[**https://python.maateen.me/docs/class-object-method/**](https://python.maateen.me/docs/class-object-method/)

**ক্লাস, অবজেক্ট এবং মেথড**

### সেলফ (self)

আমি নিশ্চিত আমাদের অনেকেরই self নিয়ে ব্যাপক দুশ্চিন্তা হচ্ছে। এটা নিয়ে মাথা ঘামানোর আসলে তেমন কিছু নাই। কারণ:

* self আসলে একটা কনভেনশনাল নাম। আমরা চাইলে এটাকে বদলে দিতে পারি।
* self হল ক্লাস ইন্সট্যান্সের একটা রেফারেন্স। ইন্সট্যান্স ভ্যারিয়েবল ডিক্লেয়ার করার সময় self এর রেফারেন্স ব্যবহার করা হয়। ক্লাস ভ্যারিয়েবল আর ইন্সট্যান্স ভ্যারিয়েবলের ভিতর পার্থক্য হল ক্লাস ভ্যারিয়েবল ক্লাসের সব ইন্সট্যান্সের ভিতরেই শেয়ার্ড, অন্যদিকে ইন্সট্যান্স ভ্যারিয়েবল প্রতিটা ইন্সট্যান্সে ইউনিক।

## অবজেক্ট (object) কী?

অবজেক্টের শাব্দিক অর্থ বস্তু। নামের মধ্যেই এর কাজের কথা বলা আছে। অবজেক্ট আসলে বস্তুবাদী মানে পার্থিব চিন্তার বিষয়। (পৃথিবীতে আমাদের সামনে দৃশ্যমান বস্তুই পার্থিব।)

আমাদের ঐ যে ‘ওয়ালটন উস্তা’ ক্লাসটা, এটা দিয়ে আমরা কিন্তু লাখ লাখ গাড়ি তৈরি করতে পারি। প্রতিটা গাড়িতেই কিন্তু ঐ চারটা ফাংশন থাকবে। এখন আমাদের গাড়িটার কথা ধরি। আমাদের গাড়িটাও তো ‘ওয়ালটন উস্তা’ ক্লাস থেকে তৈরি। তাহলে আমাদের গাড়িটা হল ঐ ক্লাসের একটা অবজেক্ট। ঐ ক্লাস থেকে তৈরি সব গাড়িই এক-একটা অবজেক্ট।

চলুন সবাই মিলে পাইথনিক উপায়ে ‘ওয়ালটন উস্তা'র অবজেক্ট তৈরি করি।

our\_car = WaltonUsta()

her\_car = WaltonUsta()

your\_car = WaltonUsta()

এখানে our\_car, her\_car, your\_car সবগুলো হল অবজেক্ট, WaltonUsta() ক্লাসের অবজেক্ট। আর প্রতিটা অবজেক্টই কিন্তু ক্লাসের সব ফাংশনকে অ্যাক্সেস করতে পারবে।

## মেথড (Method) কী?

মেথড নতুন কিছু না। এতক্ষণ আমরা যাকে ফাংশন নামে চিনেছি, সে যখন ক্লাসের ভিতর থাকে তখন তাকে আমরা মেথড বলব। শুধু নামকরণে তফাৎ। বাদবাকি সবকিছুই এক। অর্থাৎ, ক্লাসের ফাংশনকে মেথড বলে। উঁহু, আরেকটা তফাৎ আছে। সবসময় মেথডের প্রথম আর্গুমেন্ট বা প্যারামিটার হবে self। আর মেথড কল করার সময় এটা পাস করার দরকার নাই। পাইথন নিজ থেকেই এটা পাস করে দেবে।

একটা কথা হয়নি বলা। ক্লাসের নাম আমরা সবসময় বড় হাতের অক্ষর দিয়ে শুরু করব। অবজেক্টের নাম ভ্যারিয়েবলের মত করেই হবে আর মেথড তো ফাংশনের মত করেই। আপাতত এটুকু জানলেই চলবে আমাদের। নামকরণের নানা নিয়ম-কানুন আমরা কনভেনশন চাপ্টারে বিস্তারিত শিখব।

এখন আমরা বাস্তব একটা উদাহরণ দেখব। আমরা ক্যালকুলেটরের জন্য একটা ক্লাস তৈরি করব। এই ক্লাসে যোগ, বিয়োগ, গুণ ও ভাগের জন্য মেথড থাকবে।

class Calculator:

"""Do addition, subtraction, multiplication and division."""

def addition(self, a, b):

return a+b

def subtraction(self, a, b):

return a-b

def multiplication(self, a, b):

return a\*b

def division(self, a, b):

try:

return a/b

except ZeroDivisionError:

return 'It is impossible to divide by zero.'

my\_calculator = Calculator()

temp = my\_calculator.addition(12, 78)

print(temp)

temp = my\_calculator.subtraction(50, 23)

print(temp)

temp = my\_calculator.multiplication(9, 19)

print(temp)

temp = my\_calculator.division(400, 5)

print(temp)

temp = my\_calculator.division(43, 0)

print(temp)

**আউটপুট**

90

27

171

80.0

It is impossible to divide by zero.

এখানে আমরা কিভাবে আমাদের মেথডগুলো অ্যাক্সেস করেছি খেয়াল করেছি তো সবাই? অবজেক্টের পর . চিহ্ন দিয়ে মেথড অ্যাক্সেস করতে হয়। ইতিপূর্বে আমরা এই কাজ লিস্ট, টাপল, ডিকশনারি এসব জায়গায় করে এসেছি। আসলে এগুলো হল বিল্ট-ইন ক্লাস। পাইথনে সবকিছুই অবজেক্ট। আমরা তখন যাদের ফাংশন বলেছি আসলে তারা ছিল মেথড।

যদি আমরা ক্লাসের ভিতর কোন ভ্যালু পাস করতে চাই? আর পরে যদি সেই ভ্যালু নিয়েই মেথডের ভিতর কোডিং কারিশমা ফলাতে চাই? তখন ক্লাসের প্রথম মেথড হিসাবে \_\_init\_\_ ব্যবহার করতে হবে। প্রোগ্রামিংয়ের ভাষায় একে ক্লাস কনস্ট্রাক্টর (constructor) বলে। যখন কোন ক্লাসের নতুন ইন্সট্যান্স বা অবজেক্ট তৈরি করি আমরা তখন পাইথন নিজ থেকেই ক্লাস কনস্ট্রাক্টরকে কল করে। ওহ! যখন ক্লাস কনট্রাক্টর আমরা ডিফাইন করি না তখন পাইথন নিজ থেকেই ক্লাস কনস্ট্রাক্টর ডিফাইন করে নেয়।

চটপট একটা উদাহরণ দেখা যাক।

class Calculator:

"""Do addition, subtraction, multiplication and division."""

def \_\_init\_\_(self, a, b):

self.a = a

self.b = b

def addition(self):

return self.a + self.b

def subtraction(self):

return self.a - self.b

def multiplication(self):

return self.a \* self.b

def division(self):

try:

return self.a / self.b

except ZeroDivisionError:

return 'It is impossible to divide by zero.'

my\_calculator = Calculator(45, 3)

temp = my\_calculator.addition()

print(temp)

temp = my\_calculator.subtraction()

print(temp)

temp = my\_calculator.multiplication()

print(temp)

temp = my\_calculator.division()

print(temp)

**আউটপুট**

48

42

135

15.0

আমরা \_\_init\_\_ ব্লকটা রিভিউ করি একটু। বাধ্যতামূলক self বাদে আমরা আরো দুইটা প্যারামিটার a, b দিয়েছি। তারপর self.a তে a এর ভ্যালু ও self.b তে b এর ভ্যালু অ্যাসাইন করেছি। এর ফলে a, b এর ভ্যালু self.a, self.b এর মাধ্যমে পুরো ক্লাসের ভিতর অ্যাক্সেসিবল হয়ে গেল।

এবার ইনস্ট্যান্স বা অবজেক্ট ক্রিয়েশনটা রিভিউ করি। my\_calculator অবজেক্ট তৈরির সময় আমরা Calculator এর ভিতর দুইটা ভ্যালু পাস করেছি। এটার সাথে \_\_init\_\_ মেথডের যোগসাজশ আছে। এই ভ্যালু দুইটা সরাসরি \_\_init\_\_ এর ভিতর চলে আসে। সহজ কথায়, \_\_init\_\_ বা ক্লাস কনস্ট্রাক্টরের প্যারামিটারই ক্লাসের প্যারামিটার।