

Forum:[Hack Forums](#)

Thread: [Can't seem to grasp the concept of Classes and Objects.](#)

pkdarn - December 3, 2019, 7:09 pm

I've tried looking around online but I can't find a source that seems to explain it well enough for me - so maybe one of y'all could help out? I get that classes are like templates for creating an object, but whats the difference between an object and a variable? Are they the same, and if so why not use one word or the other?

Invisible - December 3, 2019, 7:25 pm

Think of variables as containers. Within those containers you are able to store data values.

Code x = 22

print(x)

Python is an OOP (Object Oriented Programming) language so pretty much everything is an object. A class is like the main ingredient for finishing a recipe, and in this case a class creates an object.

Code class TestClass:

x = 22

D1 = TestClass()

print(D1.x)

pkdarn - December 3, 2019, 8:42 pm

(Dec 3, 2019 08:25 PM)Invisible Wrote: [\[CITE\]](#) Think of variables as containers. Within those containers you are able to store data values.

Code x = 22

print(x)

Python is an OOP (Object Oriented Programming) language so pretty much everything is an object. A class is like the main ingredient for finishing a recipe, and in this case a class creates an object.

Code class TestClass:

x = 22

D1 = TestClass()

print(D1.x)

I think I'm starting to grasp it. In what real world situation would you need to use this? What's the point of putting x in TestClass if you could just print x?

gigajew - December 3, 2019, 9:08 pm

A class is like a description of an object type. If I wanted my app to understand what an apple is I can first create an interface named Fruit and then inherit from that and create a class called Apple. My fruit interface may describe a string property that I can set to describe the flavor.

Here's an example in C#:

And another in Python (not my strong language)

grady - December 3, 2019, 9:12 pm

(Dec 3, 2019 10:08 PM)gigajew Wrote: [\[CITE\]](#) A class is like a description of an object type. If I wanted my app to understand what an apple is I can first create an interface named Fruit and then inherit from that and create a class called Apple. My fruit interface may describe a string property that I can set to describe the flavor. Example:

This is the Python SF

try not to be confusing with other language's syntax

gigajew - December 3, 2019, 9:19 pm

(Dec 3, 2019 10:12 PM)?Infamy Wrote: [\[CITE\]](#) This is the Python SF

try not to be confusing with other language's syntax

Sorry just realized added a python example, perhaps some insight into other languages could strengthen his concept of a class some

realsunjester - December 3, 2019, 9:27 pm

(<https://docs.python.org/3/tutorial/classes.html>)

alright man, classes are easy. classes are "objects". you can make a class out of any real world object. for example:

Apple (thats a thing/object). Apples have color right? so...

Code >>> class Apple:

... def __init__(self):

... self.color = 'red'

...

>>> a = Apple()

>>> print a.color

red

so another example, vehicles.

Code >>> class Vehicle:

... def __init__(self):

... self.doors = 2

... self.color = 'red'

... self.milage = '32,000'

...

>>> car = Vehicle()

>>> print car.doors

2

gnome sayin?

basically people create classes so you dont have to type code over and over again, its "reusability" and also makes it possible to edit code without having to find every instance of something, you could add or modify the class.

(<https://docs.python.org/3/tutorial/classes.html>)

alright man, classes are easy. classes are "objects". you can make a class out of any real world object. for example:

Apple (thats a thing/object). Apples have color right? so...

```
Code >>> class Apple:
...     def __init__(self):
...         self.color = 'red'
...
>>> a = Apple()
>>> print a.color
red
so another example, vehicles.
```

```
Code >>> class Vehicle:
...     def __init__(self):
...         self.doors = 2
...         self.color = 'red'
...         self.milage = '32,000'
...
>>> car = Vehicle()
>>> print car.doors
2
gnome sayin?
```

grady - December 3, 2019, 9:32 pm

(Dec 3, 2019 10:19 PM)gigajew Wrote: [\[CITE\]](#) Sorry just realized added a python example, perhaps some insight into other languages could strengthen his concept of a class some ye np fam, I almost made the same mistake myself so i feel u

also @ OP

It makes more sense when you have data that makes sense to be encapsulated in a class it's very possible to create pure function to manage the data in a program

let's go with a chess board for example

You can see how much cleaner the code is when implementing objects instead of just functions

koshka1 - December 3, 2019, 9:53 pm

Here are my two cents:

(Dec 3, 2019 09:42 PM)pkdarn Wrote: [\[CITE\]](#) I think I'm starting to grasp it. In what real world situation would you need to use this? What's the point of putting x in TestClass if you could just print x?

You're right there's no point in putting x in TestClass, however in the real world when you are making applications that are much bigger you will have to worry about memory management. The stack and the heap the stack is automatically allocated by the compiler, for example if you wrote `int x = 1;` it would be stored on the stack and when the program is done it would be automatically deleted.

On the flip side the heap is explicitly controlled by the programmer, so if I created an object: `Classname c = new Classname();` the object © is on the heap, and it is waiting to be destroyed. In C++ you just say `delete c` and you're all good.

I probably didn't do a great job of answering your question but I recommend you looking into data structures like linked lists, stacks, queues etc it might make more sense when you see the organization of data along with OOP

I wanted to add an example:

Imagine you are the manager of a store and you want to manage items in your store.

If you were to approach this without object-oriented programming you'd quickly realize the program itself will become extremely bulky.

PHP Code

```
<?php
class Food {
public:
enum FOOD_TYPE{
Sweet,
Starch,
Meat,
Vegetable,
Fruit
};

Food(std::string str, FOOD_TYPE type, double cost);

FOOD_TYPE getType() const { return m_eType; }

std::string getName() const { return m_sName; }

double getCost() const { return dCost; }

friend std::ostream &operator << (std::ostream &os, const Food &f);

private:
std::string m_sName;

FOOD_TYPE m_eType;

double dCost;

};

class Sweet : public Food {
public:
Sweet();

Sweet(std::string sName, double cost);

~Sweet();
};
```

```

class Starch : public Food {
public:
    Starch();

    Starch(std::string sName, double cost);

    ~Starch();
};

class Meat : public Food {
public:
    Meat();

    Meat(std::string sName, double cost);

    ~Meat();
};

class Vegetable : public Food {
public:
    Vegetable();

    Vegetable(std::string sName, double cost);

    ~Vegetable();
};

class Fruit : public Food {
public:
    Fruit();

    Fruit(std::string sName, double cost);

    ~Fruit();
};

#endif

```

This simple example demonstrates inheritance, you have a parent class FOOD and children classes sweet, meat, starch etc.

I can now create different food objects and do what I want with them, like stock the store.

Quote: basically people create classes so you don't have to type code over and over again, its "reusability" and also makes it possible to edit code without having to find every instance of something, you could add or modify the class.

I like what this person said, see how the class children of food all have the same data members? I can organize and reuse the code efficiently because of OOP.

Feels good to shed a little bit of knowledge to some younger people after suffering in my data

structures class

0xDEAD10CC - December 3, 2019, 10:59 pm

(Dec 3, 2019 10:53 PM)Snoopy Kitten Wrote: [\[CITE\]](#) Here are my two cents:

You're right there's no point in putting x in TestClass, however in the real world when you are making applications that are much bigger you will have to worry about memory management. The stack and the heap the stack is automatically allocated by the compiler, for example if you wrote `int x = 1;` it would be stored on the stack and when the program is done it would be automatically deleted.

On the flip side the heap is explicitly controlled by the programmer, so if I created an object: `Classname c = new Classname();` the object © is on the heap, and it is waiting to be destroyed. In C++ you just say `delete c` and you're all good.

I probably didn't do a great job of answering your question but I recommend you looking into data structures like linked lists, stacks, queues etc it might make more sense when you see the organization of data along with OOP

I wanted to add an example:

Imagine you are the manager of a store and you want to manage items in your store.

If you were to approach this without object-oriented programming you'd quickly realize the program itself will become extremely bulky.

PHP Code

```
<?php
class Food {
public:
enum FOOD_TYPE{
Sweet,
Starch,
Meat,
Vegetable,
Fruit
};

Food(std::string str, FOOD_TYPE type, double cost);

FOOD_TYPE getType() const { return m_eType; }

std::string getName() const { return m_sName; }

double getCost() const { return dCost; }

friend std::ostream &operator << (std::ostream &os, const Food &f);

private:
std::string m_sName;
```

```
FOOD_TYPE m_eType;

double dCost;

};

class Sweet : public Food {
public:
    Sweet();

    Sweet(std::string sName, double cost);

    ~Sweet();
};

class Starch : public Food {
public:
    Starch();

    Starch(std::string sName, double cost);

    ~Starch();
};

class Meat : public Food {
public:
    Meat();

    Meat(std::string sName, double cost);

    ~Meat();
};

class Vegetable : public Food {
public:
    Vegetable();

    Vegetable(std::string sName, double cost);

    ~Vegetable();
};

class Fruit : public Food {
public:
    Fruit();

    Fruit(std::string sName, double cost);

    ~Fruit();
};

#endif
```

This simple example demonstrates inheritance, you have a parent class FOOD and children classes sweet, meat, starch etc.

I can now create different food objects and do what I want with them, like stock the store.

I like what this person said, see how the class children of food all have the same data members? I can organize and reuse the code efficiently because of OOP.

Feels good to shed a little bit of knowledge to some younger people after suffering in my data structures class

Why somebody would use an enum like that in addition to polymorphism is beyond me though... Makes no sense to do that. You've also only probably confused people with your C++ example in a Python forum.

koshka1 - December 3, 2019, 11:12 pm

(Dec 3, 2019 11:59 PM)0xDEAD10CC Wrote: [\[CITE\]](#) Why somebody would use an enum like that in addition to polymorphism is beyond me though... Makes no sense to do that. You've also only probably confused people with your C++ example in a Python forum.

It was a project and using the enum 'like that' was apart of the specifications.

Imo the code is so similar that it really doesn't matter what it is, hell we could even do psuedocode if it's confusing OP.

I do concede though that it would probably be best to show these concepts through pictures, I figured the UML was enough but maybe not...

0xDEAD10CC - December 3, 2019, 11:19 pm

(Dec 4, 2019 12:12 AM)Snoopy Kitten Wrote: [\[CITE\]](#) It was a project and using the enum 'like that' was apart of the specifications.

Imo the code is so similar that it really doesn't matter what it is, hell we could even do psuedocode if it's confusing OP.

I do concede though that it would probably be best to show these concepts through pictures, I figured the UML was enough but maybe not...

Diagrams are good, but OP's question is too broad and vague to answer accurately. The use of classes, which promotes encapsulation among other possibilities could spark conversation about framework extensibility and avoiding breaking changes for instance... That alone is already a huge topic, so I'm confused as to what OP is even really asking here... It would be better if he explained which parts he didn't understand than to just tell us that he doesn't understand it.

Lindros - December 4, 2019, 12:01 am

(Dec 3, 2019 09:42 PM)pkdarn Wrote: [\[CITE\]](#) I think I'm starting to grasp it. In what real world situation would you need to use this? What's the point of putting x in TestClass if you could just print x?

Variables are absolutely necessary for storing things that are returned to use later. They are good for abstracting things. You should learn more basic syntax or "grammar" of the language before worrying too much about classes. I'd recommend actually learning more about functions first then dive into classes. Python is an "object oriented" programming language so basically everything is an object or an instance of a class.

Basically a string even is just an instance of a class previously defined elsewhere in Python's default libraries. This is because Python is a higher level, interpreted language. I know you might know some of these statements but sometimes you need to repeat things until you start getting that internal picture in your mind.

(Dec 4, 2019 12:19 AM)0xDEAD10CC Wrote: [\[CITE\]](#) Diagrams are good, but OP's question is too broad and vague to answer accurately. The use of classes, which promotes encapsulation among other possibilities could spark conversation about framework extensibility and avoiding breaking changes for instance... That alone is already a huge topic, so I'm confused as to what OP is even really asking here... It would be better if he explained which parts he didn't understand than to just tell us that he doesn't understand it.

I agree with this, I think the OP is trying to dive into the concepts too quickly before breaking down and internalizing the basics.

Money- - December 4, 2019, 3:18 am

To put it simple though, the whole point of OOP concept is code reusability and maintainability. An object is created via a class with user defined data types and attributes.

Comparing it to variables and you only declare them once then you can assign how many data you want on an array without so much work as compared to variables.

Example, you are making a sales program. You have 5 customers.

Using Variable Concept.

You will obviously create 5 customer variables with 5 names, 5 Customer IDs and 5 emails variables right?

Ain't that very tedious? So this is where object oriented concept comes into play.

Using Objects built via Class.

You will create a class called Customer with attributes called name, ID, email, phone number etc.

So now you will only create 1 object of TYPE Customer or an array object of TYPE Customer with size 5 aka which is 1 line of code to assign 5 customers data such as name, email etc. without declaring 10 or 20 variables just to achieve that.

I hope I answered your question. I assume you are asking what's the benefits of objects compared to variable or how they are different.

I ain't good at writing a good explanation but I am just explaining what I understand.

Edit : If you really want to fully understand how object-oriented concept works. I highly recommend you to play around with Java Language instead of Python.

I learned my OOP concept purely via Java.

magicparty - December 4, 2019, 1:21 pm

Object Oriented Programming is more than just knowing how to make a class or instantiate an object.

Also I would STRONGLY recommend that you learn this in and out with a Java first before applying it to Python or even C++.

PM me if you want to really learn how it works I am extremely proficient in this paradigm of software development.

pkdarn - December 4, 2019, 3:23 pm

(Dec 4, 2019 04:18 AM)Money- Wrote: [\[CITE\]](#) To put it simple though, the whole point of OOP concept is code reusability and maintainability.

An object is created via a class with user defined data types and attributes.

Comparing it to variables and you only declare them once then you can assign how many data you want on an array without so many work as compared to variables.

Example, you are making a sales program. You have 5 customers.

Using Variable Concept.

You will obviously create 5 customer variables with 5 names , 5 Customer IDs and 5 emails variables right?

Ain't that very tedious? So this is where object oriented concept comes into play.

Using Objects built via Class.

You will create a class called Customer with attributes called name , ID , email , phone number etc.

So now you will only create 1 object of TYPE Customer or an array object of TYPE Customer with size 5 aka which is 1 line of code to assign 5 customers data such as name , email etc. without declaring 10 or 20 variables just to achieve that.

I hope I answered your question. I assume you are asking what's the benefits of objects compared to variable or how they are different.

I ain't good at writing a good explanation but I am just explaining what I understand.

Edit : If you really want to fully understand how object-oriented concept works. I highly recommend you to play around with Java Language instead of Python.

I learned my OOP concept purely via Java.

Wow! Thanks so much - to you and everyone else who replied. I think my main problem was that I didn't understand WHEN I should use classes, but these examples have been amazingly helpful. Would I be correct in saying that a class would be like a character in a video game, and everything in the class are things that a player can customize - like hair color, gender, height, etc - whenever a new game is started?

Money- - December 4, 2019, 11:00 pm

(Dec 4, 2019 04:23 PM)pkdarn Wrote: [\[CITE\]](#) Wow! Thanks so much - to you and everyone else who replied. I think my main problem was that I didn't understand WHEN I should use classes, but these examples have been amazingly helpful. Would I be correct in saying that a class would be like a character in a video game, and everything in the class are things that a player can customize - like hair color, gender, height, etc - whenever a new game is started?

Its actually the character is the object. The class has your hair , gender , height and skill tree which you can change, customize.

Like you mention before class is a template or blueprint for an object.

Well actually when you implement or put OOP into work normally. You will think of what classes to create instead of objects.

Say you have a sales program. Obviously Customer comes in mind right? Then you create a customer class and so on according to your logic or your project.

You want a Employee? Then create an employee class.

The object is just making the blueprint or template comes to life. <-- Well this part you actually need to look at the coding syntax to understand.

I highly recommend Java for OOP concept instead of Python if you really want to practice your OOP knowledge.

Good luck have fun.