JOMHACK
INSPIRE INNOVATION

PYTHON
BOOTCAMP

www.jomhack.com

CLASSES AND OBJECTS

JOMHACK
INSPIRE INNOVATION

Classes:

- A template of creating object

Objects:

- Instance of a classes

```python
# Basic class definition
class Person:
    # Class attribute (shared by all instances)
    species = "Homo sapiens"

    # Constructor method
    def __init__(self, name, age):
        # Instance attributes
        self.name = name
        self.age = age

    # Instance method
    def introduce(self):
        return f"Hi, I'm {self.name} and I'm {self.age} years old."

    # Method with parameters
    def have_birthday(self):
        self.age += 1
        return f"Happy birthday! {self.name} is now {self.age}."
```

```python
# Creating objects (instances)
person1 = Person("Alice", 25)
person2 = Person("Bob", 30)

# Accessing attributes
print(person1.name)  # "Alice"
print(person1.age)   # 25

# Calling methods
print(person1.introduce())
print(person1.have_birthday())

# Class attributes
print(Person.species)  # "Homo sapiens"
print(person1.species)  # "Homo sapiens"
```

CLASSES AND OBJECTS

JOMHACK
INSPIRE INNOVATION

```python
class BankAccount:
    def __init__(self, account_number, owner, balance=0):
        self.account_number = account_number
        self.owner = owner
        self.balance = balance
        self.transaction_history = []

    def deposit(self, amount):
        if amount > 0:
            self.balance += amount
            self.transaction_history.append(f"Deposited ${amount}")
            return f"Deposited ${amount}. New balance: ${self.balance}"
        else:
            return "Invalid deposit amount"

    def withdraw(self, amount):
        if amount > 0 and amount <= self.balance:
            self.balance -= amount
            self.transaction_history.append(f"Withdrew ${amount}")
            return f"Withdrew ${amount}. New balance: ${self.balance}"
        else:
            return "Invalid withdrawal amount or insufficient funds"

    def get_balance(self):
        return f"Current balance: ${self.balance}"

    def get_transaction_history(self):
        return self.transaction_history
```

```python
# Using the BankAccount class
account = BankAccount("12345", "Alice", 1000)
print(account.deposit(500))
print(account.withdraw(200))
print(account.get_balance())
```

CLASSES AND OBJECTS

JOMHACK
INSPIRE INNOVATION

Exercise:

1. Create a simple game character class with health, attack and heal methods.