Ch04: Turtle 모듈을 사용한 x,y 좌표 그리기

[연습] x,y 좌표 그리기

x, y 좌표축 그리기

함수호출 사용

(0, 0)을 기준으로 그리기

> Turtle Speed : 3

> Window Size : (0, 0) 기준 상하좌우로 300 pixel

In [3]:
```
## [연습] x, y 좌표 그래프 그리기: x, y 좌표축 그리기
import turtle as t
t.speed(3)

##[함수] (x1, y1) - (x2, y2) 직선 그리기
def line(x1, y1, x2, y2):
    t.up()
    t.goto(x1, y1)
    t.down()
    t.goto(x2, y2)

#x, y 좌표축 그리기
line(-300, 0, 300, 0)   # x축 직선 그리기
line(0, -300, 0, 300)   # y축 직선 그리기

t.exitonclick()  # 실행 창을 닫지 않도록
```

In [ ]:
```

```

좌표축 눈금 그리기

100 pixel 마다 그리기

In [7]:
```
## [연습] x, y 좌표 그래프 그리기: 좌표축 눈금 그리기
import turtle as t
t.speed(3)

##[함수] (x1, y1) - (x2, y2) 직선 그리기
def line(x1, y1, x2, y2):
    t.up()
    t.goto(x1, y1)
    t.down()
    t.goto(x2, y2)

#x, y 좌표축 그리기
line(-300, 0, 300, 0)   # x축 직선 그리기
line(0, -300, 0, 300)   # y축 직선 그리기

#좌표축 눈금 그리기
for i in range(-300, 300+1, 100):
    line(i, -5, i, 5)      # x축 눈금
for i in range(-300, 300+1, 100):
    line(-5, i, 5, i)      # y축 눈금

t.exitonclick()  # 실행 창을 닫지 않도록
```

In [ ]:
```

```

눈금에 값 쓰기

함수호출 사용

> def txtwrite(x, y, text):

Turtle 감추기 : turtle.hideturtle()

In [13]:
```
## [연습] x, y 좌표 그래프 그리기: 눈금에 값 쓰기
import turtle as t
t.speed(3)

##[함수] (x1, y1) - (x2, y2) 직선 그리기
def line(x1, y1, x2, y2):
    t.up()
    t.goto(x1, y1)
    t.down()
```

```
    t.goto(x2, y2)

##[함수] (x,y)에 텍스트 쓰기
def txtwrite(x, y, text):
    t.up()
    t.goto(x, y)
    t.down()
    t.write(text)

#x, y 좌표축 그리기
line(-300, 0, 300, 0)   # x축 직선 그리기
line(0, -300, 0, 300)   # y축 직선 그리기

#좌표축 눈금 그리기
for i in range(-300, 300+1, 100):
    line(i, -5, i, 5)      # x축 눈금
for i in range(-300, 300+1, 100):
    line(-5, i, 5, i)      # y축 눈금

#좌표 눈금 숫자 표기하기
for i in range(-300, 300+1, 100):
    line(i, -5, i, 5)          # x축 눈금
    if i != 0:  #(0, 0)은 제외
        txtwrite(i-10, -20, i)
for i in range(-300, 300+1, 100):
    line(-5, i, 5, i)          # y축 눈금
    if i != 0:
        txtwrite(20, i-10, i)

t.exitonclick()  # 실행 창을 닫지 않도록
```

In [ ]:

함수 호출로 x,y 좌표 그리기

좌표의 **x, y**축 직선과 눈금, 눈금 값을 그린다.

> **(0, 0)** 기준 좌표 크기 **wsize** 값 전달

> 눈금 간격 **step** 값 전달

In [15]:
```
## [연습] x, y 좌표 그래프 그리기: 함수 호출로 해결
import turtle as t
```

```
t.speed(3)
#t.hideturtle()

##[함수] (x1, y1) - (x2, y2) 직선 그리기
def line(x1, y1, x2, y2):
    t.up()
    t.goto(x1, y1)
    t.down()
    t.goto(x2, y2)

##[함수] (x,y)에 텍스트 쓰기
def txtwrite(x, y, text):
    t.up()
    t.goto(x, y)
    t.down()
    t.write(text)

##[함수] x, y축 좌표 그리기
def draw_xy(wsize, step, hide=0) :
    if hide :
        t.hideturtle()
    line(-wsize, 0, wsize+1, 0)    # x축 라인
    line(0, -wsize, 0, wsize)      # y축 라인

    for i in range(-wsize, wsize+1, step) :
        line(i, -5, i, 5)      # x축 눈금
        if i != 0:
            txtwrite(i-10, -20, i)
    for i in range(-wsize, wsize+1, step) :
        line(-5, i, 5, i)      # y축 눈금
        if i != 0:
            txtwrite(10, i-5, i)

wsize = 300 #좌우 최대 눈금값
step = 100  #눈금 간격
draw_xy(wsize, step)

t.exitonclick()  # 실행 창을 닫지 않도록
```

In [ ]:

[응용실습] 1차 방정식의 좌표 그래프 그리기

y = ax + b 함수 그래프를 그리는 프로그램 완성

> Turtle Speed : 0

> Window Size : 500 * 500 pixel

> 좌표축 눈금 간격 크기 : 100pixel 당 1 또는 50pixel 당 1로 축소하여 사용

In [17]:
```
## [실습] 2차 방정식의 좌표 그래프 그리기: 우측면 그리기
import turtle as t

##[함수] (x1, y1) - (x2, y2) 직선 그리기
def line(x1, y1, x2, y2):
    t.up()
    t.goto(x1, y1)
    t.down()
    t.goto(x2, y2)

##[함수] (x,y)에 텍스트 쓰기
def txtwrite(x, y, text):
    t.up()
    t.goto(x, y)
    t.down()
    t.write(text)

def newx(x):
    global step
    return x / step

def newy(y):
    global step
    return y / step

##[함수] x, y축 좌표 그리기
def draw_xy(wsize, step, hide=0) :
    if hide :
        t.hideturtle()
    line(-wsize, 0, wsize, 0)      # x축 라인
    line(0, -wsize, 0, wsize)      # y축 라인

    for i in range(-wsize, wsize+1, step) :
        line(i, -5, i, 5)      # x축 눈금
```

```
        txtwrite(i-5, -20, newx(i))
    for i in range(-wsize, wsize+1, step) :
        line(-5, i, 5, i)      # y축 눈금
        if i != 0:
            txtwrite(10, i-5, newy(i))

## x, y축 좌표 그리기
t.speed(0)
wsize = 300    #+축 크기 pixel
step = 50      #눈금 간격 pixel(1로 표시되는 길이의 pixel값)
direc = 2      #좌표 그리는 주기(default 2보다 크면 성성하게 그림)

## y = a*x + b
inlist = [int(x) for x in input(">ax + b의 a b 입력: ").split()]
a = inlist[0]
b = inlist[1]
draw_xy(wsize, step)

# 우측 면축에 좌표 그리기
for i in range(0, wsize+1, direc) :
    x1 = newx(i)        #다음 그릴 지점(i/step)
    y1 = a * x1 + b
    x2 = x1 + newx(1)   #그릴 종점 지점(+1/step)
    y2 = a * x2 + b
    if (abs(x1*step) > abs(wsize)) or (abs(y1*step) > abs(wsize)) :
        break      #window 초과 시 나가기
    line(x1*step, y1*step, x2*step, y2*step)

t.exitonclick()  # 실행 창을 닫지 않도록
```

In [9]:

```
## [실습] 2차 방정식의 좌표 그래프 그리기: 좌우측면 그리기
import turtle as t

##[함수] (x1, y1) - (x2, y2) 직선 그리기
def line(x1, y1, x2, y2):
    t.up()
    t.goto(x1, y1)
    t.down()
    t.goto(x2, y2)

##[함수] (x,y)에 텍스트 쓰기
def txtwrite(x, y, text):
```

```
    t.up()
    t.goto(x, y)
    t.down()
    t.write(text)

def newx(x):
    global step
    return x / step

def newy(y):
    global step
    return y / step

##[함수] x, y축 좌표 그리기
def draw_xy(wsize, step, hide=0) :
    if hide :
        t.hideturtle()
    line(-wsize, 0, wsize, 0)      # x축 라인
    line(0, -wsize, 0, wsize)      # y축 라인

    for i in range(-wsize, wsize+1, step) :
        line(i, -5, i, 5)      # x축 눈금
        txtwrite(i-5, -20, newx(i))
    for i in range(-wsize, wsize+1, step) :
        line(-5, i, 5, i)      # y축 눈금
        if i != 0:
            txtwrite(10, i-5, newy(i))

def draw_fn(wsize, step, direc=2, a=1, b=0) :
    for i in range(0, wsize+1, direc) :
        x1 = newx(i)        #i/step
        y1 = a * x1 + b
        x2 = x1 + newx(1)   #(1/step)
        y2 = a * x2 + b
        if (abs(x1*step) > abs(wsize)) or (abs(y1*step) > abs(wsize)) :
            break      #window 초과 시 나가기
        line(x1*step, y1*step, x2*step, y2*step)

## x, y축 좌표 그리기
t.speed(0)
wsize = 300    #+축 크기 pixel
step = 50      #눈금 간격 pixel(1로 표시되는 길이의 pixel값)
direc = 2      #좌표 그리는 주기(default 2보다 크면 성성하게 그림)
```

```
## y = a*x + b
inlist = [int(x) for x in input(">ax + b의 a b 입력: ").split()]
a = inlist[0]
b = inlist[1]
draw_xy(wsize, step)

# 우측 면축에 좌표 그리기
for i in range(0, wsize+1, direc) :
    x1 = newx(i)        #다음 그릴 지점(i/step)
    y1 = a * x1 + b
    x2 = x1 + newx(1)   #그릴 종점 지점(+1/step)
    y2 = a * x2 + b
    if (abs(x1*step) > abs(wsize)) or (abs(y1*step) > abs(wsize)) :
        break      #window 초과 시 나가기
    line(x1*step, y1*step, x2*step, y2*step)

# 좌측 면축에 좌표 그리기
for i in range(0, -wsize+1, -direc) :
    x1 = newx(i)        #다음 그릴 지점(i/step)
    y1 = a * x1 + b
    x2 = x1 + newx(1)   #그릴 종점 지점(+1/step)
    y2 = a * x2 + b
    if (abs(x1*step) > abs(wsize)) or (abs(y1*step) > abs(wsize)) :
        break      #window 초과 시 나가기
    line(x1*step, y1*step, x2*step, y2*step)

t.exitonclick()  # 실행 창을 닫지 않도록
```

In [ ]:

[응용실습] 1차 방정식의 좌표 그래프 그리기

y = ax + b 함수 그래프를 그리는 프로그램 완성

함수 호출로 해결

> Turtle Speed : 0

> Window Size : 500 * 500 pixel

> 좌표축 눈금 간격 크기 : 100pixel 당 1 또는 50pixel 당 1로 축소하여 사용

In [5]:

```
## [실습] 2차 방정식의 좌표 그래프 그리기: 함수 호출로 해결
import turtle as t

##[함수] (x1, y1) - (x2, y2) 직선 그리기
def line(x1, y1, x2, y2):
    t.up()
    t.goto(x1, y1)
    t.down()
    t.goto(x2, y2)

##[함수] (x,y)에 텍스트 쓰기
def txtwrite(x, y, text):
    t.up()
    t.goto(x, y)
    t.down()
    t.write(text)

def newx(x):
    global step
    return x / step

def newy(y):
    global step
    return y / step

##[함수] x, y축 좌표 그리기
def draw_xy(wsize, step, hide=0) :
    if hide :
        t.hideturtle()
    line(-wsize, 0, wsize, 0)      # x축 라인
    line(0, -wsize, 0, wsize)      # y축 라인

    for i in range(-wsize, wsize+1, step) :
        line(i, -5, i, 5)      # x축 눈금
        txtwrite(i-5, -20, newx(i))
    for i in range(-wsize, wsize+1, step) :
        line(-5, i, 5, i)      # y축 눈금
        if i != 0:
            txtwrite(10, i-5, newy(i))

def draw_fn(wsize, step, direc=2, a=1, b=0) :
    for i in range(0, wsize+1, direc) :
        x1 = newx(i)        #i/step
```

```
        y1 = a * x1 + b
        x2 = x1 + newx(1)   #(1/step)
        y2 = a * x2 + b
        if (abs(x1*step) > abs(wsize)) or (abs(y1*step) > abs(wsize)) :
            break      #window 초과 시 나가기
        line(x1*step, y1*step, x2*step, y2*step)

## x, y축 좌표 그리기
t.speed(0)
wsize = 300    #+축 크기 pixel
step = 50      #눈금 간격 pixel(1로 표시되는 길이의 pixel값)
direc = 2      #좌표 그리는 주기(default 2보다 크면 성성하게 그림)

## y = a*x + b
inlist = [int(x) for x in input(">ax + b의 a b 입력: ").split()]
a = inlist[0]
b = inlist[1]
draw_xy(wsize, step)
draw_fn(wsize, step, direc, a, b)       #(+)축 그리기
draw_fn(-wsize, step, -direc, a, b)     #(-)축 그리기

t.exitonclick()  # 실행 창을 닫지 않도록
```

In [ ]:

[응용실습] 2차 방정식의 좌표 그래프 그리기

y = ax2 + bx +c 함수 그래프를 그리는 프로그램 완성

> Turtle Speed : 0

> Window Size : 500 * 500 pixel

> 좌표축 눈금 간격 크기 : 100pixel 당 1 또는 50pixel 당 1로 축소하여 사용

y = ax2 + bx +c 형식의 그래프를 그릴 수 있도록 a, b, c 변수를 함수로 넘기는 방식을 적용해보세요.

In [1]:
```
## [실습] 2차 방정식의 좌표 그래프 그리기
import turtle as t

##[함수] (x1, y1) - (x2, y2) 직선 그리기
def line(x1, y1, x2, y2):
```

```
    t.up()
    t.goto(x1, y1)
    t.down()
    t.goto(x2, y2)

##[함수] (x,y)에 텍스트 쓰기
def txtwrite(x, y, text):
    t.up()
    t.goto(x, y)
    t.down()
    t.write(text)

def newx(x):
    global step
    return x / step

def newy(y):
    global step
    return y / step

##[함수] x, y축 좌표 그리기
def draw_xy(wsize, step, hide=0) :
    if hide :
        t.hideturtle()
    line(-wsize, 0, wsize, 0)      # x축 라인
    line(0, -wsize, 0, wsize)      # y축 라인

    for i in range(-wsize, wsize+1, step) :
        line(i, -5, i, 5)      # x축 눈금
        txtwrite(i-5, -20, newx(i))
    for i in range(-wsize, wsize+1, step) :
        line(-5, i, 5, i)      # y축 눈금
        if i != 0:
            txtwrite(10, i-5, newy(i))

def draw_fn(wsize, step, direc, a, b, c) :
    for i in range(0, wsize+1, direc) :
        x1 = newx(i)    #i / step
        y1 = a * (x1 * x1) + b * (x1) + c
        x2 = x1 + newx(1)   #(1/step)
        y2 = a * (x2 * x2) + b * (x2) + c
        if (abs(x1*step) > abs(wsize)) or (abs(y1*step) > abs(wsize)) :
            break      #window 초과 시 나가기
        line(x1*step, y1*step, x2*step, y2*step)
```

```
## x, y축 좌표 그리기
t.speed(0)
wsize = 300    #+축 크기 pixel
step = 50      #눈금 간격 pixel(1로 표시되는 길이의 pixel값)

## y = a*(x*x) + b*(x) + c
inlist = [int(x) for x in input(">ax**2 + bx + c의 a b c 입력: ").split()]
a = inlist[0]
b = inlist[1]
c = inlist[2]
draw_xy(wsize, step)
draw_fn(wsize, step, 2, a, b, c)       #(+)축 그리기
draw_fn(-wsize, step, -2, a, b, c)     #(-)축 그리기

t.exitonclick()  # 실행 창을 닫지 않도록
```

In [ ]: