Functions Continued...

Prog

```
def fn( x ):
    return x * x
```

parameter

return value

Def. of a fn.

```
SomeVariable(s) = fn(arguments)
y = fn(10)
```

calling the fn

Maths

$f(x) = x^2$ / x * x

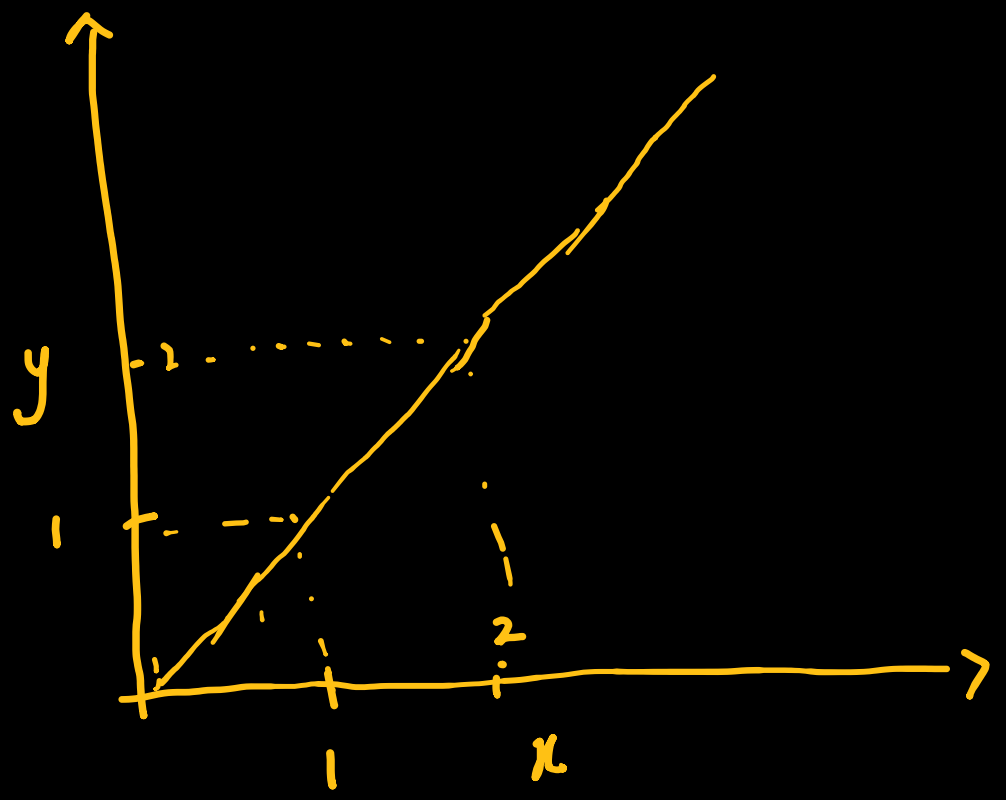
Definition

$y = f(10)$

y becomes 100

calling.

$y = f(x)$

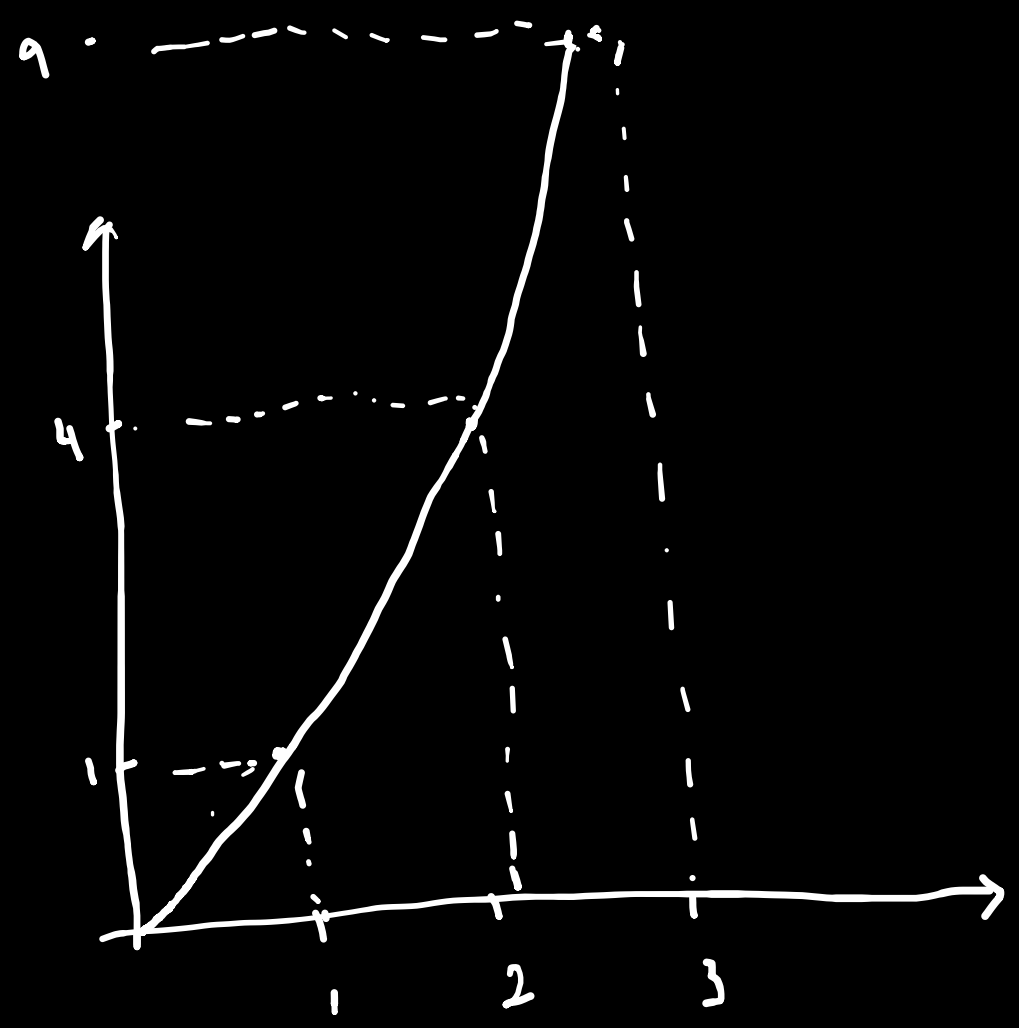



```
def f(x):
    return x
```

$y = f(1)$ ← 1

$y = f(2)$ ← 2

$y = f(3)$ ← 3

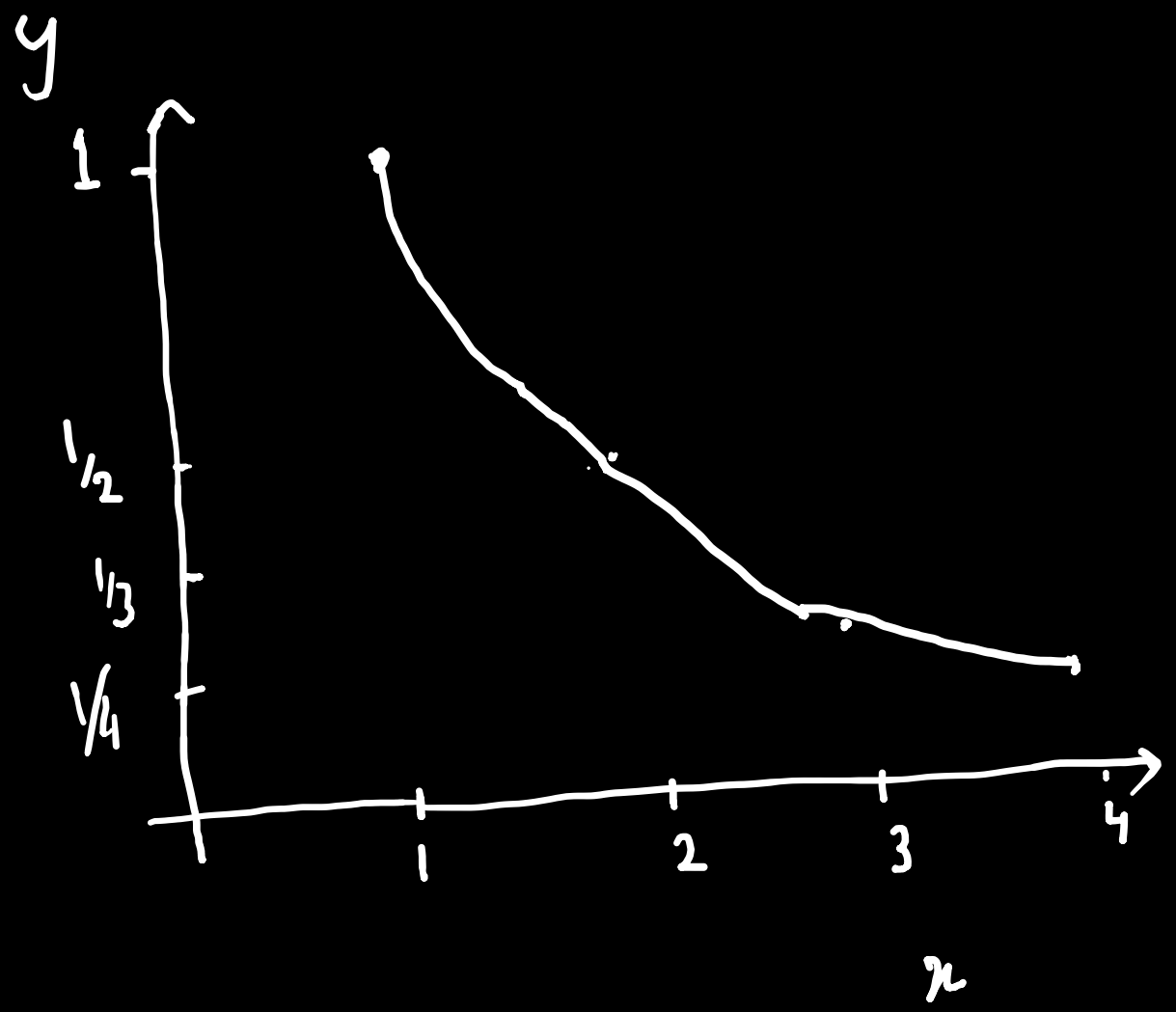



```
def g(x):
    return x * x
```

$y = g(1) \leftarrow 1$

$y = g(2) \leftarrow 4$

$y = g(3) \leftarrow 9$



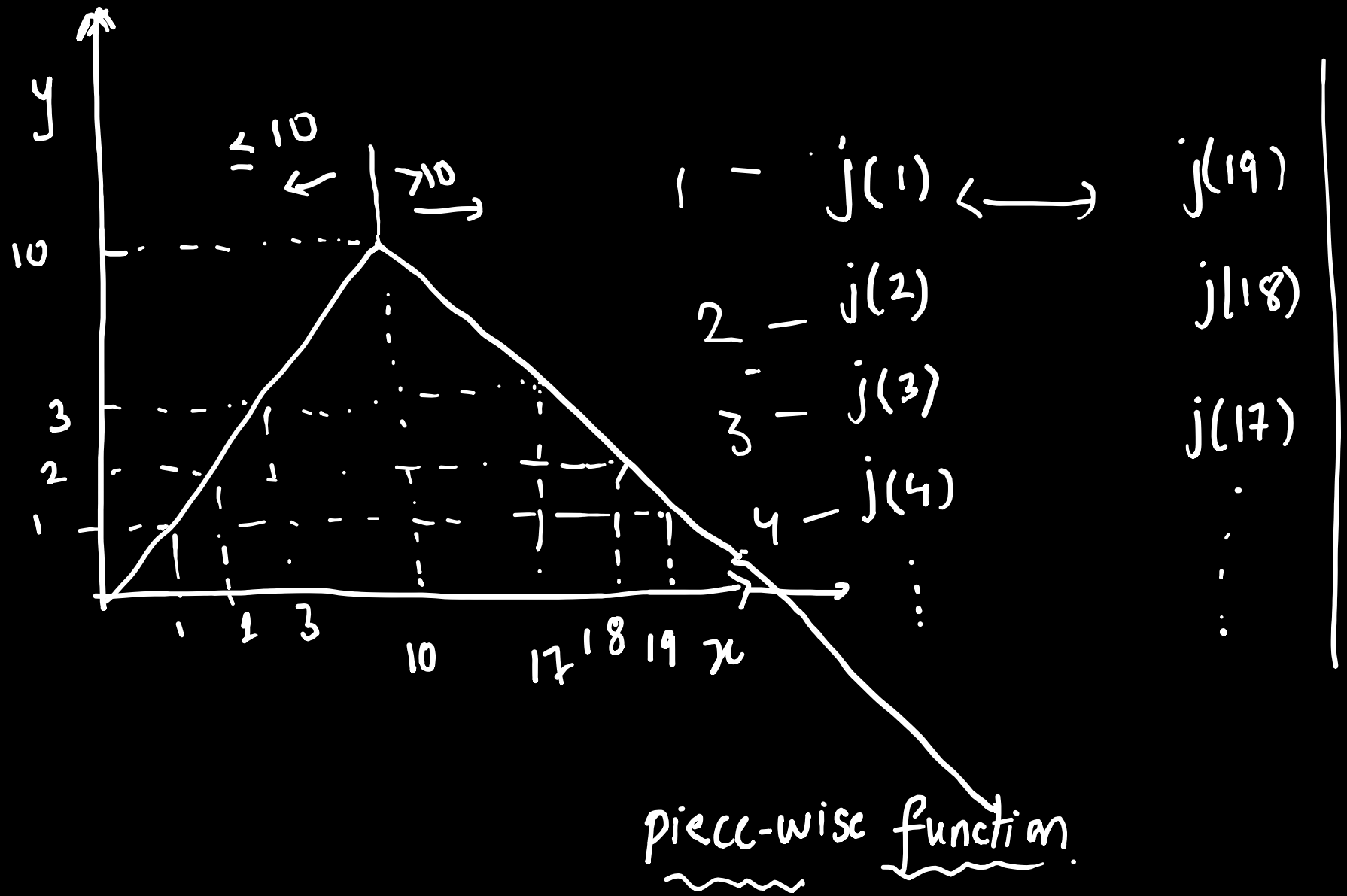

```
def h(x):
    return (1/x)
```

$y = h(1) \leftarrow 1 \quad (1/1)$

$y = h(2) \leftarrow 0.5 \quad (1/2)$

$y = h(3) \leftarrow 0.3333 \quad (1/3)$

$y = h(4) \leftarrow 0.25 \quad (1/4)$



1 — j(1) ⟷ j(19)

2 — j(2) j(18)

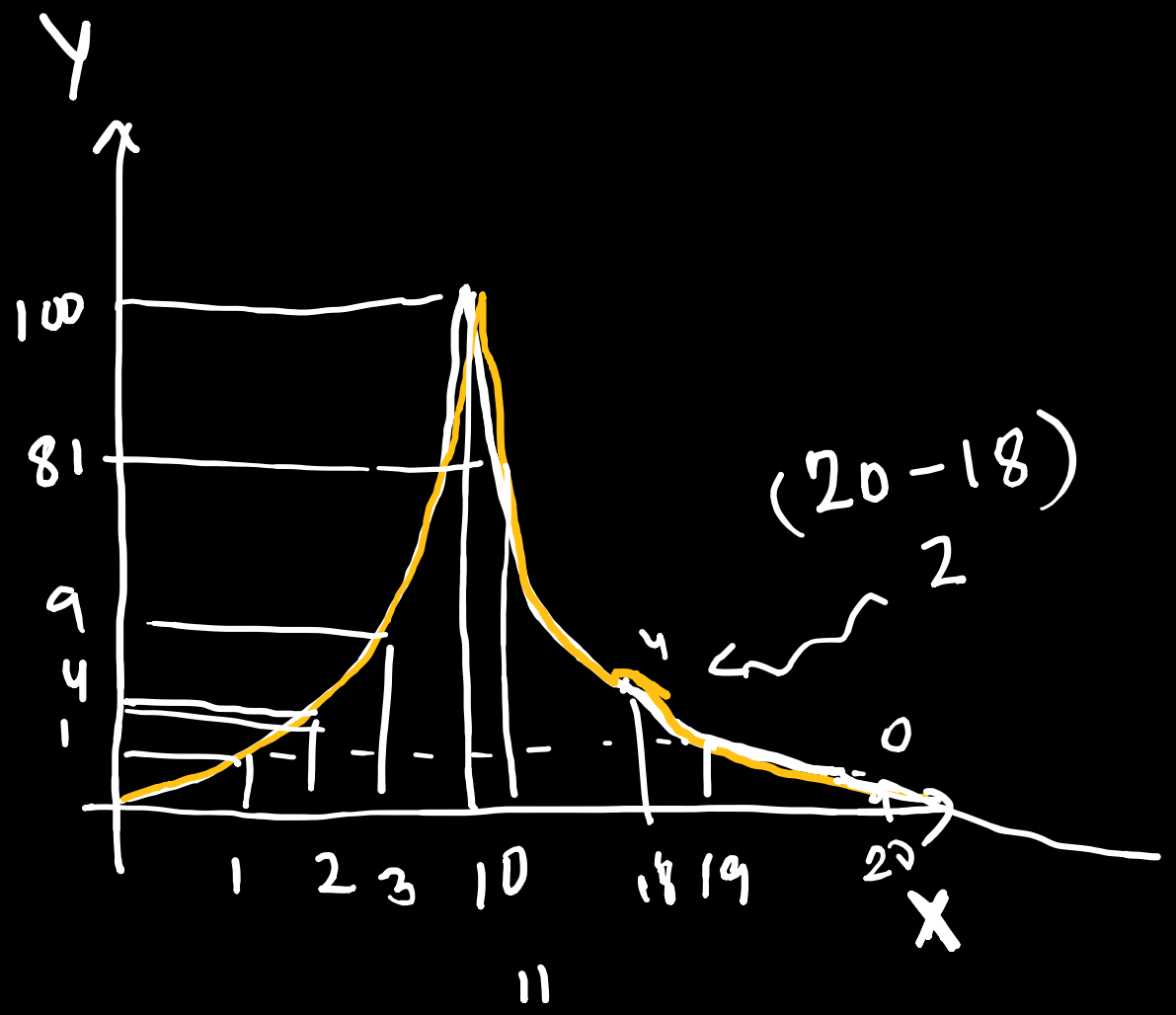
3 — j(3) j(17)

4 — j(4)

⋮ ⋮

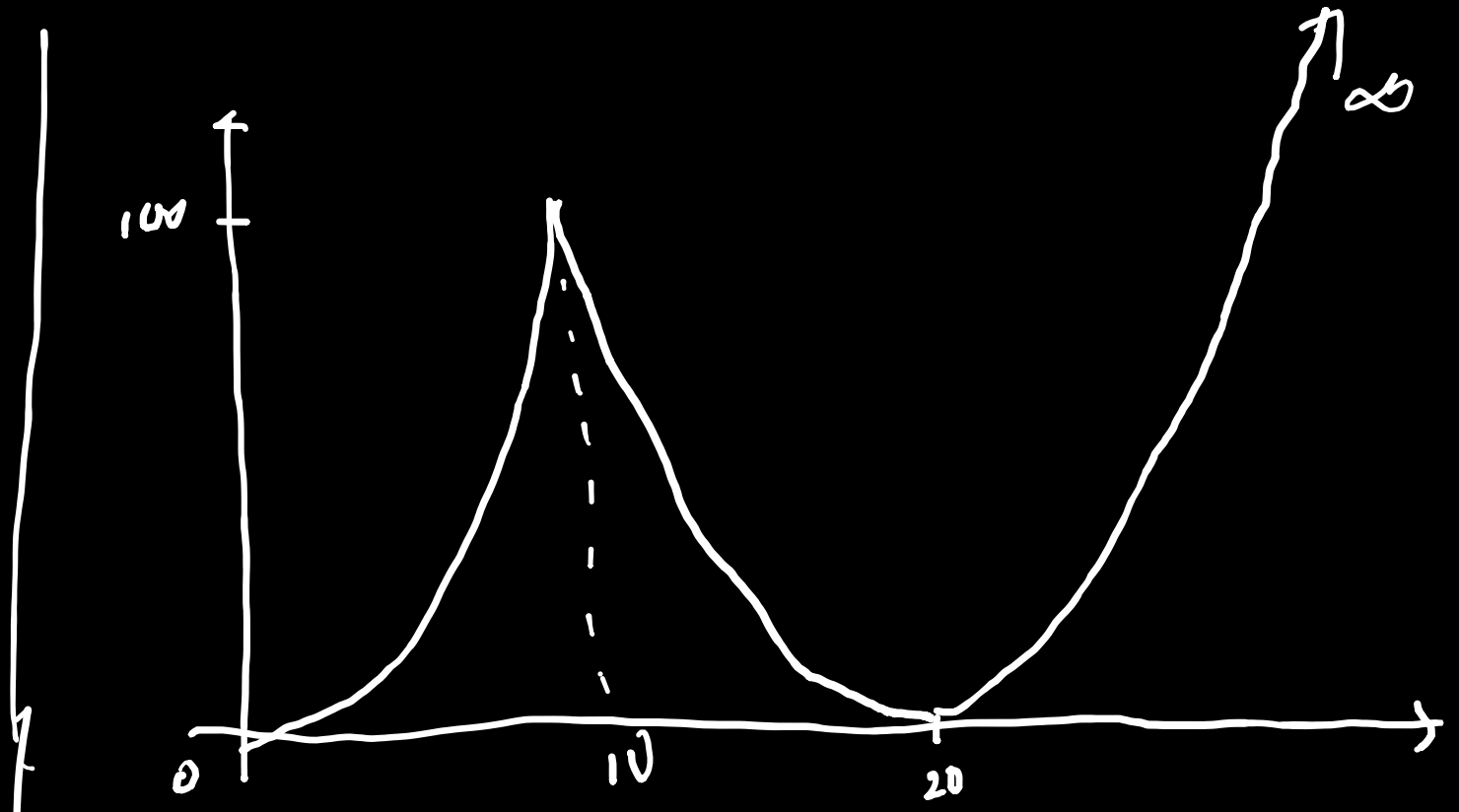
```
def j(x):
    if x <= 10:
        return x
    else:
        return (20 - x)
```

piece-wise function




```
def k(x):
    - if x<=10:
          return x*x
    - else:
          return (20-x)*(20-x)
```

```
def k(x):
    if x <= 10:
        return x * x
    else
        return (20 - x) *
               (20 - x)
```



k(21) → 1
k(22) → 4
k(23) → 9

k(30) → 100
k(31) →

Functions of one variable

More than one

```
def g(x, y):
    return x + y
```

g(20, 30) → 50

g(-10, 20) → 10

g("hello", "world") → 'helloworld'

A B C

_ _ _

3 × 2 × 1

A — B — C

AB C

AC B

Fractal

Factorial

A B C D

4

24 <~ 4 × 3 × 2 × 1

100 × 99 × 98 ..... × 1

A ..... N

n - letters

n × (n-1) × (n-2) ..... × 1

factorial (n)

↳ n × (n-1) ... × 1

$n! \rightarrow n \times (n-1) \times (n-2) \times \ldots \times 1$

Number of ways to arrange 'n' things

$n! \rightarrow n \times (n-1)!$

$4! \rightarrow 4 \times 3 \times 2 \times 1$

$15! \rightarrow 15 \times 14 \times 13 \times 12 \times 11 \times 10 \times 9 \times 8 \times 7 \times 6 \times 5 \times 4 \times 3 \times 2 \times 1$

$1 \times 1 \times 2 \times 3 \times 4 \times 5 \times 6 \times 7 \times 8 \times 9 \times 10 \times 11 \times 12 \times 13 \times 14 \times 15$

$n! \rightarrow 1 \times 2 \times 3 \times 4 \times 5 \ldots \times n$

$1! \rightarrow 1$

$0! \rightarrow 1$

```
def factorial ( n ):
    v = 1
    for i in range( 1 , n+1 ):
        v = v * i
    return v
```

$n! \rightarrow 1 \times 2 \times 3 \times 4 \times 5 \ldots \ldots \times n$

v = 1 n = 5

[1, 2, 3, 4, 5]

i = 1

v = v * i

= 1 x 1

i = 2

v = v * 2

= 1 x 1 x 2

i = 3

v = 1 x 1 x 2 x 3

i = 4

v = v x 1 x 2 x 3 x 4

i = 5

v = 1 x 1 x 2 x 3 x 4 x 5

Fibonacci Function

$F(0) = 1$

$F(1) = 1$

$F(2) = F(1) + F(0) \rightsquigarrow 2$

$F(3) = F(2) + F(1) \rightsquigarrow 3$

$F(4) = F(3) + F(2) \rightsquigarrow 5$

$F(5) = F(4) + F(3) \rightsquigarrow 8$

13

21

$\vdots$

$F(n) = F(n-1) + F(n-2)$

F(n) Calculate this F(20)

$F(5) = F(4) + F(3)$

$F(5) = F(3) + F(2) + F(2) + F(1)$

$F(5) = F(2) + F(1) + F(1) + F(0) + F(1) + F(1) + F(1)$

$F(1) + F(0) + F(1) + F(1) + F(0) + F(1) + F(1) + F(0) + F(1)$

n = 5

a = 0 b = 1 c = 0

i → [0 1 2 3 4]

---

i = 0

a = 1 b = 1 c = 1

i = 1

a = 1 b = 2 c = 2

i = 3

a = 2 b = 3 c = 3

i = 4

a = 3 b = 5 c = 5

3

0, 1, 1, 2, 3, 5 n = 6

```
def fibon(n):
    a = 0
    b = 1
    c = 0
    for i in range(n-1):
        c = a + b
        a = b
        b = c
    return c
```