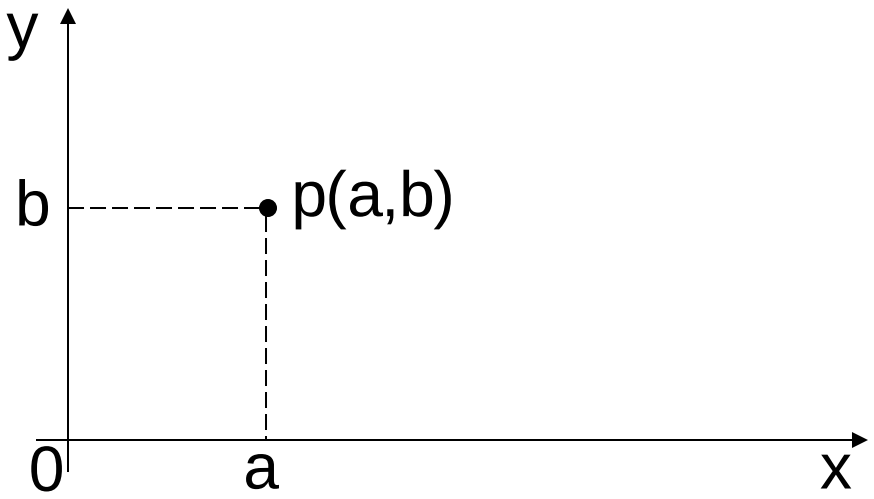
소프트웨어와 문제해결

점(Points)

- 평면에서 두 개의 좌표정보에 의해 정의되는 도형

p = [a, b]



거리(Distance)

- 두 점을 잇는 선분의 길이

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

```
import math
p1 = [4, 0]
p2 = [6, 6]
def distance(p1, p2) {
    distance = math.sqrt( ((p1[0]-p2[0])**2)+((p1[1]-p2[1])**2) )
}
```

선(Lines)

- 점의 이동의 궤적
- 직선의 표현법
  - 방정식으로 표현
    - $y = ax + b$
    - a : 기울기, b : y절편 값
  - 두 점의 쌍 (x_1, y_1) 와 (x_2, y_2)로 표현
    - $(y_1-y_2)/(x_1-x_2)$: 기울기
- 일반적인 직선의 방정식
  - $ax + by + c = 0$

선(Lines)

- 방정식으로 표현하는 방법

  a = x 계수

  b = y 계수

  c = 상수

x축에 평행한 직선의 방정식

y = k (단, k != 0), 이때, a = 0

y축에 평행한 직선의 방정식

x = k (단, k != 0), 이때, b = 0

선(Lines)

– 두 점의 쌍 p1=[x_1, y_1], p2=[x_2, y_2]으로 표현하는 방법

```
def points_to_line(p1, p2) {
        if (p1[0] == p2[0]) {
                a = 1;
                b = 0;
                c = -p1[0];
        } else {
                b = 1;
                a = -(p1[1] – p2[1]) / (p1[0] – p2[0]);
                c = -(a * p1[0]) – (b * p1[1]);
        }
}
```

p1.x 와 p2.x는 아주 근소한 값의 차이를 가질 수 있다.

(math.fabs(p1.x - p2.x) <= sys.float_info.epsilon) 이때, epsilon은 매우 작은 양수

EPSILON 값

- 0.1 + 0.2의 값은?

- 0.1 + 0.2 == 0.3 의 값은?

```
>>> import math, sys
>>> x = 0.1 + 0.2
>>> math.fabs(x - 0.3) <= sys.float_info.epsilon
True
```

선(Lines)

- 한 점과 기울기가 주어졌을 때

```
def point_and_slope_to_line(p, double s) {
        a = -s;
        b = 1;
        c = -( (a * p[0]) + (b * p[1]) );
}
```

실습 1 : 일직선 찾기

- 세 점이 주어질 때 세 점이 일직선 상에 있는가를 알아내는 프로그램
- **입력**
  - (x,y) 좌표 세 쌍이 한 줄에 하나씩 입력으로 주어진다. 모든 수는 정수이다.
- **출력**
  - 일 직선 상에 있으면 yes , 아니면 no 를 출력한다.
- **입출력 예**
  - 입력
    0 1 1 3 2 5
  - 출력
    yes

선의 교차(Line Intersection)

- 2 차원에서의 두 직선의 관계
  - 평행, 일치, 한 점에서 만남

```
def parallelQ(m1, m2):  /* 두 개의 선이 평행인지 확인 m1 = [a, b, c] */
  return ( math.fabs((m1[0]/m1[1]) - (m2[0]/m2[1])) <= sys.float_info.epsilon)
```

```
def same_lineQ(m1, m2):   /* 두 개의 선이 일치하는지 확인 */
  return(parallelQ(m1,m2) and math.fabs((m1[2]/m1[1]) - (m2[2]/m2[1])) <=
  sys.float_info.epsilon)
```

선의 교차(Line Intersection)

```
def intersection_point(m1, m2): /* 두 직선이 만나는 점을 찾음 */
    p = [0.0, 0.0]     //초기화
    if same_lineQ(m1, m2)):
        print("Warning: identical lines.\n")
        p = [0.0, 0.0]

    if parallelQ(m1, m2):
        print("Error: Distinct parallel lines do not intersect.\n")

    p[0] = (m2[1] * m1[2] – m1[1] * m2[2]) / (m2[0] * m1[1] – m1[0] * m2[1])

    if math.fabs(m1[1]) > sys.float_info.epsilon:/* 수직 선의 경우*/
        p[1] = - (m1[0] * p[0] + m1[2] ) / m1[1]
    else:
        p[1] = - (m2[0] * p[0] + m2[2] ) / m2[1]
```

실습 2 : 교차하는 모든 점 찾기

- N개의 직선의 정보가 두 정점으로 주어질 때 직선 중에서 서로 교차하는 모든 점을 찾아 출력하기
- 입력
  - 3 // 직선 개수
  - 1.0 1.0 1.0 2.0
  - 2.0 2.0 3.0 3.0
  - 1.0 2.0 2.0 4.0
- 출력
  - (1,2) : (1.0, 1.0)
  - (1,3) : (1.0, 2.0)
  - (2,3) : (0.0, 0.0)

선분(Line Segments)

- 직선상의 두 점과 그 사이의 점으로 구성되는 유한인 직선의 부분

```
/* 선분의 끝점들 */
p1 = [x1, y1]
P2 = [x2, y2]
```

실습 3 : 선분의 교차 여부 확인하기

- N개의 선분의 정보가 두 정점으로 주어질 때 선분 중에서 서로 교차하는 선분을 모두 찾아 출력하기
- 입력
  - 3 // 선분 개수
  - 1.0 1.0 1.0 2.0
  - 2.0 2.0 3.0 3.0
  - 0.5 1.0 2.0 4.0
- 출력
  - (1,3)

실습 4: 괄호쌍 찾기

- 짝이 맞지 않는 괄호 문제
  - 수식들의 괄호가 쌍이 맞는지 확인하는 프로그램을 작성하시오.
    - 둥근괄호는 (로 열고,)로 닫음
    - 중괄호는 {로 열고, }로 닫음
    - 대괄호는 [로 열고,]로 닫음
  - 조건
    - 모든 괄호는 해당하는 짝이 있어야 함
    - 모든 괄호 쌍은 먼저 열린 뒤 닫힘
    - 한 괄호 쌍이 다른 괄호쌍과 서로 교차해 있으면 안됨
      - 예: [(])

실습 4: 괄호쌍 찾기

- 입력
  - n, 총 입력 갯수
  - 1번째 입력
  - 2번째 입력
  - n번째 입력
- 출력
  - 각 경우에 대해 YES or NO

- 입력
  - 3
  - ()()
  - ({[]})
  - ({}[(){}])
- 출력
  - YES
  - NO
  - YES

실습 5 - 가위바위보 게임

- 사용자와 가위바위보 게임을 하는 프로그램에서, 내가 만든 프로그램이 가위, 바위, 보를 무엇을 낼지를 결정하는 알고리즘을 작성하고자 한다. 다음 요구사항에 따라 코드를 작성하시오.
- 파이썬에서 난수 생성 방법

```
from random import *
i = randint(1, 100)
```

실습 5 - 가위바위보 게임

- 문제 1) 0부터 X사이의 정수값을 반환하는 함수 randint()를 이용하여 가위, 바위, 보를 동일한 확률로 내는 알고리즘을 작성하시오
- 문제 2) 0부터 X사이의 정수값을 반환하는 함수 randint(0, X)를 이용하여 가위, 바위, 보를 각각 0.5, 0.3, 0.2의 확률로 내는 알고리즘을 작성하시오.
- 문제 3) 대전을 하는 상대를 관찰한 결과, 그 상대는 10번 가위바위보를 할 때 가위를 5번, 바위를 2번, 보를 3번을 낸다는 사실을 알아내었다. 이 상대와 대전을 할 때, 내가 작성한 프로그램이 이길 확률을 평균 0.4 정도로 유지하기 위해서는 가위, 바위, 보를 어떤 확률로 내도록 알고리즘을 만들어야 할지 결정하시오.

- 의사코드, 순서도, 프로그램을 모두 작성하시오.