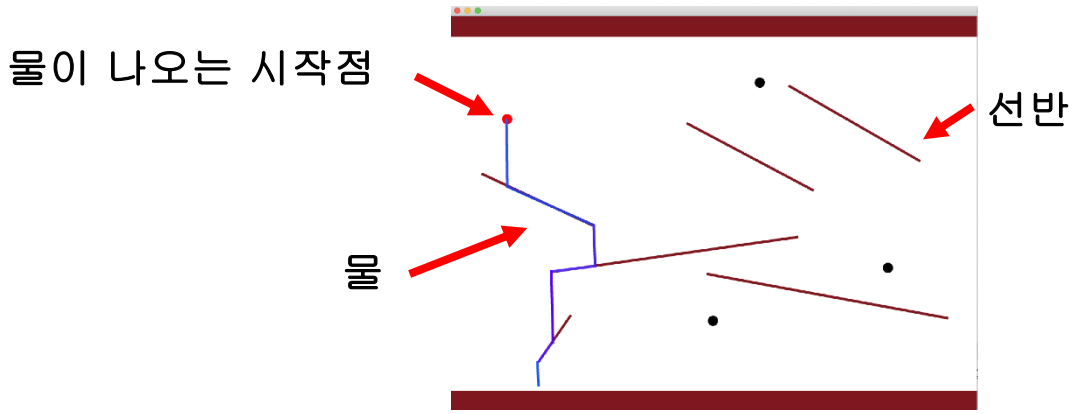
컴퓨터 공학 설계 및 실험 I
WaterFall – 2 week

10 주차 실험

2020.05.20

[실습] WaterFall Problem

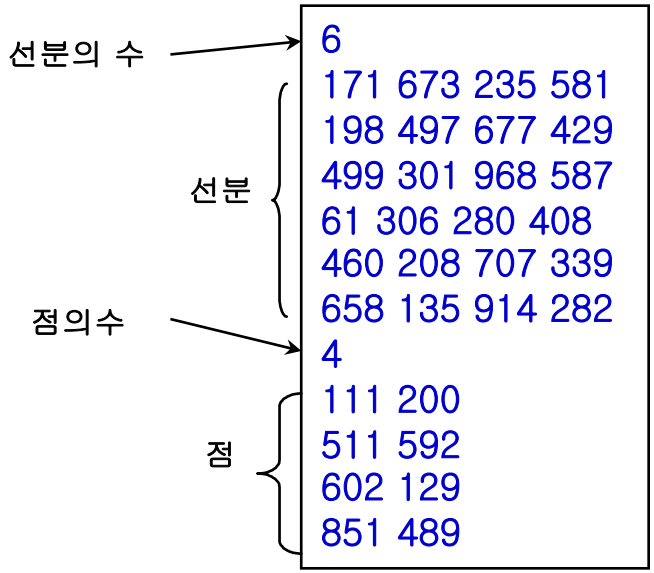
- WaterFall 문제
  - 물받이용 선반이 벽면에 임의로 놓여있을 때, 물이 떨어져 흐르는 경로를 계산하여 화면에 나타내는 문제.



  - 물받이를 선분으로 (직선이 아니라) 표시하고 다음과 같은 가정을 한다.
    - 1. 주어진 각 선분은 서로 교차하지 않는다.
    - 2. 선분들의 양 끝점의 y 좌표는 모두 0보다 크다. 즉, 모든 선분은 x 축 위에 있다.
    - 3. 물이 선분의 끝에 도달하면 바로 떨어진다고 가정한다. 즉, 관성의 법칙은 무시한다.
    - 4. 각 점은 임의의 위치에 존재할 수 있다. (단, x 축 제외). 즉, 구멍이 천장 뿐만 아니라 벽면에도 있을 수 있다고 가정한다.

[실습] WaterFall Problem (Cont')

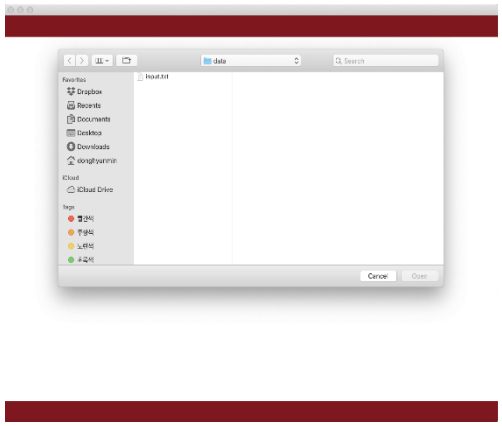
- 입력
  - 첫 번째 줄은 주어진 선분의 수 N(>= 0) 이다.
  - 두 번째 줄부터 N 개의 줄에는 각 선분의 양 끝 좌표가 x_l, y_l, x_r, y_r 순으로 표시된다. 여기서 (x_l, y_l)과 (x_r, y_r) 은 각각 선분의 왼쪽과 오른쪽 끝 좌표이다.
  - 선분에 대한 자료 다음 줄에는 점의 수 M 이 주어지고 그 다음 줄부터 M 개의 점에 대한 좌표 (x, y) 가 M 개의 줄로 표시된다. 만일 N=0 인 경우, 아무런 선분이 없다는 의미이고 따라서 바로 다음에 M 이 표시되게 한다.
  - Example.



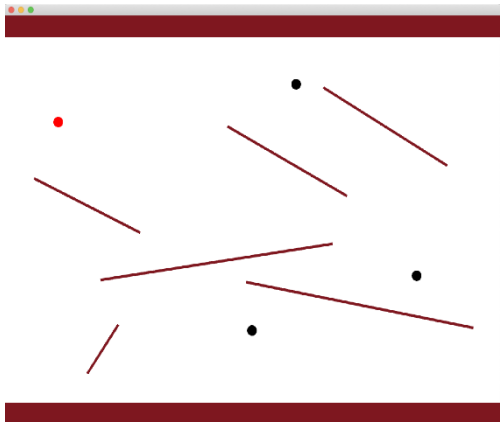
[실습] WaterFall Problem (Cont')

- 1 week 출력
  - 다음과 같은 Screen 을 생성하여 출력한다.
  - 'L' key 를 눌러서 input.txt 데이터를 입력으로 받는다. (Load)
  - 'D' key 를 눌러서 선분과 점을 각 해당하는 위치에 그린다. (Draw)
  - 왼쪽 '←' , 오른쪽 '→' 화살표를 통해 물이 흘러나올 점을 빨간색 표시하고 그 외의 점은 검은색 표시를 한다. (Selecting start point)
  - 'Q' key 를 눌러서 동적 할당된 메모리 해제 및 프로그램 종료 (Quit)
  - Example.

초기 화면



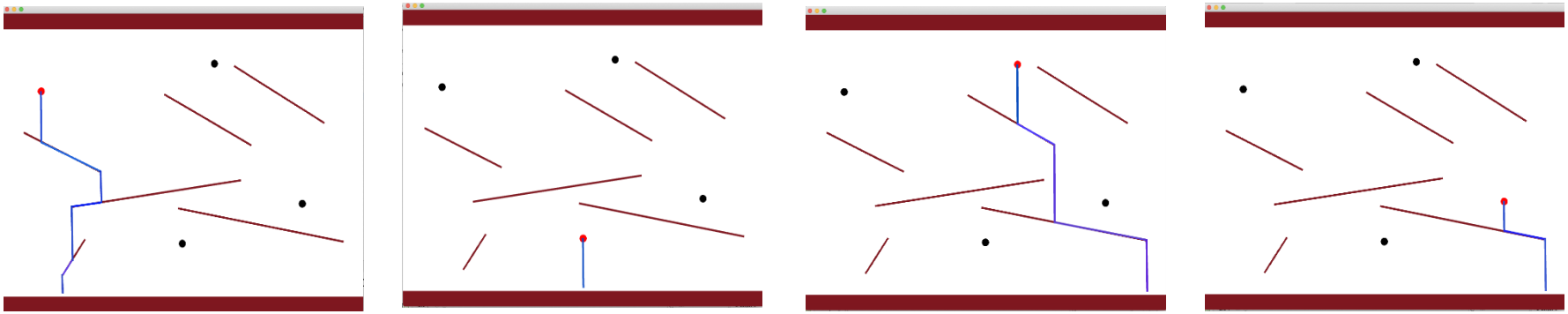
L 키를 누른 화면



D 키를 누른 화면

[실습] WaterFall Problem (Cont')

- 2 week 출력
  - 다음과 같은 Screen 을 생성하여 출력한다.
  - 'S' key 를 눌러서 선택된 위치에서 물을 흐르게 한다. (Start)
    - 물이 지나는 경로에 선분이 닿으면, 선분의 두 개의 끝 점 중 더 낮은 위치에 있는 쪽으로 물이 흐르도록 한다.
    - 물이 흐르는 동안에는 물이 시작하는 위치 (선택된 점)는 바꿀 수 없다.
  - 'E' key 를 눌러서 물을 멈추게 한다. (End)
  - 'Q' key 를 눌러서 동적 할당된 메모리 해제 및 프로그램 종료 (Quit)
  - Example.



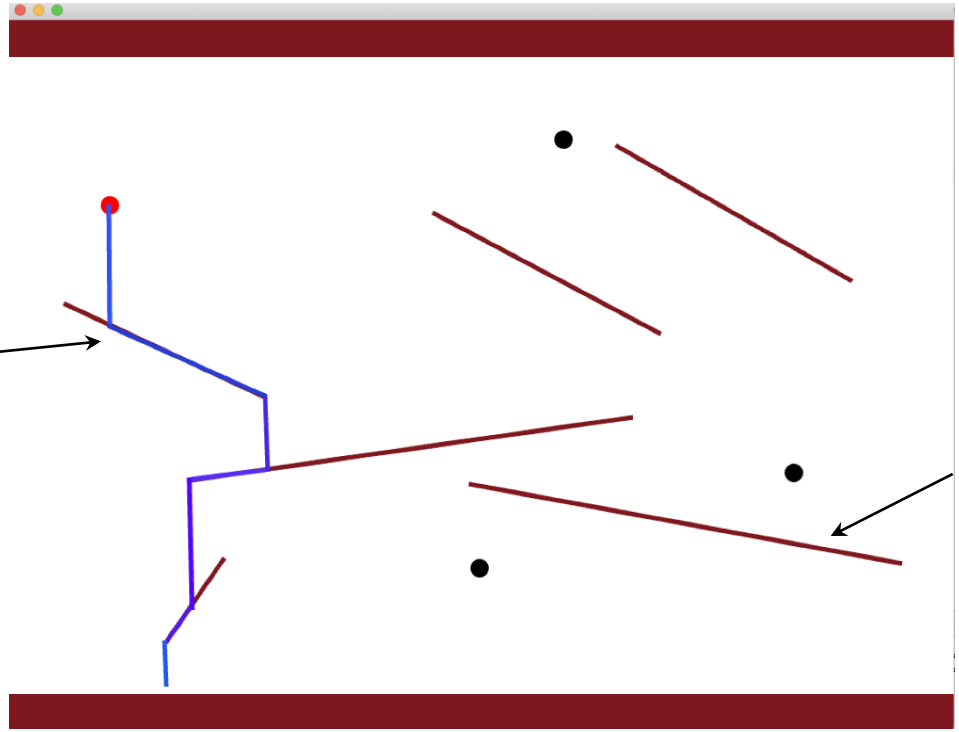
물이 흐르는 모습

Design For WaterFall

- 물 object 만들고(어떻게?), 물 흐르는 시작점 인지해서 물이 어떻게 흐를지 경로 계산해야 하고, screen 에 물도 같이 그려야 함.
- 다음의 고려 사항을 결정해야 한다.
  - 언제 어떻게 어디서 물 객체를 만들까?
    - 물을 표현하기 위해 class를 만들고, ofApp class 에서 객체화 시키고 초기화 하고 관리하는 것이 한 가지 방법.
    - Waterfall 정보를 받을 때나 setup 등 객체를 만드는 시점도 다양한 방법이 가능.
  - 물을 어떤 모양으로 그릴까?
    - 단순하게 사각형이나 선으로 여러 겹을 만드는 방법
    - 원으로 물방울 표현하는 방법
  - 각각의 물 객체의 흐르는 경로를 어떻게 계산하고 저장할까?
    - 경로 계산 알고리즘 (다음 슬라이드 참고)
    - 물을 선으로 표현한다면, 시작점, 선분과 맞닿는 점, 선분 끝 점 등 점들의 좌표만 저장하고 연결하면 됨.
    - 물을 물방울로 표현하는 경우는 update() 함수를 써서 물방울 마다 위치를 업데이트 시키면 되고 선으로 하는 경우와 달리, 경로 저장을 따로 할 필요 없음.

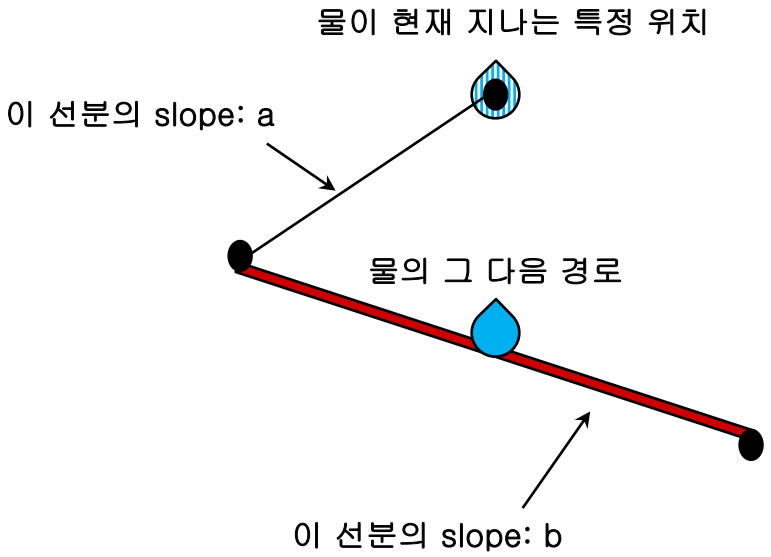
How To Calculate The Path

- 선 (line) 으로 물을 표현하는 경우를 가정한다.
- 1) 물이 흐르는 시작점 보다 더 높이 있는 선분은 절대 만나지 않으므로 더 높이 있는 선분은 계산과정에서 제외시킨다.
- 2) 물이 수직으로 떨어지기 시작할 때, 나머지 아래에 있는 선분들 중 어떤 선분이 맞닿게 될지를 확인한다.

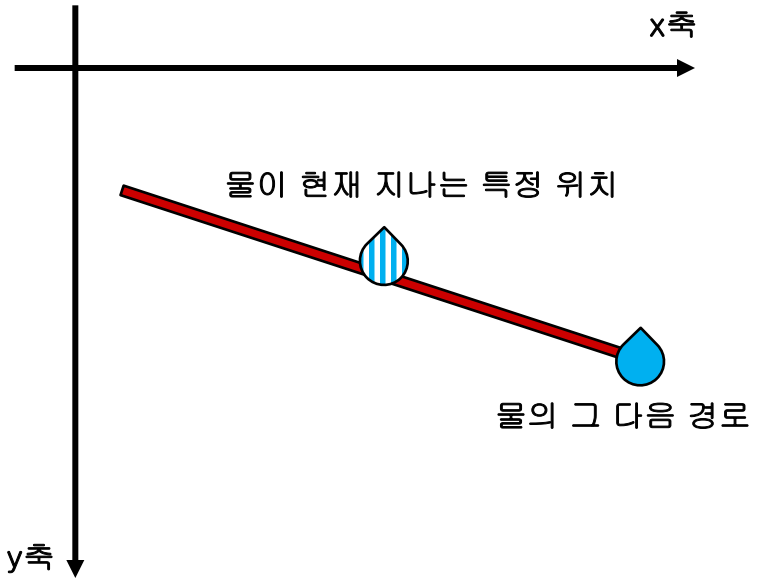


How To Calculate The Path (Cont')

- 3) 물과 결국 만나게 되는 선분들 중 가장 먼저 만나는 선분을 찾는다.
- 4) a 와 b 가 같거나 굉장히 유사하게 (≤ EPSILON) 되는 순간의 물이 지나는 위치를 기록. (<slope 비교> 그림 참고)
- 5) 선분에 닿는 경우, 선분의 양 끝점 중, y 좌표가 더 높은 쪽을 갖는 끝점을 다음 경로로 선택.(<선분 내에서 다음 경로 찾기> 그림 참고)
- 6) ofGetHeight() 에 도달할 때 까지 반복.



<slope 비교>



<선분 내에서 다음 경로 찾기>

결과 레포트

- 물이 흐르는 것을 표현하기 위해서, 본인이 구현한 알고리즘과 자료구조를 기술한다.

제출 방식

☐ 실습 제출물

- main.cpp, ofApp.cpp, ofApp.h 를 압축하여 제출

☐ 제출 방식

- 첨부 파일(과제제출물 압축해서 첨부) :

  컴실1_X주차_실습_학번.word(또는 hwp)

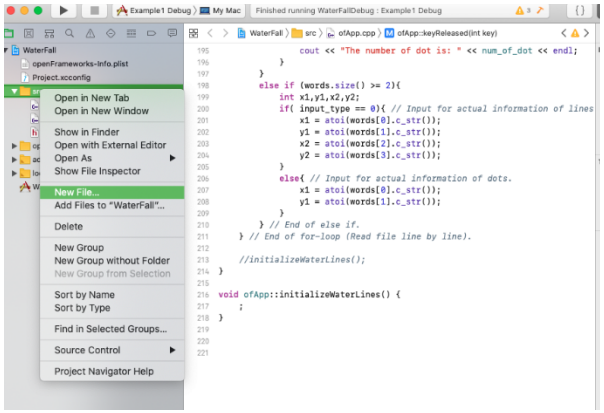
  컴실1_X주차_실습_학번.zip

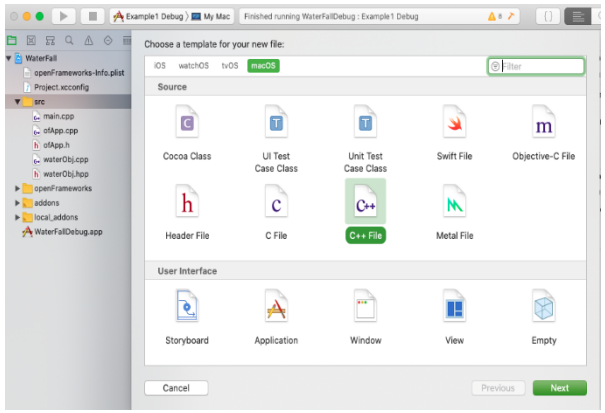
  Ex > 컴실1_10주차_실습_20190000.zip

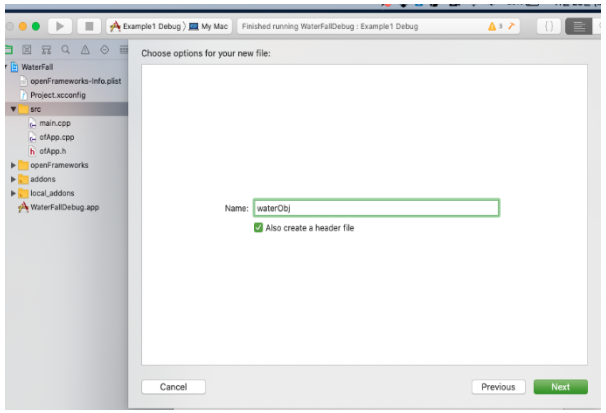
형식 틀릴 시 감점!

Appendix: waterObj.h, waterObj.cpp 생성

- waterObj.h 에서 물 객체에 대한 인스턴스 만들기 위해 waterObj 라는 class 를 선언한다.
- waterObj.cpp 에서 waterObj class 에 대해 함수 등을 정의한다.
- 아래와 같이 waterObj.h, cpp 를 만든다.







- src 디렉토리 안에 waterObj 에 대한 헤더, cpp 파일이 만들어진 것을 확인.