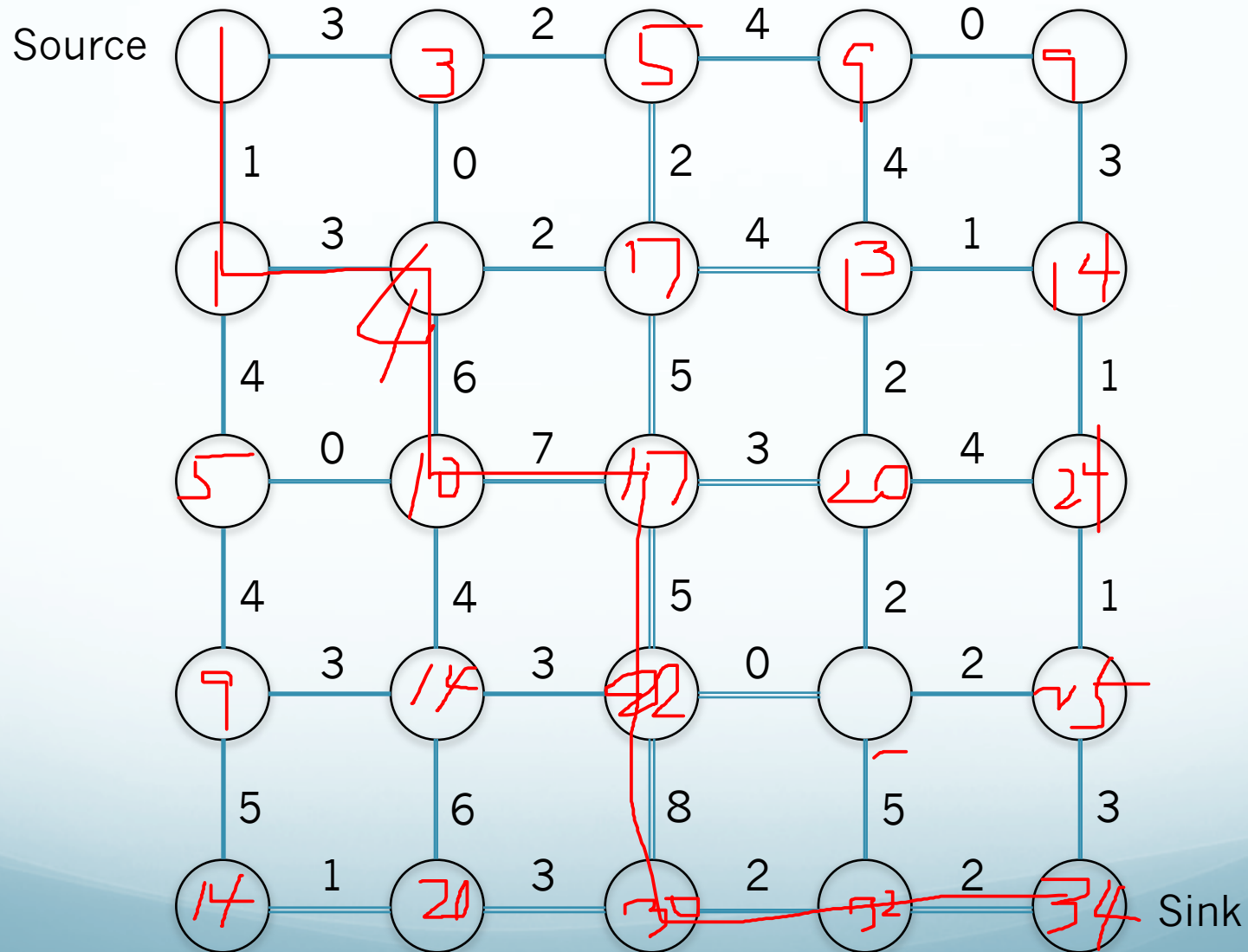
문제 3 (50점)

- Manhattan Tourist Problem
 - Manhattan Tourist Problem의 greedy 경우 경로를 구하시오. (10점)
 - Manhattan Tourist Problem의 Dynamic Programming 경우 경로를 구하시오. (10점)
 - Manhattan Tourist Problem의 Dynamic Programming 유사코드를 작성하시오. (30점)

Greedy Algorithm



Dynamic Programming



Dynamic Programming

- Manhattan Tourist Problem의 Dynamic Programming 유사코드를 작성하시오.
 - 각 vertex 는 $S_{i,j}$
 - 각 edge 는 $\overrightarrow{W_{i,j}}$, $W_{i,j}\downarrow$ 로 표기하시오. $\overrightarrow{W_{i,j}}$: east 로 이동시 $S_{i,j}$ 에 도달하는 edge를 가르킴.
 - n: column 수, m: row 수
 - $\text{ManhattanTourist}(\overrightarrow{W}, W\downarrow, n, m)$ 을 작성하시오.

정렬 복습

제출할 필요 없습니다. 각자 공부해보고 정리해 두면 중간고사에 많은 도움이 될 것입니다. 실습시간을 활용해서 미리 복습해 두도록 하세요.

비교기반 정렬 VS 분포기반 정렬

- 비교기반(comparison-based) 기반 알고리즘
 - 선택(selection) 정렬, 버블(bubble) 정렬, 삽입(insertion) 정렬, 셸(shell) 정렬, 퀵(quick) 정렬, 합병(merge) 정렬, 힙(heap) 정렬.
- 분포기반(distribution-based) 정렬
 - 계수(counting) 정렬, 기수(radix)정렬, 버킷(bucket) 정렬.

정렬 알고리즘 시간 복잡도 비교

- 비교기반 정렬 알고리즘

	Worst case	Average case
Selection sort		
Bubble sort		
Insertion sort		
Quick sort		
Merge sort		
Heap sort		

비교기반 정렬 VS 분포기반 정렬

- 비교기반(comparison-based) 기반 알고리즘
 - 선택(selection) 정렬, 버블(bubble) 정렬, 삽입(insertion) 정렬, 셸(shell) 정렬, 퀵(quick) 정렬, 합병(merge) 정렬, 힙프(heap) 정렬.

최악의 시간 복잡도가 $O(?)$ 미만인 비교 기반 알고리즘은 구할 수 없다.

비교기반 정렬 VS 분포기반 정렬

- 비교기반(comparison-based) 기반 알고리즘
 - 선택(selection) 정렬, 버블(bubble) 정렬, 삽입(insertion) 정렬, 셸(shell) 정렬, 퀵(quick) 정렬, 합병(merge) 정렬, 힙(heap) 정렬.
- 분포기반(distribution-based) 정렬
 - 계수(counting) 정렬, 기수(radix)정렬, 버킷(bucket) 정렬.

키들의 비교가 아닌 키의 분포를 이용하면 최악 또는 평균 실행시간이 $O(?)$ 인 알고리즘을 구할 수 있다.

비교기반 정렬

- 비교기반 정렬 알고리즘들을 설명해보아라.
 - 선택(selection) 정렬
 - 버블(bubble) 정렬
 - 삽입(insertion) 정렬
 - 쉘(shell) 정렬
 - 퀵(quick) 정렬
 - 합병(merge) 정렬
 - 힙(heap) 정렬

비교기반 정렬

- 아래 입력에 대하여 각 정렬을 단계별 변화를 적어보시오.

30 20 40 10 5 10 30 15

- 선택(selection) 정렬
- 버블(bubble) 정렬
- 삽입(insertion) 정렬
- 셸(shell) 정렬
- 퀵(quick) 정렬
- 합병(merge) 정렬
- 히프(heap) 정렬

정렬

- 각 정렬의 제자리성, 안정성에 대하여 논하여 보아라
 - Ex) 버블정렬은 제자리 정렬인가?
 - Ex) 버블정렬은 안정된 정렬인가? 그이유는 무엇인가?
인접 레코드 끼리만 자리를 바꿈
- 최악의 시간 복잡도, 최선의 시간 복잡도.
 - Ex) 버블정렬이 최악의 시간 복잡도를 가지는 경우는 어떤 경우인가? 그 이유와 그 경우 시간복잡도에 대하여 논하여 보아라.
 - Ex) 삽입 정렬이 최선의 시간 복잡도를 가지는 경우는 어떤 경우인가? 그 이유와 그 경우 시간복잡도에 대하여 논하여 보아라

정렬

- 각 정렬의 제자리성, 안정성에 대하여 논하여 보아라
 - Ex) 버블정렬은 제자리 정렬인가?
 - Ex) 버블정렬은 안정된 정렬인가? 그이유는 무엇인가?
인접 레코드 끼리만 자리를 바꿈
- 최악의 시간 복잡도, 최선의 시간 복잡도.
 - Ex) 버블정렬이 최악의 시간 복잡도를 가지는 경우는 어떤 경우인가? 그 이유와 그 경우 시간복잡도에 대하여 논하여 보아라.
역순으로 이미 정렬된 레코드의 경우 $(n-1) + (n-2) + \dots + 1 = n(n-1)/2$ 비교를 하고 자리바꿈을 해야함. $O(n^2)$
 - Ex) 삽입 정렬이 최선의 시간 복잡도를 가지는 경우는 어떤 경우인가? 그 이유와 그 경우 시간복잡도에 대하여 논하여 보아라
이미 정렬된 레코드의 경우 $n-1$ 번의 비교만 하면 된다. $O(n)$

정렬

- 시간복잡도를 구해 보아라
- Ex) 퀵정렬의 최악의 경우시간 복잡도를 유도해 보아라.

힌트

- 한번의 QuickSort 호출에 의하여 분할원소 하나만 제자리를 잡고 나머지 원소 모두가 왼쪽 또는 오른쪽으로 부분 배열 되는 경우.
- 퀵정렬 알고리즘 실행시간 $T(n)$ 은, 이 경우 두 부분배열에 대한 QuickSort 순환 호출 중 하나는 상수 시간, 다른 하나는 $T(n-1)$ 시간이 걸림.
- Partition은 $\Theta(n)$ 걸림
- 시간복잡도에 상수는 무시할 수 있다.

$$T(n) = T(n-1) + \theta(n)$$

?

$$= \theta(n^2)$$

분포기반 정렬

- 주어진 A 에 대하여 계수 정렬의 단계를 보여라

	1	2	3	4	5	6	7	8
A	5	3	4	1	5	4	1	4

분포기반 정렬

- 주어진 숫자들에 대하여 기수정렬을 수행해 보아라.

567 654 124 457 830 911 555

특정 순서 원소 찾기

- 선택문제를 최악의 경우 $O(n)$ 으로 수행할 수 있는 알고리즘을 기술해 보아라.

균형적 다방향 합병정렬

- 원소수: n , 주기억장치에 읽을 수 있는 수: b , 4개의 테이프 T_0, T_1, T_2, T_3 를 이용하여 정렬시 균형적 다방향 합병정렬의 과정을 기술해 보아라. 아래 테이프로 시작.



외부정렬

- 외부정렬에서 효율성 높이기위해 보조기억장치와 주기억장치 사이의 데이터 이동 횟수 최소화하는 두가지 전략에 대하여 간략하게 논하고 각 어떤 외부정렬방법이 있는지 대표적인 두 방법을 적어 보아라.