AI VIETNAM
@aivietnam.edu.vn

WarmUp Class

Advanced Variables & Operators

Code - Data Quiz-Feedback

MSc. Nguyen Quoc Thai
STA. Xin Quy Hung

Objectives

Review

- Variable
- Built-in Functions
- Basic Operators

Advanced Operators

- Assignment Operator
- Relation Operator
- Logical Operator
- Precedence and Associativity

Advanced Variable

- Overflow and Underflow
- Inf and NaN
- Floating point

Exercise

- Velocity calculation
- Derivative definition
- Application
- Approximating derivative

Outline

SECTION 1

Review

SECTION 2

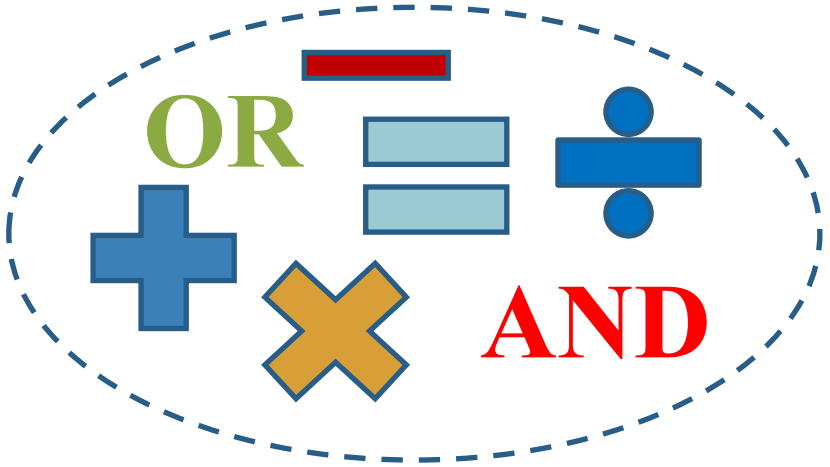
Advanced Operators

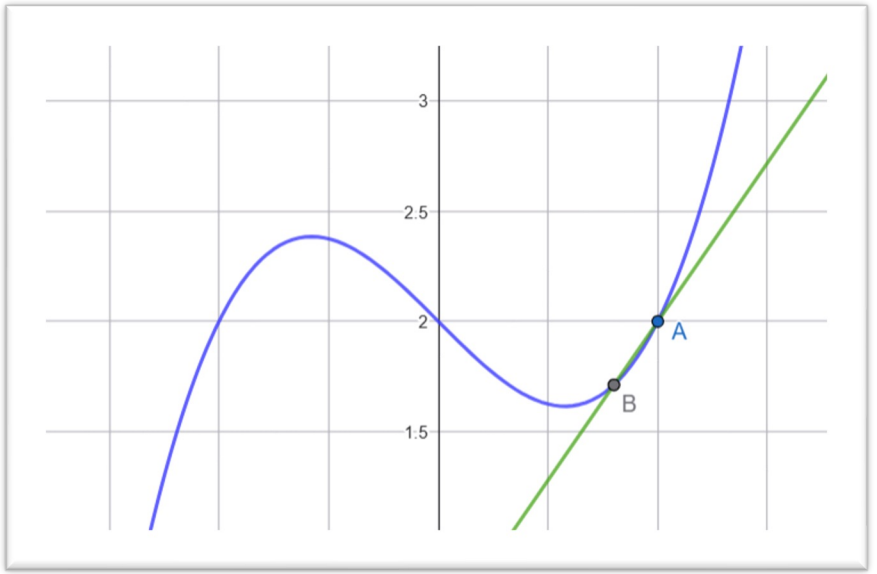
SECTION 3

Advanced Variable

SECTION 3

Exercise





Review

Basics of Python

Variable_name = Value

```
# An example of int
a = 5

# An example of float
b = 0.5

# An example of string
c = "AI Viet Nam"

# An example of bool
d = True
```

Type	Possible values
Int	-3; 0; 5; 100;…
Float	0.05; 3.5; 3.0;…
String	“AI”, “AI Viet Nam”;…
Boolean	True, False

Review

Basics of Python: Variables

Notes: Avoid variables' name that:

- Are Python keywords: print, return,…
- Start with numeric characters: 1, 2, 3…

```
return = "Hello World"
```

```
  File "/tmp/ipython-input-2170012169.py", line 1
    return = "Hello World"
           ^
SyntaxError: invalid syntax
```

```
1a = "Hello World"
```

```
  File "/tmp/ipython-input-3567537076.py", line 1
    1a = "Hello World"
    ^
SyntaxError: invalid decimal literal
```

Good practice: Declaring meaningful names when possible!

```
a = 5 # km

b = 2 # hours

c = a/b # km/h
```

```
distance = 5 # km

time = 2 # hours

velocity = distance / time # km/h
```

Review

Basics of Python: Builtin-functions

Print

```
a = 10
print(a)

b = 0.5
print(b)

c = "AI Viet Nam"
print(c)

d = True
print(d)
----
10
0.5
AI Viet Nam
True
```

Type

```
a = 10
print(type(a))

b = 0.5
print(type(b))

c = "AI Viet Nam"
print(type(c))

d = True
print(type(d))
----
<class 'int'>
<class 'float'>
<class 'str'>
<class 'bool'>
```

Review

Basics of Python: Builtin-functions

Type Conversion

```
a = 10.5
b = int(a)
print(b)

c = -1
d = bool(c)
print(d)

e = 0
f = bool(e)
print(f)
----
10
True
False
```

Why!?

Input

```
name = input("What is your name? ")
print("Hello,", name)
----
What is your name? AI Viet Nam
Hello, AI Viet Nam
```

Review

Basics of Python: Operators

Basic operators

```
A = 10
B = 4

print(A + B)
print(A - B)
print(A * B)
print(A / B)
print(A // B)
print(A % B)
----
14
6
40
2.5
2
2
```

Arithmetic Operator	Meaning
+	Addition
-	Subtraction
*	Multiplication
/	Division (Float)
//	Division (Int)
%	Modulo
**	Power

Outline

SECTION 1

Review

SECTION 2

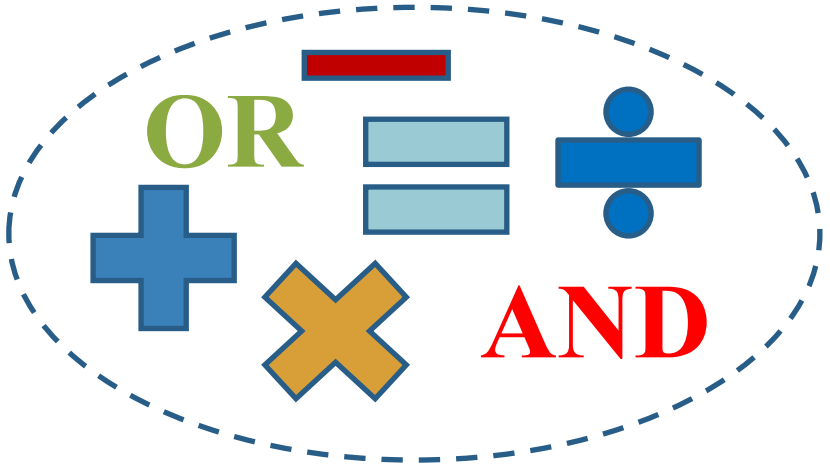
Advanced Operators

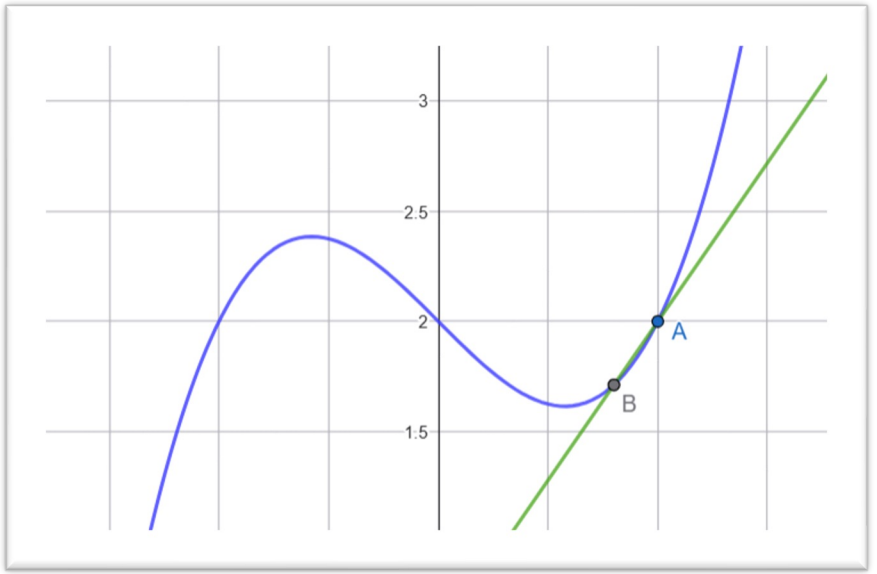
SECTION 3

Advanced Variable

SECTION 3

Exercise





Operators

Operator overview

Operator type	Examples
Arithmetic operators	**+, -, *, /, //, %, ** **
Relational operators	**<, <=, >, >=, ==, !=**
Logical operators	**AND, OR, NOT**
Bitwise operators	&, |, ^, ~, <<, >>
Assignment operators	**=, +=, -=, *=, /=, //=, %=, **=**

There are many types of operators in Python

Assignment Operator

Some assignment operators

```
a = 10
b = 20


a += b
print(a)


a -= b
print(a)


a *= b
print(a)


a /= b
print(a)
```

```
30
10
200
10.0
```

Arithmetic Operator	Equivalent Assignment Operator
+	+=
-	-=
*	*=
/	/=
//	//=
%	%=
**	**=

Assignment Operator

Why use assignment operators?

Correct but too long!

```
a_very_long_variable_name = 10
a_very_long_variable_name = a_very_long_variable_name + 10
print(a_very_long_variable_name)
----
20
```

Much faster!

```
a_very_long_variable_name = 10
a_very_long_variable_name += 10
print(a_very_long_variable_name)
----
20
```

Relational Operator

Some relational operators

```
a = 10
b = 5

print(a > b)
print(a >= b)
print(a < b)
print(a <= b)
print(a == b)
print(a != b)
----
True
True
False
False
False
True
```

Relational Operator	Meaning	Example
>	Greater	10 > 5 → True
>=	Greater or equal to	10 >= 10 → True
<	Smaller	10 < 5 → False
<=	Smaller or equal to	10 <= 5 → False
==	Equal	10 == 5 → False
!=	Different	10 != 5 → True

Relational Operator

Example: Check if two number is the same

```
a = int(input("Enter the first number: "))
b = int(input("Enter the second number: "))

print("Two number is the same: ", a = b)
```

```
Enter the first number: 10
Enter the second number: 20
```

```
TypeError                                 Traceback (most
/tmp/ipython-input-2310854366.py in <cell line: 0>()
      2 b = int(input("Enter the second number: "))
      3
----> 4 print("Two number is the same: ", a = b)

TypeError: 'a' is an invalid keyword argument for print()
```

```
a = int(input("Enter the first number: "))
b = int(input("Enter the second number: "))

print("Two number is the same: ", a == b)
```

```
Enter the first number: 10
Enter the second number: 20
Two number is the same:  False
```

What is the difference?

Common mistake in programming!

Logical Operator

Some logical operator

```
T = True
F = False

print(T and F)
print(T or F)
print(not T)

----
False
True
False
```

A	B	A and B	A or B
True	True	True	True
True	False	False	True
False	True	False	True
False	False	False	False

A	Not A
True	False
False	True

- **And**: Only True when both A and B are True
- **Or**: Only False when both A and B are False
- **Not**: Reverse True and False

Logical Operator

Example: XOR operator

Result?

```
A = True
B = True

print((A and (not B)) or ((not A) and B))
```

Fun fact: The above equation is equivalent to another operator in Python: XOR (^)

```
A = True
B = True

print(A ^ B)
----
False
```

XOR: Only True when A and B are different

A	B	A ^ B
True	True	False
True	False	True
False	True	True
False	False	False

Order of operation

How to calculate?

P	( )	Parenthases
E	A^2	Exponents
M or D	× ÷	Multiplication Division
A or S	+ −	Addition Subtraction

```
print(4 > 5 * 3)
----
19
```

Easy calculation! Multiplication first, then addition later

```
print(4 + 5 * 3 ** 2 > 5 % 2 + 1)
```

!??

Order of operation

Precedence and Associativity

```
print(4 + 5 * 3 ** 2 > 5 % 2 + 1)
# 4 + 5 * 9 > 5 % 2 + 1
# 4 + 45 > 1 + 1
# 49 > 2
---
True
```

Priority	Operator	Example
1	Power	**
2	Multiplication, Division, Modulo	*, /, //, %
3	Addition, Subtraction	+, -
4	Comparison	>, >=, <, <=
5	Equality comparison	==, !=
6	Assignment operators	=, +=, -=, *=, /=, //=, %=, **=

Outline

SECTION 1

Review

SECTION 2

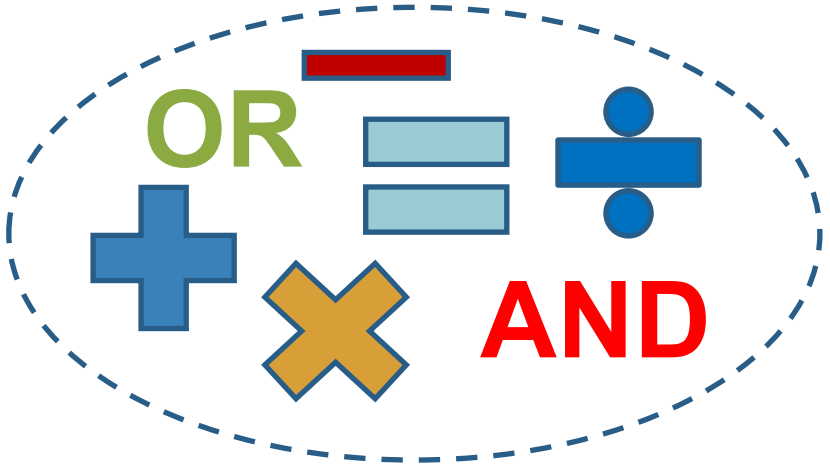
Advanced Operators

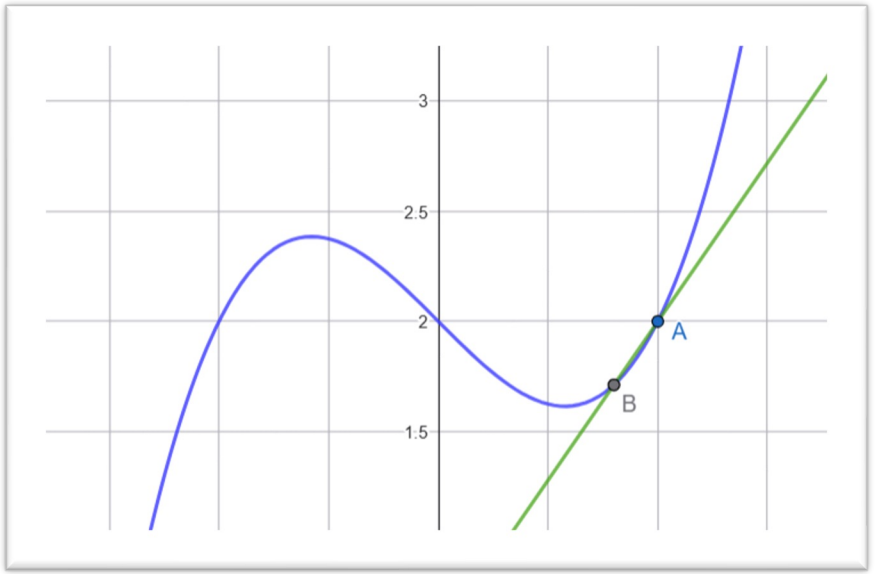
SECTION 3

Advanced Variable

SECTION 3

Exercise





Mutable and Immutable

What are mutable and immutable objects?

Mutable objects: Objects that can be changed: list, set or dict

New item is added to the list

```
a = [1, 2, 3]
a.append(4)
print(a)
----
[1, 2, 3, 4]
```

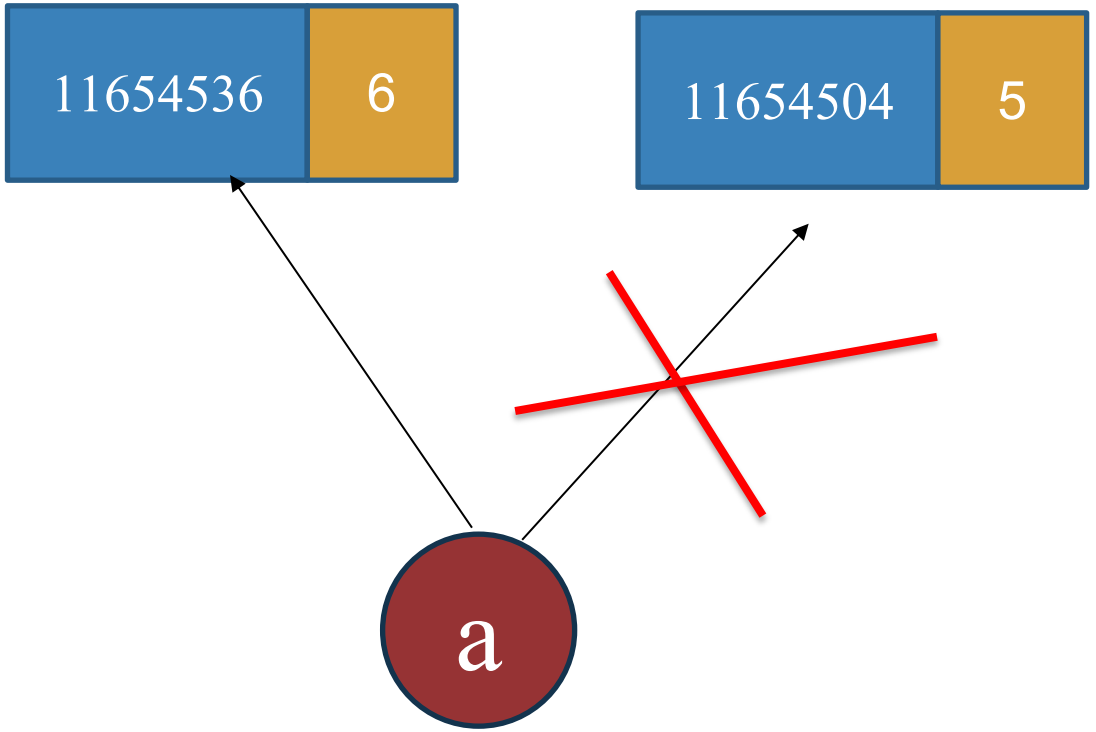
Immutable objects: Objects that cannot be changed: int, float, bool or string

Wait, how can int, float or bool be immutable?

```
a = 10

a += 10
print(a)
----
20
```

Mutable and Immutable

Immutable Object



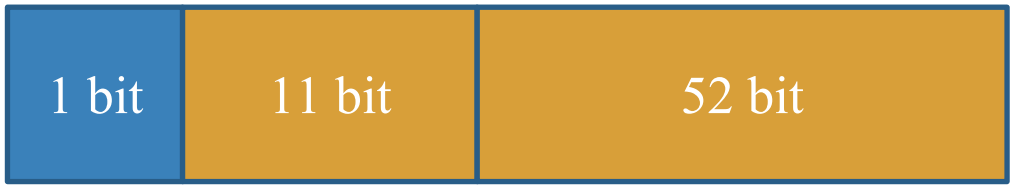

```
a = 5
print(id(a))
a = 6
print(id(a))
----
11654504
11654536
```

ID: A number that indicates the address of the object

Python does not change the value but create a new one instead

Overflow and Underflow

Underflow and Overflow in Python




```
a = 1e-400
print(a)
----
0.0
```

```
a = 1e400
print(a)
----
inf
```

Float in Python has its capacity

Underflow! The value becomes 0

Overflow! But what is that value?

Inf

Overflow in Python

```
a = 1e-400 # 10^400
print(a)
----
0.0
```

Larger than the capacity of float in Python!

```
print(1e200, "*", 1e200, "=", 1e200 * 1e200)
----
1e200 * 1e200 = inf
```

Be careful while doing calculation

Inf usually happens when overflow

Inf

Operator with inf

```
a = float('inf')
print(a)
print(type(a))
----
inf
<class 'float'>
```

```
print(float('inf')/100)
print(float('inf') * float('inf'))
print(float('inf') > 1e200)
print(float('inf') == float('inf'))
----
inf
inf
True
True
```

```
print(float('inf') * 0)
```

Inf in Python is considered as a float value

Can perform normal calculation

How about this? The result is 0 or inf?

NaN

What is NaN?

```
print(float('inf')*0)
print(float('inf') / float('inf'))
print(float('inf') - float('inf'))
print(pow(float('inf'), 0))
print(pow(1, float('inf')))
----
nan
nan
nan
1
1
```

NaN = Not a number

$$\frac{0}{0},\ \frac{\infty}{\infty},\ \infty - \infty,\ 0 \cdot \infty,\ 0^0,\ 1^\infty,\ \infty^0.$$

Usually happens when we perform "meaningless" calculation that leads to non-determinant form

But not all non-determinant form in Math will lead to NaN

NaN

NaN in raw data

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S
5	6	0	3	Moran, Mr. James	male	NaN	0	0	330877	8.4583	NaN	Q
7	8	0	3	Palsson, Master. Gosta Leonard	male	2.0	3	1	349909	21.0750	NaN	S
...	...	...	...	...	...	...	...	...	...	...	...	...
884	885	0	3	Sutehall, Mr. Henry Jr	male	25.0	0	0	SOTON/OQ 392076	7.0500	NaN	S
885	886	0	3	Rice, Mrs. William (Margaret Norton)	female	39.0	0	5	382652	29.1250	NaN	Q
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.0000	NaN	S
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.4500	NaN	S
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.7500	NaN	Q

Can also be found in raw data with missing/NaN values

Approximation in Float

How float is stored in Python

```
print(0.1 + 0.2 == 0.3)
----
False
```

How!?

```
print(0.1 + 0.2)
----
0.30000000000000004
```

Python only stored an approximation of the value, not the exact value

Outline

SECTION 1

Review

SECTION 2

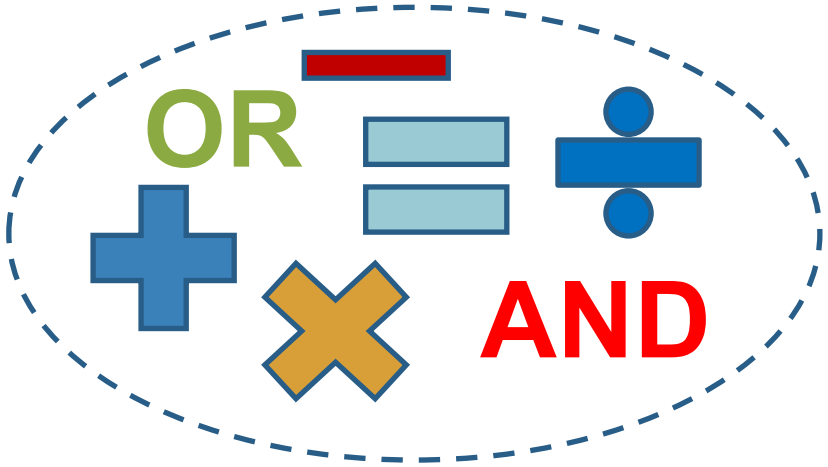
Advanced Operators

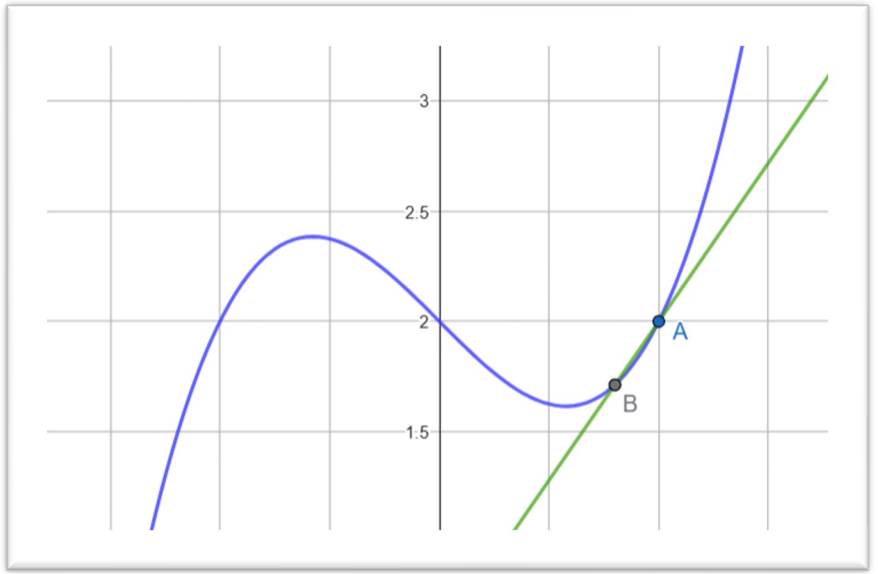
SECTION 3

Advanced Variable

SECTION 3

Exercise

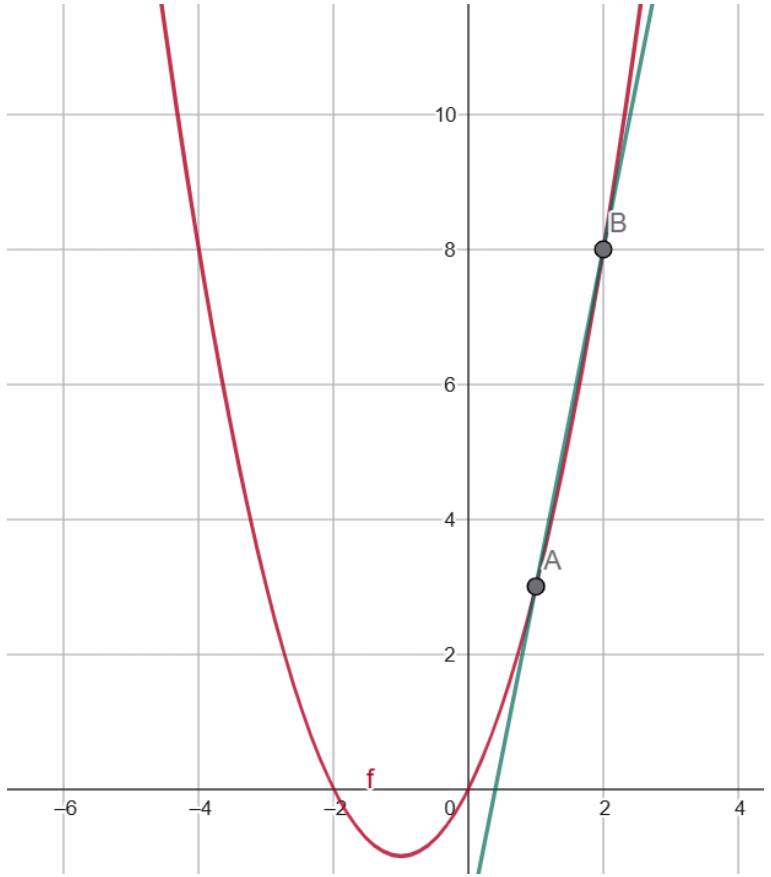




Velocity calculation

Problem #1

Calculating the average velocity from t =1 to t = 2



Problem: A person travels from A to B with the following formula:

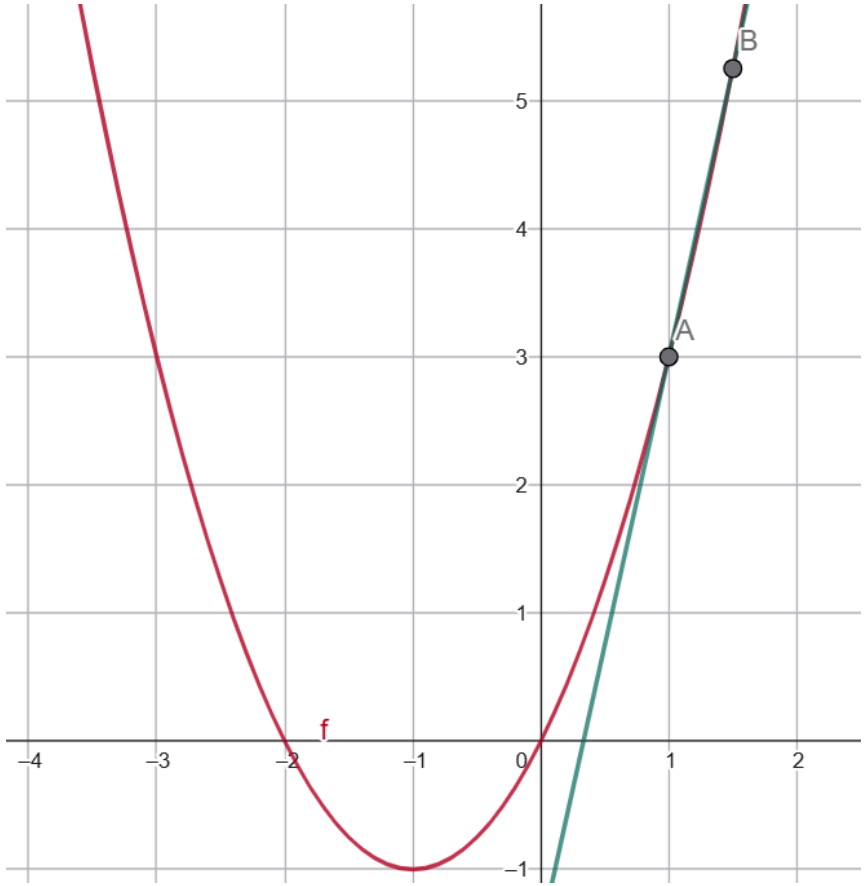
$$s(t) = t^2 + 2t$$

```
t1 = 1
t2 = 2

s1 = t1*t1 + 2*t1
s2 = t2*t2 + 2*t2
velocity = (s2 - s1)/(t2 - t1)
print(velocity)
----
5.0
```

Velocity calculation

Problem #2

Calculating the average velocity from t =1 to t = 1.5



Problem: A person travels from A to B with the following formula:

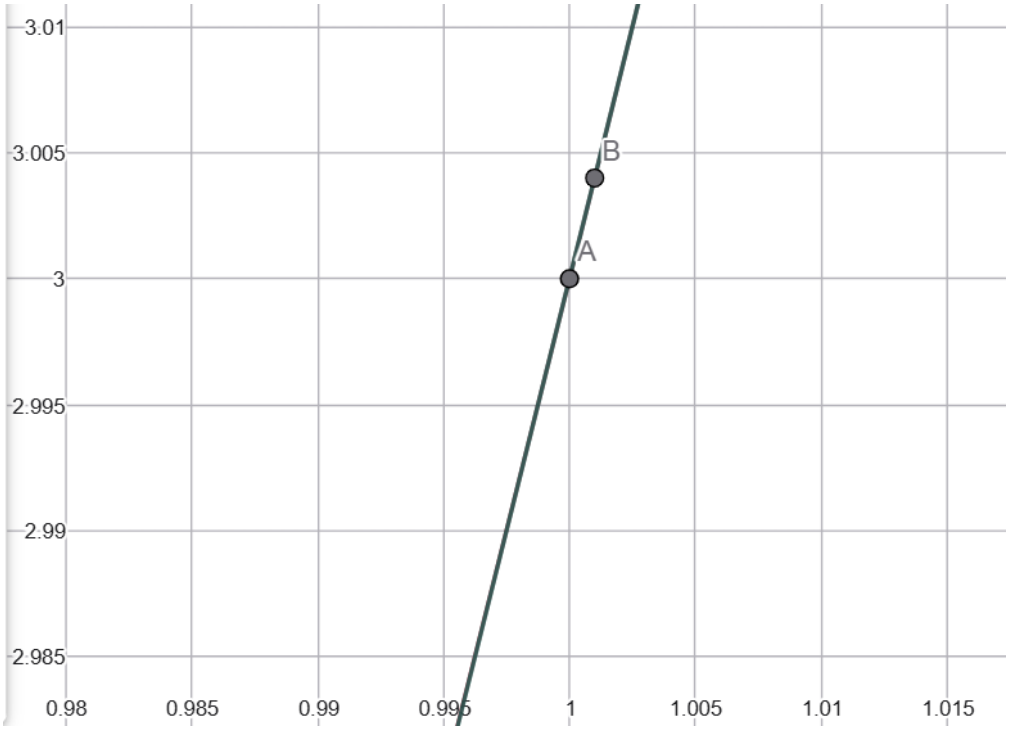
$$s(t) = t^2 + 2t$$

```
t1 = 1
t2 = 1.5

s1 = t1*t1 + 2*t1
s2 = t2*t2 + 2*t2
velocity = (s2 - s1)/(t2 - t1)
print(velocity)
----
4.5
```

Velocity calculation

Problem #3

Calculating the average velocity from t =1 to t = 1.001



Problem: A person travels from A to B with the following formula:

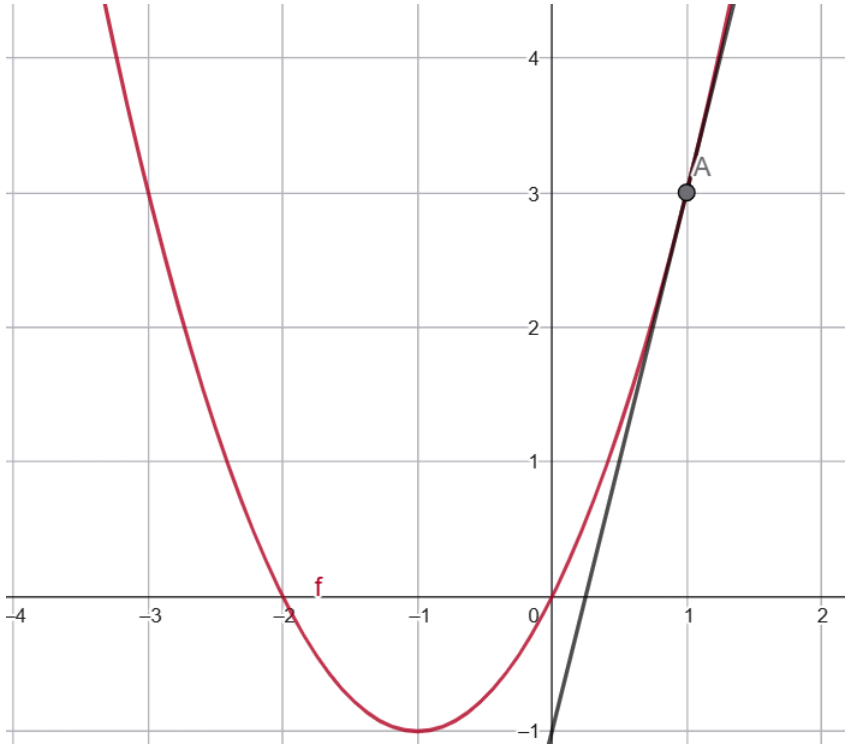
$$s(t) = t^2 + 2t$$

```
t1 = 1
t2 = 1.001

s1 = t1*t1 + 2*t1
s2 = t2*t2 + 2*t2
velocity = (s2 - s1)/(t2 - t1)
print(velocity)
----
4.00100000000014
```

Velocity calculation

Problem #4

Calculating the instantaneous velocity at t = 1?



Problem: A person travels from A to B with the following formula:

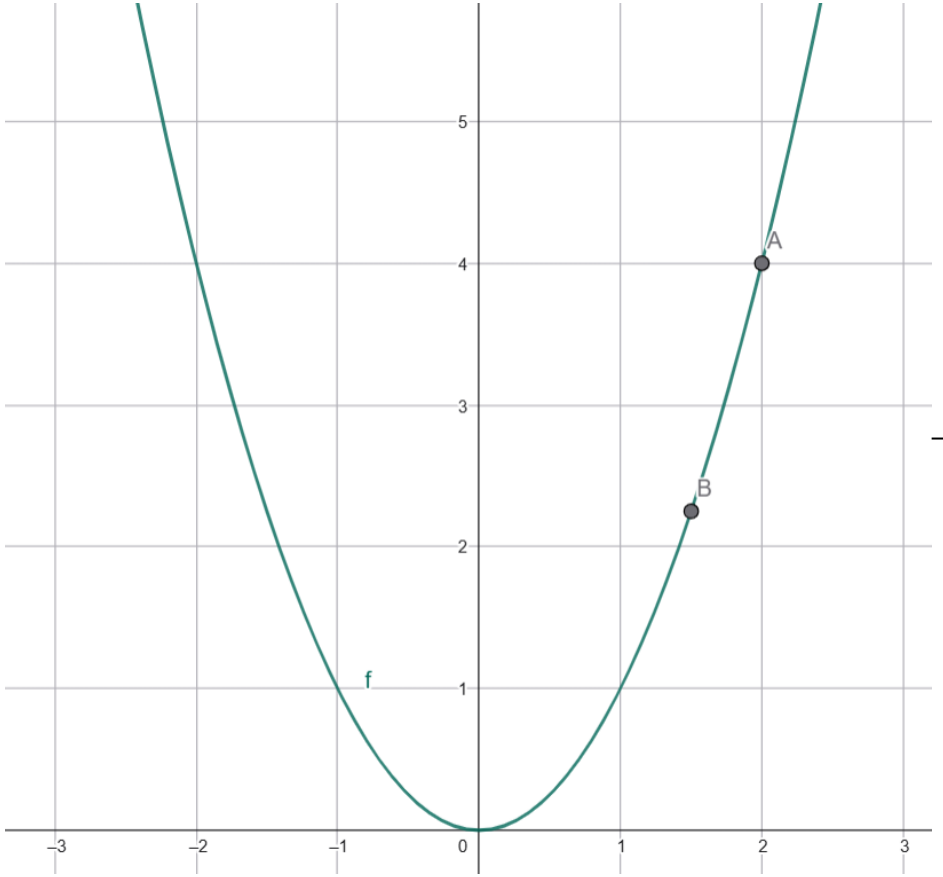
$$s(t) = t^2 + 2t$$

```
t1 = 1
t2 = 1

s1 = t1*t1 + 2*t1
s2 = t2*t2 + 2*t2
velocity = (s2 - s1)/(t2 - t1)
print(velocity)
```

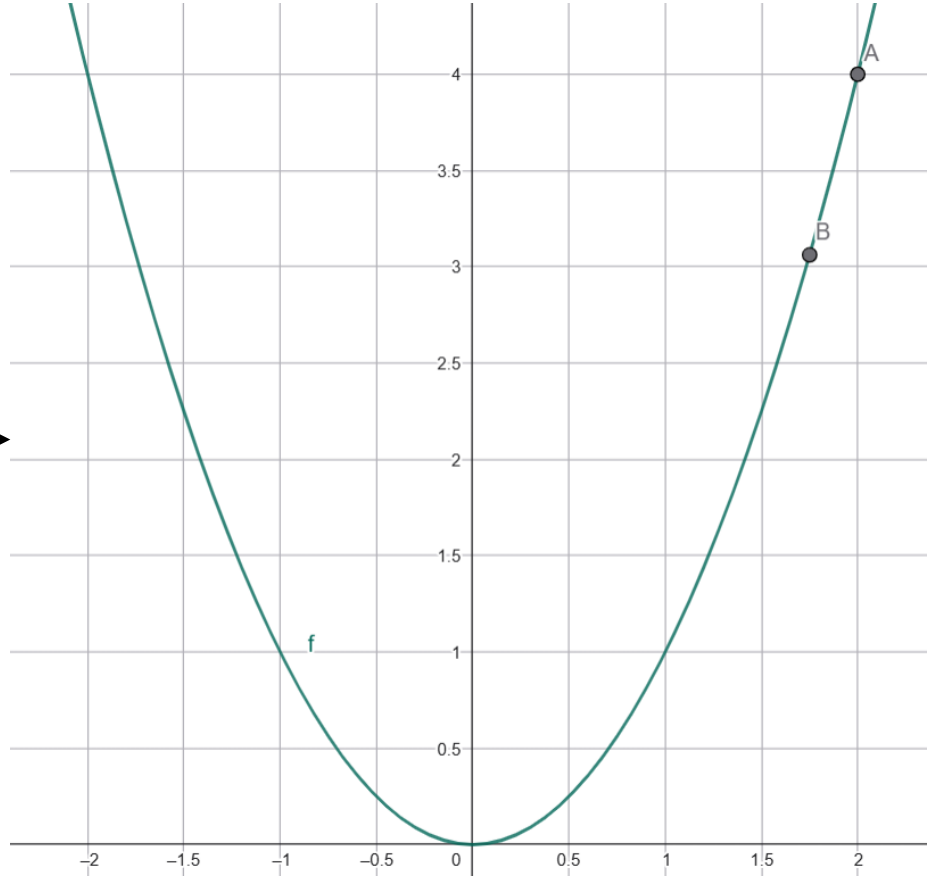
```
---------------------------------------------------------------
ZeroDivisionError                         Traceback
/tmp/ipython-input-1802911845.py in <cell line: 0>()
      4 s1 = t1*t1 + 2*t1
      5 s2 = t2*t2 + 2*t2
----> 6 velocity = (s2 - s1)/(t2 - t1)
      7 print(velocity)

ZeroDivisionError: division by zero
```

Derivative

Derivative illustration



Closer

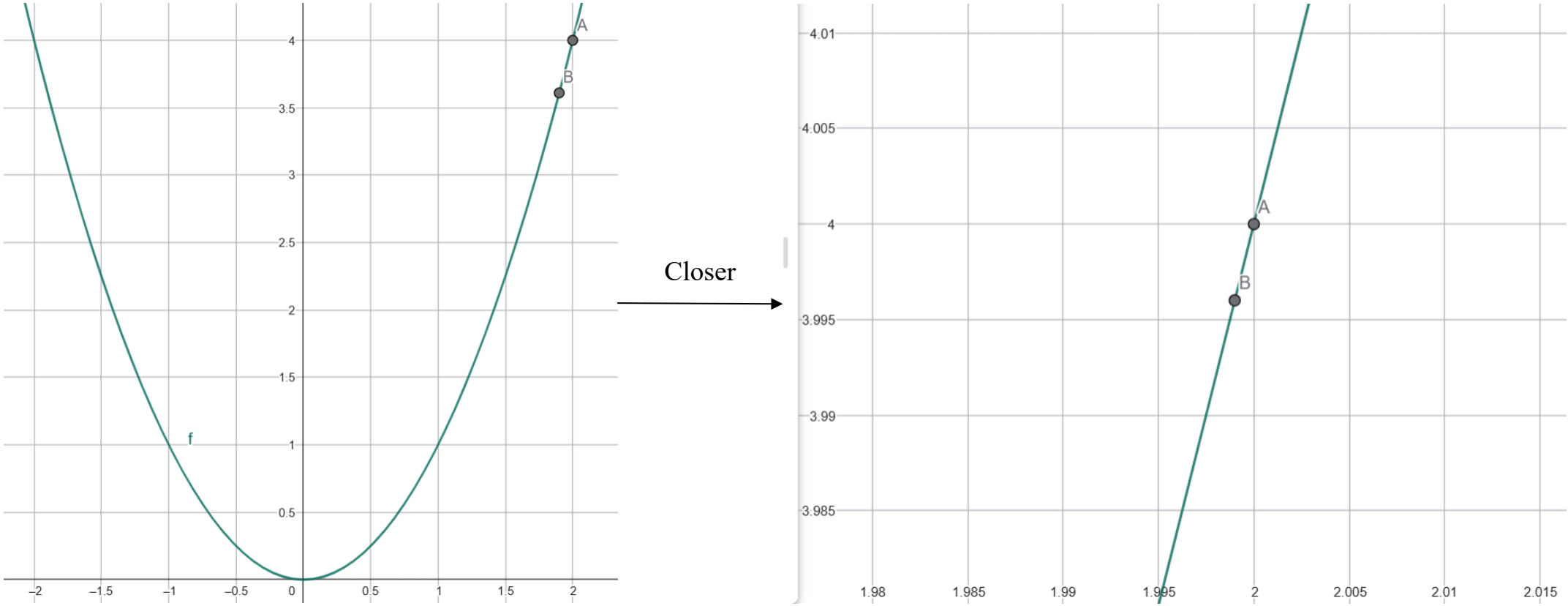


Let B approach A

Let B approach closer

Derivative

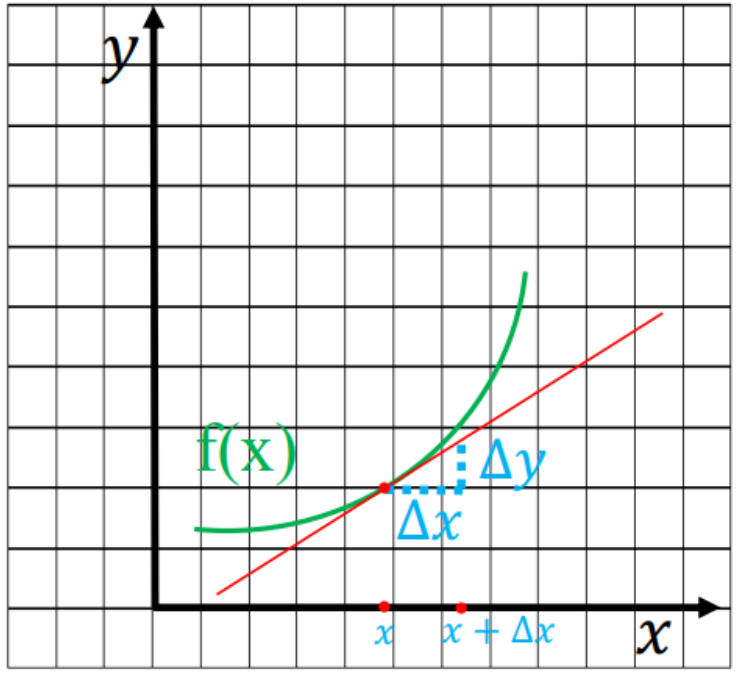
Derivative illustration



Can we make B even closer?

Derivative

Definition



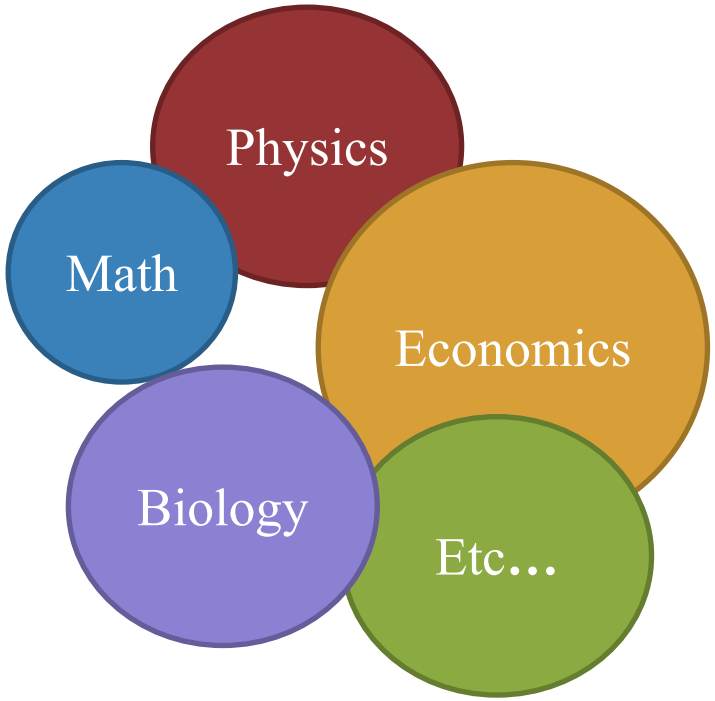
$$\frac{d}{dx}f(x) = \lim_{\Delta x \to 0} \frac{f(x + \Delta x) - f(x)}{\Delta x}$$

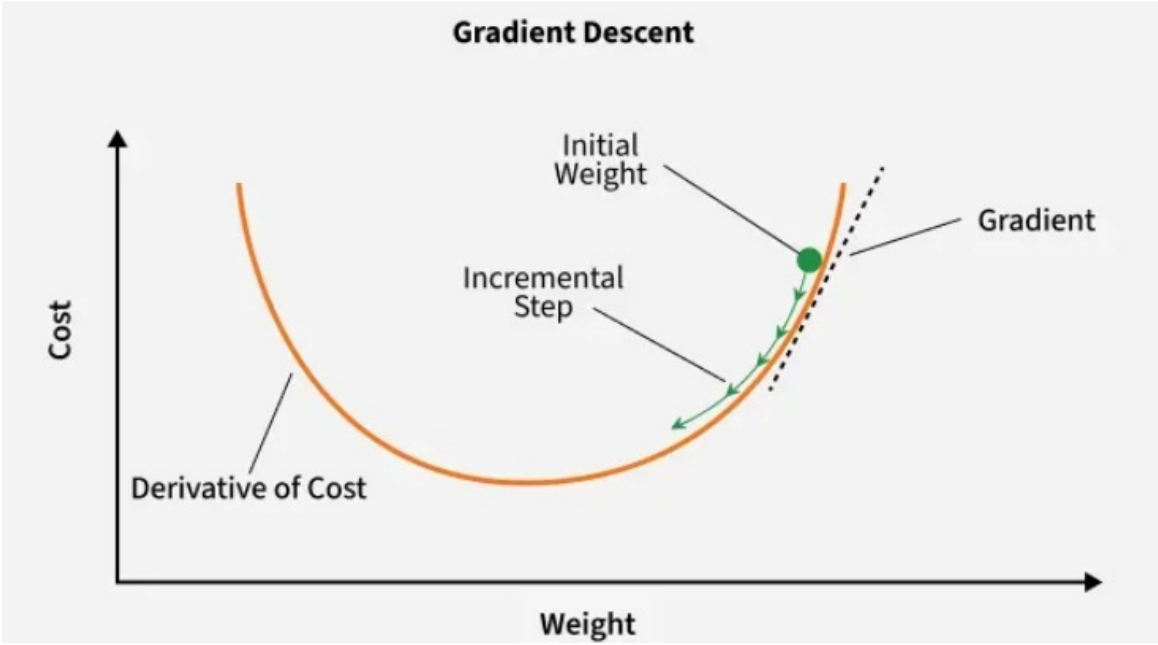
Make Δx approaches 0.
Notes: Approaches 0, not equal to 0

$$\frac{d}{dx}f(x), \frac{dy}{dx}, y', f'(x)$$

Derivative

Applications





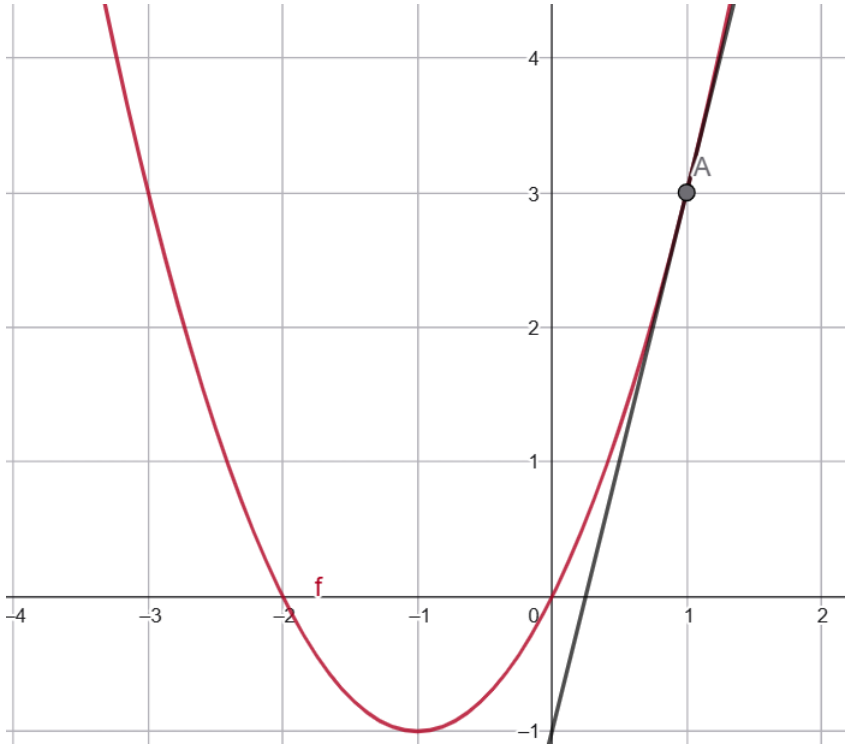
Have applications in many fields

In AI: Gradient descent,…

Velocity calculation

Problem #4

Calculating the instantaneous velocity at t = 1?



Problem: A person travels from A to B with the following formula:

$$s(t) = t^2 + 2t \longrightarrow s'(t) = 2t + 2$$

Approximation

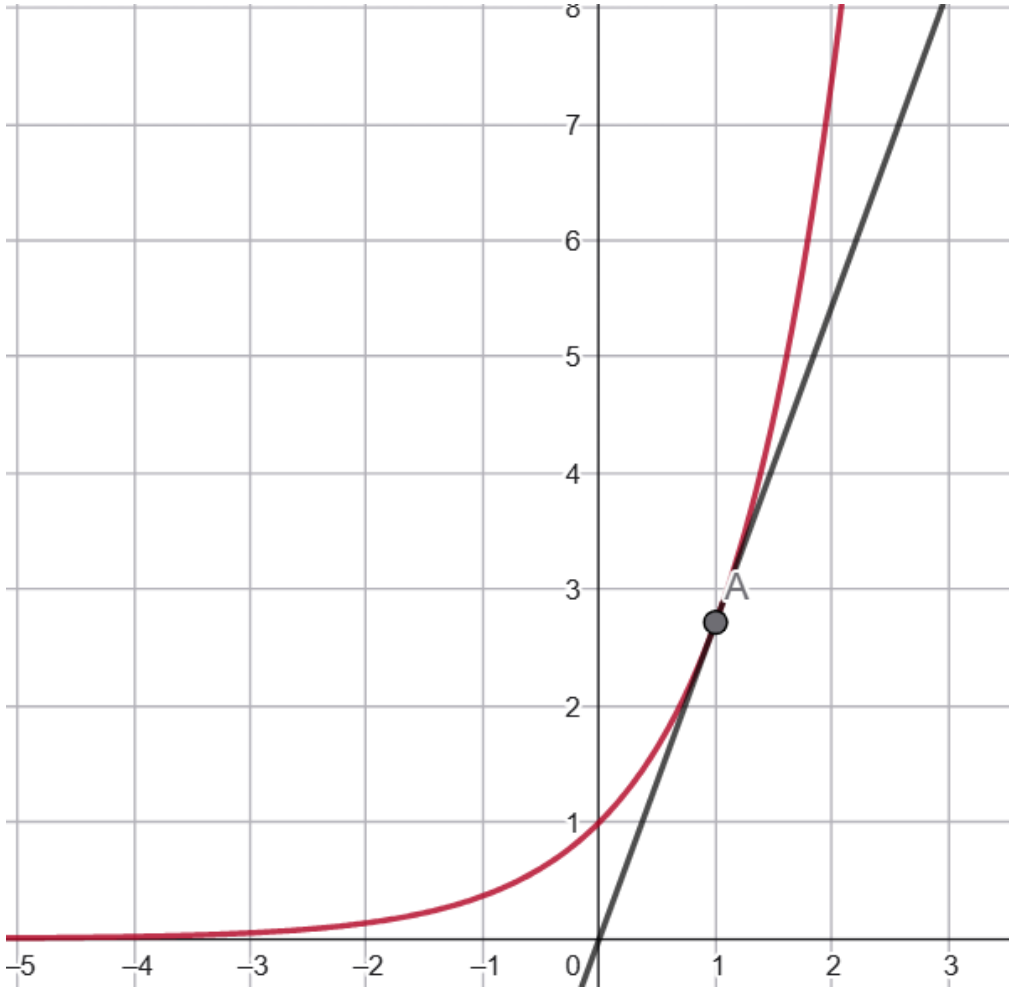
```
t1 = 1
t2 = 1.001

s1 = t1*t1 + 2*t1
s2 = t2*t2 + 2*t2
velocity = (s2 - s1)/(t2 - t1)
print(velocity)
----
4.00100000000014
```

Correct result

```
derivative = 2*t1 + 2
print(derivative)
----
4
```

Derivative

Exercise: Approximating derivative



$$\frac{d}{dx}e^x = e^x$$

```
print(math.exp(x))
```

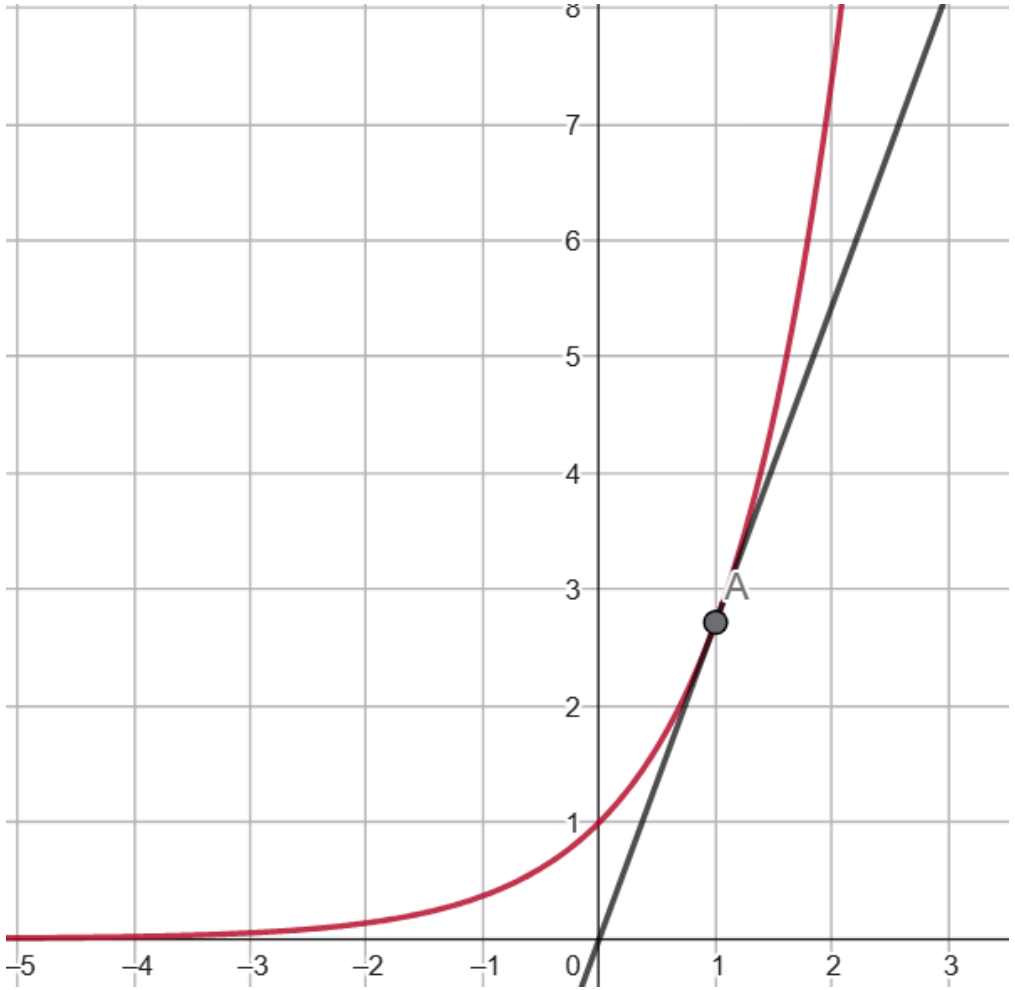
```
2.718281828459045
```

```
x = 1
h = 0.5

f_x = math.exp(x)
f_x_h = math.exp(x + h)

derivative = (f_x_h - f_x) / h
print(derivative)
----
3.526814483758039
```

```
x = 1
h = 0.1

f_x = math.exp(x)
f_x_h = math.exp(x + h)

derivative = (f_x_h - f_x) / h
print(derivative)
----
2.858841954873883
```

Derivative

Exercise: Approximating derivative



$$\frac{d}{dx}e^x = e^x$$

```
print(math.exp(x))
```

```
2.718281828459045
```

```
x = 1
h = 0.01

f_x = math.exp(x)
f_x_h = math.exp(x + h)

derivative = (f_x_h - f_x) / h
print(derivative)
----
2.7319186557871245
```

```
x = 1
h = 1e-20

f_x = math.exp(x)
f_x_h = math.exp(x + h)

derivative = (f_x_h - f_x) / h
print(derivative)
----
0.0
```

Why!?


QUIZ TIME

Objectives

Review

- Variable
- Built-in Functions
- Basic Operators

Advanced Operators

- Assignment Operator
- Relation Operator
- Logical Operator
- Precedence and Associativity

Advanced Variable

- Overflow and Underflow
- Inf and NaN
- Floating point

Exercise

- Velocity calculation
- Derivative definition
- Application
- Approximating derivative

Thanks!

Any questions?