week-0

August 4, 2023

[1]:
```
print("hello world")
```

```
hello world
```

[8]:
```
# Ask user for their name
name = input("what's your name? ")
#say hello to user
print("hello, " + name)
```

```
hello, ahmed
```

[11]:
```
# Ask user for their name
name = input("what's your name? ")
#say hello to user
print("hello,", name)
```

```
hello, ahmed
```

[13]:
```
# Ask user for their name
name = input("what's your name? ")
#say hello to user
print("hello,",end=" ")
print(name)
```

```
hello, ahmed
```

[14]:
```
print("hello, \"friend\"")
```

```
hello, "friend"
```

[15]:
```
print(f"hello, {name}")
```

```
hello, ahmed
```

[18]:
```
# Ask user for their name
name = input("what's your name? ").strip().title()
#say hello to user
print("hello,",end=" ")
```

```
print(name)
```

```
hello, Ahmed Darwish
```

[21]:
```
# Ask user for their name
name = input("what's your name? ")
name = name.strip()
name = name.title()
#say hello to user
print("hello,",name)
```

```
hello, Ahmed Darwish
```

[23]:
```
# Ask user for their name
name = input("what's your name? ").strip().title()
first, last = name.split( )
#say hello to user
print(f"hello, {last}")
```

```
hello, Darwish
```

[28]:
```
x = int(input("what is x? "))
y = int(input("what is y? "))
print(x+y)
```

```
3
```

[30]:
```
x = float(input("what is x? "))
y = float(input("what is y? "))
z = round(x+y)
print(f"{z:,}")
```

```
1,000
```

[32]:
```
x = float(input("what is x? "))
y = float(input("what is y? "))
z = x/y
print(round(z,2))
print(f"{z:.2f}")
```

```
1.17
1.17
```

[36]:
```
def hello(to="world"):
    print("hello,", to)
hello()
name = input("what is your name? ")
hello(name)
```

```
hello, world
hello, ahmed
```

[37]:
```
def main():
    name = input("what is your name? ")
    hello(name)

def hello(to="world"):
    print("hello,", to)

main()
```

```
hello, ahmed
```

[40]:
```
def main():
    x = int(input("whats x?"))
    print("x squared is", square(x))

def square(n):
    return pow(n,2)
main()
```

```
x squared is 25
```

week-1

August 4, 2023

[1]:
```
x = int(input("what is x? "))
y = int(input("what is y? "))

if x<y:
    print("x is less than y")
if x>y:
    print("x is greater than y")
if x==y:
    print("x is equal to y")
```

```
x is less than y
```

[2]:
```
x = int(input("what is x? "))
y = int(input("what is y? "))

if x<y:
    print("x is less than y")
elif x>y:
    print("x is greater than y")
elif x==y:
    print("x is equal to y")
```

```
x is less than y
```

[3]:
```
x = int(input("what is x? "))
y = int(input("what is y? "))

if x<y:
    print("x is less than y")
elif x>y:
    print("x is greater than y")
else:
    print("x is equal to y")
```

```
x is less than y
```

[4]:
```
x = int(input("what is x? "))
y = int(input("what is y? "))
```

```
if x < y or x > y:
    print("x is not equal to y")
else:
    print("x is equal to y")
```

```
x is not equal to y
```

[5]:
```
x = int(input("what is x? "))
y = int(input("what is y? "))

if x != y:
    print("x is not equal to y")
else:
    print("x is equal to y")
```

```
x is not equal to y
```

[6]:
```
score = int(input("score: "))

if score >= 90 and score <= 100:
    print("Grade: A")
elif score >= 80 and score < 90:
    print("Grade B")
elif score >= 70 and score < 80:
    print("Grade C")
elif score >= 60 and score < 70:
    print("Grade D")
else:
    print("Grade: F")
```

```
Grade: F
```

[7]:
```
score = int(input("score: "))

if 90 <= score <= 100:
    print("Grade: A")
elif  80 <= score <= 90:
    print("Grade B")
elif 70 <= score <= 80:
    print("Grade C")
elif 60 <= score <= 70:
    print("Grade D")
else:
    print("Grade: F")
```

```
Grade: F
```

[8]:
```
score = int(input("score: "))

if score >= 90:
    print("Grade: A")
elif  score >= 80:
    print("Grade B")
elif score >= 70:
    print("Grade C")
elif score >= 60:
    print("Grade D")
else:
    print("Grade: F")
```

```
Grade: F
```

[10]:
```
x = int(input("what is x? "))

if x % 2 ==0:
    print("Even")
else:
    print("Odd")
```

```
Odd
```

[13]:
```
def main():
    x = int(input("what is x? "))
    if is_even(x):
        print("Even")
    else:
        print("Odd")

def is_even(n):
    if n % 2 ==0:
        return True
    else:
        return False

main()
```

```
Even
```

[14]:
```
def main():
    x = int(input("what is x? "))
    if is_even(x):
        print("Even")
    else:
        print("Odd")
```

```
def is_even(n):
    return True if n % 2 == 0 else False

main()
```

```
Odd
```

[15]:
```
def main():
    x = int(input("what is x? "))
    if is_even(x):
        print("Even")
    else:
        print("Odd")

def is_even(n):
    return n % 2 == 0

main()
```

```
Odd
```

[16]:
```
name = input("what is yout name? ")

if name == "Harry":
    print("Gryffindor")
elif name == "Hermione":
    print("Gryffindor")
elif name == "Ron":
    print("Gryffindor")
elif name == "Draco":
    print("Slytherin")
else:
    print("who?")
```

```
who?
```

[17]:
```
name = input("what is yout name? ")

if name == "Harry" or name == "Hermione" or name == "Ron":
    print("Gryffindor")
elif name == "Draco":
    print("Slytherin")
else:
    print("who?")
```

```
who?
```

[18]:
```
name = input("what is yout name? ")

match name:
    case "Harry":
        print("Gryffindor")
    case "Hermione":
        print("Gryffindor")
    case "Ron":
        print("Gryffindor")
    case "Draco":
        print("Slytherin")
    case _:
        print("who?")
```

```
who?
```

[19]:
```
name = input("what is yout name? ")

match name:
    case "Harry" | "Hermione" | "Ron":
        print("Gryffindor")
    case "Draco":
        print("Slytherin")
    case _:
        print("who?")
```

```
Gryffindor
```

week-2

August 4, 2023

[1]:
```
print("meow")
print("meow")
print("meow")
```

```
meow
meow
meow
```

[3]:
```
i = 3
while i !=0:
    print("meow")
    i -= 1
```

```
meow
meow
meow
```

[5]:
```
i = 0
while i < 3:
    print("meow")
    i += 1
```

```
meow
meow
meow
```

[6]:
```
for i in [0,1,2]:
    print("meow")
```

```
meow
meow
meow
```

[7]:
```
for i in range(3):
    print("meow")
```

```
meow
meow
meow
```

[8]:
```
for _ in range(3):
    print("meow")
```

```
meow
meow
meow
```

[12]:
```
print("meow\n" * 3, end="")
```

```
meow
meow
meow
```

[13]:
```
while True:
    n = int(input("what's n? "))
    if n < 0:
        continue
    else:
        break
```

[15]:
```
while True:
    n = int(input("what's n? "))
    if n>0:
        break

for _ in range(n):
    print("meow")
```

```
meow
meow
```

[17]:
```
def main():
    number = get_number()
    meow(number)

def get_number():
    while True:
        n = int(input("what's n? "))
        if n > 0:
            return n

def meow(n):
    for _ in range(n):
        print("meow")

main()
```

```
meow
```

```
meow
meow
```

[19]:
```
students = ["Hermione","Harry","Ron"]

print(students[0])
print(students[1])
print(students[2])
```

```
Hermione
Harry
Ron
```

[20]:
```
students = ["Hermione","Harry","Ron"]

for student in students:
    print(student)
```

```
Hermione
Harry
Ron
```

[23]:
```
students = ["Hermione","Harry","Ron"]

for i in range(len(students)):
    print(i + 1,students[i])
```

```
1 Hermione
2 Harry
3 Ron
```

[24]:
```
students = ["Hermione","Harry","Ron","Draco"]
houses = ["Gryffindor","Gryffindor","Gryffindor","Slytherin"]
```

[25]:
```
students = {
    "Hermione": "Gryffindor",
    "Harry": "Gryffindor",
    "Ron": "Gryffindor",
    "Draco": "Slytherin"
}

print(students["Hermione"])
```

```
Gryffindor
```

[27]:
```
students = {
    "Hermione": "Gryffindor",
    "Harry": "Gryffindor",
```

```
    "Ron": "Gryffindor",
    "Draco": "Slytherin"
}

for student in students:
    print(student,students[student],sep=", ")
```

```
Hermione, Gryffindor
Harry, Gryffindor
Ron, Gryffindor
Draco, Slytherin
```

[30]:
```
students = [
    {"name":"Hermione","house": "Gryffindor","patronus":"Otter"},
    {"name":"Harry","house": "Gryffindor","patronus":"Stag"},
    {"name":"Ron","house": "Gryffindor","patronus":"Jaack Russel terrior"},
    {"name":"Draco","house": "Slytherin","patronus":"None"}
]

for student in students:
    print(student["name"],student["house"],student["patronus"],sep=", ")
```

```
Hermione, Gryffindor, Otter
Harry, Gryffindor, Stag
Ron, Gryffindor, Jaack Russel terrior
Draco, Slytherin, None
```

[31]:
```
print("#")
print("#")
print("#")
```

```
#
#
#
```

[32]:
```
for _ in range(3):
    print("#")
```

```
#
#
#
```

[33]:
```
def main():
    print_column(3)

def print_column(height):
    for _ in range(height):
        print("#")
```

```
main()
```

```
#
#
#
```

[34]:
```
def main():
    print_column(3)

def print_column(height):
    print("#\n" * height , end="")
main()
```

```
#
#
#
```

[35]:
```
def main():
    print_row(3)

def print_row(width):
    print("?" * width)
main()
```

```
???
```

[36]:
```
def main():
    print_square(3)

def print_square(size):
    for i in range(size):
        for j in range(size):
            print("#", end ="")
        print()


main()
```

```
###
###
###
```

[37]:
```
def main():
    print_square(3)

def print_square(size):
    for i in range(size):
        print("#" * size)
```

```
main()
```

```
###
###
###
```

[38]:
```
def main():
    print_square(3)

def print_square(size):
    for _ in range(size):
        print_row(size)

def print_row(width):
    print("#" * width)


main()
```

```
###
###
###
```

[ ]:
```

```

week-3

August 4, 2023

syntax error is up to me for fix like missing a (

[3]:
```
x = int(input("what's x? "))#cat will result value error
print(f"x is {x}")
```

```
x is 50
```

[4]:
```
# handling value error
try:
    x = int(input("what's x? "))
    print(f"x is {x}")
except ValueError:
    print("x is not an integer")#cat
```

```
x is not an integer
```

[7]:
```
# handling value error
try:
    y = int(input("what's y? "))
except ValueError:
    print("x is not an integer")

print(f"y is {y}") #cat will result name error
```

```
y is 10
```

[10]:
```
try:
    y = int(input("what's y? "))
except ValueError:
    print("y is not an integer")
else:
    print(f"y is {y}")#works only if no errors
```

```
y is not an integer
```

[12]:
```
while True:
    try:
        y = int(input("what's y? "))
    except ValueError:
```

```
        print("y is not an integer")
    else:
        break

print(f"y is {y}")
```

```
y is not an integer
y is not an integer
y is 5
```

[13]:
```
while True:
    try:
        y = int(input("what's y? "))
        break
    except ValueError:
        print("y is not an integer")

print(f"y is {y}")
```

```
y is not an integer
y is 50
```

[15]:
```
def main():
    x = get_int()
    print(f"x is {x}")

def get_int():
    while True:
        try:
            x = int(input("what's x? "))
        except ValueError:
            print("x is not an integer")
        else:
            break
    return x

main()
```

```
x is not an integer
x is not an integer
x is 5
```

[16]:
```
def main():
    x = get_int()
    print(f"x is {x}")

def get_int():
```

```
    while True:
        try:
            x = int(input("what's x? "))
        except ValueError:
            print("x is not an integer")
        else:
            return x

main()
```

```
x is not an integer
x is not an integer
x is 5
```

[17]:
```
def main():
    x = get_int()
    print(f"x is {x}")

def get_int():
    while True:
        try:
            return int(input("what's x? "))
        except ValueError:
            print("x is not an integer")

main()
```

```
x is not an integer
x is not an integer
x is 5
```

[18]:
```
def main():
    x = get_int()
    print(f"x is {x}")

def get_int():
    while True:
        try:
            return int(input("what's x? "))
        except ValueError:
            pass #capture error without printing anything

main()
```

```
x is 5
```

try:
# Some code that might raise an exception

except SomeException:
# Exception handling code
else:
# Code to be executed if no exception occurred in the try block

[19]:
```python
def main():
    x = get_int("whats x? ")
    print(f"x is {x}")

def get_int(prompt):
    while True:
        try:
            return int(input(prompt))
        except ValueError:
            pass #capture error without printing anything

main()
```

```
x is 50
```

week-4

August 4, 2023

[4]:
```
from random import choice

coin = choice(["heads", "tails"])
print(coin)
```

```
heads
```

[5]:
```
import random

coin = random.choice(["heads", "tails"])
print(coin)
```

```
heads
```

[6]:
```
import random

number = random.randint(1,10)
print(number)
```

```
8
```

[7]:
```
import random

cards = ["jack", "queen", "king"]
random.shuffle(cards)

for card in cards:
    print(card)
```

```
queen
king
jack
```

[8]:
```
import statistics

print(statistics.mean([100,90]))
```

```
95
```

[9]:
```
import sys

print("hello, my name is", sys.argv[1])
```

```
hello, my name is --ip=127.0.0.1
```

python name.py Ahmed #sys.argv[1]== Ahemd

[10]:
```
import sys

try:
    print("hello, my name is", sys.argv[1])
except IndexError:
    print("Too few arguements")
```

```
hello, my name is --ip=127.0.0.1
```

[11]:
```
import sys

if len(sys.argv)<2:
    print("Too few errors")
elif len(sys.argv)>2:
    print("too many arguements")
else:
    print("hello, my name is", sys.argv[1])
```

```
too many arguements
```

[1]:
```
import sys

#check for errors
if len(sys.argv)<2:
    print("Too few arguments")
elif len(sys.argv)>2:
    print("Too many arguments")

print("hello, my name is", sys.argv[1]) #bug, this line will always work
```

```
Too many arguments
hello, my name is --ip=127.0.0.1
```

[22]:
```
import sys

# check for errors
if len(sys.argv) < 2:
    sys.exit("Too few arguments")
elif len(sys.argv) > 11: #had to write 11 instead of 2 to avoid error in the␣
 ↪notebook
```

```
    sys.exit("Too many arguments")

print("hello, my name is", sys.argv[1])
```

```
hello, my name is --ip=127.0.0.1
```

“‘python import sys

if len(sys.argv) < 2: sys.exit(“Too few arguments”)

for arg in sys.argv[1:]: print(“hello, my name is”,arg)

Packages: https://pypi.org/

pip install cowsay

[27]:
```
import cowsay
import sys

if len(sys.argv)==11:
    cowsay.cow("hello, "+sys.argv[1])
```

```
  ----------------------
| hello, --ip=127.0.0.1 |
  ======================
                      \
                       \
                         ^__^
                         (oo)\_______
                         (__)\       )\/\
                             ||----w |
                             ||     ||
```

[28]:
```
import cowsay
import sys

if len(sys.argv)==11:
    cowsay.trex("hello, "+sys.argv[1])
```

```
  ----------------------
| hello, --ip=127.0.0.1 |
  ======================
                            \
                             \
                              \
                               \
                                  .-=-==--==--.
                               ..-==" ,'o`)      `.
                             ,'         `"'        \
                            :  (                    `.__‴_
```

pip install requests

https://itunes.apple.com/search?entity=song&limit=1&term=weezer
downloads json file, used to exchange data

[30]:
```
import requests
import sys
```

[45]:
```
import requests
import sys

if len(sys.argv)!=11:
    sys.exit()

response = requests.get("https://itunes.apple.com/search?
 ↪entity=song&limit=1&term=weezer")
print(response)
```

```
<Response [200]>
```

[47]:
```
print(response.json())
```

```
{'resultCount': 1, 'results': [{'wrapperType': 'track', 'kind': 'song',
'artistId': 115234, 'collectionId': 1440878798, 'trackId': 1440879551,
'artistName': 'Weezer', 'collectionName': 'Weezer', 'trackName': "Say It Ain't
So", 'collectionCensoredName': 'Weezer', 'trackCensoredName': "Say It Ain't So",
'artistViewUrl': 'https://music.apple.com/us/artist/weezer/115234?uo=4',
'collectionViewUrl': 'https://music.apple.com/us/album/say-it-aint-
so/1440878798?i=1440879551&uo=4', 'trackViewUrl':
'https://music.apple.com/us/album/say-it-aint-so/1440878798?i=1440879551&uo=4',
'previewUrl': 'https://audio-ssl.itunes.apple.com/itunes-assets/AudioPreview122/
v4/5c/07/84/5c078405-d5db-0762-d346-0f6ae3ccb530/mzaf_5370611585102254803.plus.a
ac.p.m4a', 'artworkUrl30': 'https://is2-ssl.mzstatic.com/image/thumb/Music125/v4
/fc/74/67/fc74674a-1eb0-d50d-33fe-215caee529d1/16UMGIM52971.rgb.jpg/30x30bb.jpg'
, 'artworkUrl60': 'https://is2-ssl.mzstatic.com/image/thumb/Music125/v4/fc/74/67
/fc74674a-1eb0-d50d-33fe-215caee529d1/16UMGIM52971.rgb.jpg/60x60bb.jpg',
'artworkUrl100': 'https://is2-ssl.mzstatic.com/image/thumb/Music125/v4/fc/74/67/
fc74674a-1eb0-d50d-33fe-215caee529d1/16UMGIM52971.rgb.jpg/100x100bb.jpg',
'collectionPrice': 9.99, 'trackPrice': 1.29, 'releaseDate':
'1994-05-10T12:00:00Z', 'collectionExplicitness': 'notExplicit',
'trackExplicitness': 'notExplicit', 'discCount': 1, 'discNumber': 1,
'trackCount': 10, 'trackNumber': 7, 'trackTimeMillis': 258853, 'country': 'USA',
'currency': 'USD', 'primaryGenreName': 'Rock', 'isStreamable': True}]}
```

[49]:
```
import json
import requests
import sys

if len(sys.argv) != 11:
    sys.exit()

response = requests.get(
    "https://itunes.apple.com/search?entity=song&limit=1&term=weezer")
print(json.dumps(response.json(),indent=2))
```

```
{
  "resultCount": 1,
  "results": [
    {
      "wrapperType": "track",
      "kind": "song",
      "artistId": 115234,
      "collectionId": 1440878798,
      "trackId": 1440879551,
      "artistName": "Weezer",
      "collectionName": "Weezer",
```

```
      "trackName": "Say It Ain't So",
      "collectionCensoredName": "Weezer",
      "trackCensoredName": "Say It Ain't So",
      "artistViewUrl": "https://music.apple.com/us/artist/weezer/115234?uo=4",
      "collectionViewUrl": "https://music.apple.com/us/album/say-it-aint-
so/1440878798?i=1440879551&uo=4",
      "trackViewUrl": "https://music.apple.com/us/album/say-it-aint-
so/1440878798?i=1440879551&uo=4",
      "previewUrl": "https://audio-ssl.itunes.apple.com/itunes-assets/AudioPrevi
ew122/v4/5c/07/84/5c078405-d5db-0762-d346-0f6ae3ccb530/mzaf_5370611585102254803.
plus.aac.p.m4a",
      "artworkUrl30": "https://is2-ssl.mzstatic.com/image/thumb/Music125/v4/fc/7
4/67/fc74674a-1eb0-d50d-33fe-215caee529d1/16UMGIM52971.rgb.jpg/30x30bb.jpg",
      "artworkUrl60": "https://is2-ssl.mzstatic.com/image/thumb/Music125/v4/fc/7
4/67/fc74674a-1eb0-d50d-33fe-215caee529d1/16UMGIM52971.rgb.jpg/60x60bb.jpg",
      "artworkUrl100": "https://is2-ssl.mzstatic.com/image/thumb/Music125/v4/fc/
74/67/fc74674a-1eb0-d50d-33fe-215caee529d1/16UMGIM52971.rgb.jpg/100x100bb.jpg",
      "collectionPrice": 9.99,
      "trackPrice": 1.29,
      "releaseDate": "1994-05-10T12:00:00Z",
      "collectionExplicitness": "notExplicit",
      "trackExplicitness": "notExplicit",
      "discCount": 1,
      "discNumber": 1,
      "trackCount": 10,
      "trackNumber": 7,
      "trackTimeMillis": 258853,
      "country": "USA",
      "currency": "USD",
      "primaryGenreName": "Rock",
      "isStreamable": true
    }
  ]
}
```


[51]:
```
import json
import requests
import sys

if len(sys.argv) != 11:
    sys.exit()

response = requests.get(
    "https://itunes.apple.com/search?entity=song&limit=20&term=weezer")

o = response.json()
for result in o["results"]:
```

```
    print(result["trackName"])
```

```
Say It Ain't So
Buddy Holly
Undone - The Sweater Song
Holiday
My Name Is Jonas
In the Garage
Weezer
Only in Dreams
Surf Wax America
The World Has Turned and Left Me Here
No One Else
Lost in the Woods
Sundown
Buddy Holly
Africa
Take on Me
Say It Ain't So
Everybody Wants to Rule the World
Undone - The Sweater Song
Paranoid
```

[55]:
```
# Creating module
# in a file called sayings.py
def main():
    hello("world")
    goodbye("world")

def hello(name):
    print(f"hello, {name}")


def goodbye(name):
    print(f"goodbye, {name}")

if __name__ == "__main__":
    main()
```

```
hello, world
goodbye, world
```

[56]:
```
# in a file called say.py

# from sayings import hello

if len(sys.argv)==11:
```

```
    hello(sys.argv[1])
```

```
hello, --ip=127.0.0.1
```

week-5

August 4, 2023

[1]:
```
#calculator.py
def main():
    x = int(input("whats x? "))
    print("x squared is ", square(x))

def square(n):
    return n * n

if __name__ == "__main__":
    main()
```

```
x squared is  25
```

[2]:
```
# test_calculator.py
# from calculator import square

def main():
    test_square()


def test_square():
    if square(2) != 4:
        print("2 squared was not 4")
    if square(3) != 9:
        print("3 squared was not 9")


if __name__ == "__main__":
    main()
```

[3]:
```
# calculator.py
def main():
    x = int(input("whats x? "))
    print("x squared is ", square(x))


def square(n):
    return n + n
```

```
if __name__ == "__main__":
    main()
```

```
x squared is  6
```

[4]:
```
# test_calculator.py
# from calculator import square

def main():
    test_square()


def test_square():
    if square(2) != 4:
        print("2 squared was not 4")
    if square(3) != 9:
        print("3 squared was not 9")


if __name__ == "__main__":
    main()
```

```
3 squared was not 9
```

“‘python test_calculator.py from calculator import square

def main(): test_square()

def test_square(): assert square(2) == 4 assert square(3) == 9

if **name** == “**main**”: main()

[9]:
```
# test_calculator.py
#from calculator import square



def main():
    test_square()


def test_square():
    try:
        assert square(2) == 4
    except AssertionError:
        print("2 squared was not 4")
    try:
```

```
        assert square(3) == 4
    except AssertionError:
        print("3 squared was not 9")
    try:
        assert square(-3) == 9
    except AssertionError:
        print("-3 squared was not 9")
    try:
        assert square(0) == 0
    except AssertionError:
        print("0 squared was not 0")
if __name__ == "__main__":
    main()
```

```
3 squared was not 9
-3 squared was not 9
```

pip install pytest

[10]:
```
# test_calculator.py
# from calculator import square

def test_square():
    assert square(2) == 4
    assert square(3) == 4
    assert square(-3) == 9
    assert square(0) == 0
```

pytest test_calculator.py

will terminate once it finds an error

[11]:
```
# test_calculator.py
# from calculator import square

def test_positive():
    assert square(2) == 4
    assert square(3) == 4

# test_calculator.py
# from calculator import square


def test_negative():
    assert square(-2) == 4
    assert square(-3) == 9

def test_zero():
    assert square(0) == 0
```

pytest test_calculator.py

[3]:
```
# calculator.py
def main():
    x = (input("whats x? "))
    print("x squared is ", square(x))


def square(n):
    return n * n


#if __name__ == "__main__":
#    main()
```

[4]:
```
# test_calculator.py
# from calculator import square
import pytest

def test_positive():
    assert square(2) == 4
    assert square(3) == 4

# test_calculator.py
# from calculator import square


def test_negative():
    assert square(-2) == 4
    assert square(-3) == 9


def test_zero():
    assert square(0) == 0

def test_str():
    with pytest.raises("TypeError"):
        square("cat")
```

[5]:
```
#hello.py

def main():
    name = input("whats your name? ")
    hello(name)


def hello(to="world"):
```

```
    print("hello,", to) #side effect

if __name__ == "__main__":
    main()
```

```
hello, ahmed
```

[6]:
```
#test_hello.py
# from hello import hello

def test_hello():
    hello("David") == "hello, David" #wont work as hello doesnt return
```

[7]:
```
# hello.py

def main():
    name = input("whats your name? ")
    print(hello(name))


def hello(to="world"):
    return f"hello, {to}"


if __name__ == "__main__":
    main()
```

```
hello, ahmed
```

[9]:
```
#test_hello.py
# from hello import hello

def test_default():
    assert hello() == "hello, world"

def test_argument():
    assert hello("David") == "hello, David"
```

pytest test_hello.py

[10]:
```
#test_hello.py
# from hello import hello

def test_default():
    assert hello() == "hello, world"

def test_argument():
    for name in ["Hermione","Harry","Ron"]:
```

```
        assert hello(name) == f"hello, {name}"
```

mkdir test

code test/test_hello.py

[11]:
```
# test_hello.py
# from hello import hello

def test_default():
    assert hello() == "hello, world"


def test_argument():
    assert hello("David") == "hello, David"
```

code test/___init___.py

pytest test

week-6

August 4, 2023

[1]:
```python
#names.py
names = []

for _ in range(3):
    names.append(input("whats your name? "))

for name in sorted(names):
    print(f"hello, {name}")
```

```
hello, harry
hello, hermione
hello, ron
```

[2]:
```python
# names.py
name = input("whats your name? ")

file = open("names.txt", "w") #file will be rewritten each time
file.write(name)
file.close()
```

code names.txt

[6]:
```python
# names.py
name = input("whats your name? ")

file = open("names.txt", "a") # will be appended
file.write(name)
file.close()
```

[8]:
```python
# names.py
name = input("whats your name? ")

file = open("names.txt", "a")
file.write(f"{name}\n") #start a new line each time
file.close()
```

[9]:
```python
# names.py
name = input("whats your name? ")
```

```
with open("names.txt", "a") as file: #best practice
    file.write(f"{name}\n")  # start a new line each time
```

[12]:
```
with open("names.txt", "r") as file:
    lines = file.readlines()

for line in lines:
    print("hello,", line.rstrip())
```

```
hello, harry
hello, hermione
hello, draco
```

[13]:
```
with open("names.txt", "r") as file:
    for line in file:
        print("hello,", line.rstrip())
```

```
hello, harry
hello, hermione
hello, draco
```

[14]:
```
names = []

with open("names.txt") as file:
    for line in file:
        names.append(line.rstrip())

for name in sorted(names):
    print(f"hello, {name}")
```

```
hello, draco
hello, harry
hello, hermione
```

[19]:
```
with open("names.txt") as file:
    for line in sorted(file):
        print("hello", line.rstrip())
```

```
hello draco
hello harry
hello hermione
```

[20]:
```
names = []

with open("names.txt") as file:
    for line in file:
        names.append(line.rstrip())
```

```
for name in sorted(names,reverse=True):
    print(f"hello, {name}")
```

```
hello, hermione
hello, harry
hello, draco
```

code students.csv

```
Hermione,Gryffindor
Harry,Gryffindor
Ron,Gryffindor
Draco,Slytherin
```

[21]:
```
with open("students.csv") as file:
    for line in file:
        row = line.rstrip().split(",")
        print(f"{row[0]} is in {row[1]}")
```

```
Hermione is in Gryffindor
Harry is in Gryffindor
Ron is in Gryffindor
Draco is in Slytherin
```

[22]:
```
with open("students.csv") as file:
    for line in file:
        name, house = line.rstrip().split(",")
        print(f"{name} is in {house}")
```

```
Hermione is in Gryffindor
Harry is in Gryffindor
Ron is in Gryffindor
Draco is in Slytherin
```

[23]:
```
students = []

with open("students.csv") as file:
    for line in file:
        name, house = line.rstrip().split(",")
        students.append(f"{name} is in {house}")

for student in sorted(students):
    print(student)
```

```
Draco is in Slytherin
Harry is in Gryffindor
Hermione is in Gryffindor
Ron is in Gryffindor
```

[25]:
```
students = []

with open("students.csv") as file:
    for line in file:
        name, house = line.rstrip().split(",")
        student = {}
        student["name"] = name
        student["house"] = house
        students.append(student)

for student in students:
    print(f"{student['name']} is in {student['house']}")
```

```
Hermione is in Gryffindor
Harry is in Gryffindor
Ron is in Gryffindor
Draco is in Slytherin
```

[26]:
```
students = []

with open("students.csv") as file:
    for line in file:
        name, house = line.rstrip().split(",")
        student = {"name":name, "house":house}
        students.append(student)

for student in students:
    print(f"{student['name']} is in {student['house']}")
```

```
Hermione is in Gryffindor
Harry is in Gryffindor
Ron is in Gryffindor
Draco is in Slytherin
```

[28]:
```
students = []

with open("students.csv") as file:
    for line in file:
        name, house = line.rstrip().split(",")
        student = {"name": name, "house": house}
        students.append(student)

def get_name(student):
    return student["name"]

for student in sorted(students, key= get_name):
    print(f"{student['name']} is in {student['house']}")
```

```
Draco is in Slytherin
Harry is in Gryffindor
Hermione is in Gryffindor
Ron is in Gryffindor
```

[29]:
```
students = []

with open("students.csv") as file:
    for line in file:
        name, house = line.rstrip().split(",")
        student = {"name": name, "house": house}
        students.append(student)

# lambda function
for student in sorted(students, key=lambda student: student["name"]):
    print(f"{student['name']} is in {student['house']}")
```

```
Draco is in Slytherin
Harry is in Gryffindor
Hermione is in Gryffindor
Ron is in Gryffindor
```

code students.csv

“‘ Harry,Number Four, Privet Drive Ron,The Burrow Draco,Malfoy Manor

“‘py students = []

with open(“students.csv”) as file: for line in file: name, home = line.rstrip().split(“,”) student = {“name”: name, “home”: home} students.append(student)

for student in sorted(students, key=lambda student: student[“name”]): print(f“{student[‘name’]} is in {student[‘home’]}”)

error

code students.csv

“‘ Harry,“Number Four, Privet Drive” Ron,The Burrow Draco,Malfoy Manor

[34]:
```
import csv

students = []

with open("students.csv") as file:
    reader = csv.reader(file)
    for name,home in reader:
        students.append({"name":name, "home":home})

for student in sorted(students, key=lambda student: student["name"]):
    print(f"{student['name']} is in {student['home']}")
```

```
Draco is in Malfoy Manor
Harry is in Number Four, Privet Drive
Ron is in The Burrow
```

code students.csv

“‘ name,home Harry,“Number Four, Privet Drive” Ron,The Burrow Draco,Malfoy Manor

[37]:
```
import csv

students = []

with open("students.csv") as file:
    reader = csv.DictReader(file)
    for row in reader:
        students.append({"name": row["name"], "home": row["home"]})

for student in sorted(students, key=lambda student: student["name"]):
    print(f"{student['name']} is in {student['home']}")
```

```
Draco is in Malfoy Manor
Harry is in Number Four, Privet Drive
Ron is in The Burrow
```

code students.csv

“‘ name,home

[42]:
```
import csv

name = input("what's your name? ")
home = input("where's yout home? ")

with open("students.csv", "a") as file:
    writer = csv.writer(file)
    writer.writerow([name,home])
```

[44]:
```
import csv

name = input("what's your name? ")
home = input("where's yout home? ")


with open("students.csv", "a") as file:
    writer = csv.DictWriter(file, fieldnames=["name","home"])
    writer.writerow({"name": name, "home": home})
```

code costume1.gif
code costume2.gif

code costumes.py

[45]:
```
from IPython.display import Image
Image(filename='costume1.gif')
```

[45]:
```
<IPython.core.display.Image object>
```

[46]:
```
from IPython.display import Image
Image(filename='costume2.gif')
```

[46]:
```
<IPython.core.display.Image object>
```

“‘py import sys

from PIL import Image

images = []

for arg in sys.argv[1:]: image = Image.open(arg) images.append(image)

images[0].save(“costumes.gif”, save_all=True, append_images=[images[1]],duration=200,loop = 0)

python costumes.py costume1.gif costume2.gif

[47]:
```
from PIL import Image

images = []

while True:
    filename = input("Enter the filename of an image (or 'done' to finish): ")
    if filename == "done":
        break
    try:
        image = Image.open(filename)
        images.append(image)
    except:
        print(f"Error opening {filename}")

if len(images) < 2:
    print("At least two images are required.")
else:
    images[0].save(
        "costumes.gif", save_all=True, append_images=images[1:], duration=200,␣
 ↪loop=0
    )
```

code costumes.gif

[48]:
```
from IPython.display import Image
Image(filename='costumes.gif')
```

[48]: <IPython.core.display.Image object>

week-7

August 4, 2023

1 validate.py

[14]:
```
email = input("what's your email? ").strip()

print(email)

if "@" in email:
    print("Valid")
else:
    print("Invalid")
```

```
ahmedh457@gmail.com
Valid
```

[13]:
```
email = input("what's your email? ").strip()

print(email)

if "@" in email:
    print("Valid")
else:
    print("Invalid")
```

```
@
Valid
```

[15]:
```
email = input("what's your email? ").strip()

print(email)

if "@" in email and "." in email:
    print("Valid")
else:
    print("Invalid")
```

```
@.
Valid
```

[17]:
```
email = input("what's your email? ").strip()

username, domain = email.split("@")

print(email)
print(username)
print(domain)

if (username) and ("." in domain):
    print("Valid")
else:
    print("Invalid")
```

```
ahmed@edu
ahmed
edu
Invalid
```

[18]:
```
email = input("what's your email? ").strip()

username, domain = email.split("@")

print(email)
print(username)
print(domain)

if username and domain.endswith(".edu"):
    print("Valid")
else:
    print("Invalid")
```

```
ahmed@gmail.edu
ahmed
gmail.edu
Valid
```

[19]:
```
import re

email = input("what's your email? ").strip()

print(email)

if re.search("@", email):
    print("Valid")
else:
    print("Invalid")
```

```
@
```

```
Valid
```

```
.       any character except a newline
*       0 or more repetitions
+       1 or more repeitions
?       0 or 1 repetition
{m}     m repetitions
{m,n}   m-n repetitions
```

[21]:
```
import re

email = input("what's your email? ").strip()

print(email)

if re.search(".*@.*", email):
    print("Valid")
else:
    print("Invalid")
```

```
malan@
Valid
```

[22]:
```
import re

email = input("what's your email? ").strip()

print(email)

if re.search(".+@.+", email):
    print("Valid")
else:
    print("Invalid")
```

```
ahmed@
Invalid
```

.+@.+ is equivalent to ..*@..*

[24]:
```
import re

email = input("what's your email? ").strip()

print(email)

if re.search(".+@.+.edu", email):  # bug
    print("Valid")
else:
    print("Invalid")
```

```
malan@harvard?edu
Valid
```

[25]:
```
import re

email = input("what's your email? ").strip()

print(email)

if re.search(r".+@.+\.edu", email):
    print("Valid")
else:
    print("Invalid")
```

```
malan@harvard?edu
Invalid
```

[27]:
```
import re

email = input("what's your email? ").strip()

print(email)

if re.search(r".+@.+\.edu", email):
    print("Valid")
else:
    print("Invalid")
```

```
my email is malan@harvard.edu…
Valid
```

```
^       matches the start of the string
$       matches the end  of the string or just before the newline at the end of the string
```

[29]:
```
import re

email = input("what's your email? ").strip()

print(email)

if re.search(r"^.+@.+\.edu$", email):
    print("Valid")
else:
    print("Invalid")
```

```
my email is malan@harvard.edu.
Invalid
```

[33]:
```
import re

email = input("what's your email? ").strip()

print(email)

if re.search(r"^.+@.+\.edu$", email):
    print("Valid")
else:
    print("Invalid")
```

```
malan@@@harvard.edu
Valid
```

```
[]      set of characters
[^]     complementing the set
```

[35]:
```
import re

email = input("what's your email? ").strip()

print(email)

if re.search(r"^[^@]+@[^@]+\.edu$", email):
    print("Valid")
else:
    print("Invalid")
```

```
malam@@@harvard.edu
Invalid
```

[36]:
```
import re

email = input("what's your email? ").strip()

print(email)

if re.search(r"^[a-zA-z0-9_]+@[a-zA-z0-9_]+\.edu$", email):
    print("Valid")
else:
    print("Invalid")
```

```
david_malan@harvard.edu
Valid
```

[37]:
```
import re

email = input("what's your email? ").strip()
```

```
print(email)

if re.search(r"^\w+@\w+\.edu$", email):
    print("Valid")
else:
    print("Invalid")
```

```
david_malan@harvard.edu
Valid
```

```
\d      decimal digit
\D      not a decimal digit
\s      whitespace charcters
\S      not a whitespace character
\w      word character...as well as numbers and the underscore
\W      not a word character
```

```
A|B         either A or B
(...)       a group
(?:...)     non-capturing version
```

```
re.IGNORECASE
re.MULTILINE
re.DOTALL
```

[38]:
```
import re

email = input("what's your email? ").strip()

print(email)

if re.search(r"^\w+@\w+\.edu$", email, re.IGNORECASE):
    print("Valid")
else:
    print("Invalid")
```

```
malan_david@HARVARD.EDU
Valid
```

[39]:
```
import re

email = input("what's your email? ").strip()

print(email)

if re.search(r"^\w+@\w+\.edu$", email, re.IGNORECASE):
    print("Valid")
else:
```

```
    print("Invalid")
```

```
malan@cs50.harvard.edu
Invalid
```

[40]:
```
import re

email = input("what's your email? ").strip()

print(email)

if re.search(r"^\w+@(\w+\.)?\w+\.edu$", email, re.IGNORECASE):
    print("Valid")
else:
    print("Invalid")
```

```
malan@cs50.harvard.edu
Valid
```

2 format.py

[42]:
```
name = input("what's your name? ").strip()
print(f"hello, {name}")
```

```
hello, malan,david
```

[49]:
```
import re

name = input("what's your name? ").strip()
print(name)

matches = re.search(r"^(.+), *(.+)$", name)

if matches:
    name = matches.group(2) + " " + matches.group(1)

print(f"hello, {name}")
```

```
malan,       david
hello, david malan
```

[50]:
```
import re

name = input("what's your name? ").strip()
print(name)

if matches := re.search(r"^(.+), *(.+)$", name):  # walrus operator
```

```
    name = matches.group(2) + " " + matches.group(1)

print(f"hello, {name}")
```

```
malan,       david
hello, david malan
```

3 twitter.py

[52]:
```
url = input("URL: ").strip()
print(url)

username = url.replace("https://twitter.com/", "")
print(f"Username: {username}")
```

```
https://twitter.com/ahmed
Username: ahmed
```

[54]:
```
url = input("URL: ").strip()
print(url)

username = url.removeprefix("https://twitter.com/")
print(f"Username: {username}")
```

```
https://twitter.com/ahmed
Username: ahmed
```

[56]:
```
import re

url = input("URL: ").strip()
print(url)

username = re.sub(r"https://twitter.com/", "", url)
print(f"username: {username}")
```

```
https://twitter.com/ahmed
username: ahmed
```

[57]:
```
import re

url = input("URL: ").strip()
print(url)

username = re.sub(r"^(https?://)?(www\.)?twitter\.com/", "", url)
print(f"username: {username}")
```

```
twitter.com/ahmed
username: ahmed
```

[58]:
```python
import re

url = input("URL: ").strip()
print(url)

matches = re.search(
    r"^https?://(www\.)?twitter\.com/(.+)$", url, re.IGNORECASE)

if matches:
    print(f"Username:", matches.group(2))
```

```
https://google.com
```

[59]:
```python
import re

url = input("URL: ").strip()
print(url)

matches = re.search(
    r"^https?://(www\.)?twitter\.com/(.+)$", url, re.IGNORECASE)

if matches:
    print(f"Username:", matches.group(2))
```

```
https://twitter.com/ahmed
Username: ahmed
```

[60]:
```python
import re

url = input("URL: ").strip()
print(url)

if matches := re.search(
        r"^https?://(www\.)?twitter\.com/(.+)$", url, re.IGNORECASE):
    print(f"Username:", matches.group(2))
```

```
https://twitter.com/ahmed
Username: ahmed
```

[63]:
```python
import re

url = input("URL: ").strip()
print(url)

if matches := re.search(
```

```
        r"^https?://(?:www\.)?twitter\.com/([a-z0-9_]+)$", url, re.IGNORECASE):
    print(f"Username:", matches.group(1))
```

```
https://twitter.com/ahmeddarwish
Username: ahmeddarwish
```

week-8

August 4, 2023

1 student.py

[1]:
```
name = input("Name: ")
house = input("House: ")
print(f"{name} from {house}")
```

```
harry from gryffindor
```

[2]:
```
def main():
    name = get_name()
    house = get_house()
    print(f"{name} from {house}")


def get_name():
    return input("Name: ")


def get_house():
    return input("House: ")


if __name__ == "__main__":
    main()
```

```
harry from gryffindor
```

[3]:
```
def main():
    name, house = get_student()
    print(f"{name} from {house}")


def get_student():
    name = input("Name: ")
    house = input("House: ")
    return name, house  # tuple
```

```
if __name__ == "__main__":
    main()
```

```
harry from gryffindor
```

[4]:
```
def main():
    student = get_student()
    print(f"{student[0]} from {student[1]}")


def get_student():
    name = input("Name: ")
    house = input("House: ")
    return (name, house)  # tuple


if __name__ == "__main__":
    main()
```

```
harry from gryffindor
```

```
def main():
    student = get_student()
    if student[0] == "padma":
        student[1] = "ravenclaw"
    print(f"{student[0]} from {student[1]}")


def get_student():
    name = input("Name: ")
    house = input("House: ")
    return (name, house)  # tuple


if __name__ == "__main__":
    main()
```

error, tuple is immutable

[9]:
```
def main():
    student = get_student()
    if student[0] == "padma":
        student[1] = "ravenclaw"
    print(f"{student[0]} from {student[1]}")


def get_student():
    name = input("Name: ")
```

```
    house = input("House: ")
    return [name, house]  # list


if __name__ == "__main__":
    main()
```

```
padma from ravenclaw
```

list is mutable

[10]:
```
def main():
    student = get_student()
    print(f"{student['name']} from {student['house']}")


def get_student():
    student = {}
    student["name"] = input("Name: ")
    student["house"] = input("House: ")
    return student  # dict


if __name__ == "__main__":
    main()
```

```
harry from gryffindor
```

[11]:
```
def main():
    student = get_student()
    if student["name"] == "padma":
        student["house"] = "ravenclaw"
    print(f"{student['name']} from {student['house']}")


def get_student():
    name = input("Name: ")
    house = input("House: ")
    return {"name": name, "house": house}  # dict


if __name__ == "__main__":
    main()
```

```
padma from ravenclaw
```

dicts are mutable

[15]:
```
class Student:
    ...

def main():
    student = get_student()
    print(f"{student.name} from {student.house}")

def get_student():
    student = Student()
    student.name = input("Name: ")
    student.house = input("House: ")
    return student


if __name__ == "__main__":
    main()
```

```
harry from gryffindor
```

[16]:
```
class Student:
    def __init__(self, name, house):    #instance method
        self.name = name
        self.house = house

def main():
    student = get_student()
    print(f"{student.name} from {student.house}")


def get_student():
    name = input("Name: ")
    house = input("House: ")
    student = Student(name, house)
    return student


if __name__ == "__main__":
    main()
```

```
harry from gryffindor
```

[17]:
```
class Student:
    def __init__(self, name, house):  # instance method
        self.name = name
        self.house = house
```

```
def main():
    student = get_student()
    print(f"{student.name} from {student.house}")


def get_student():
    name = input("Name: ")
    house = input("House: ")
    return Student(name, house)


if __name__ == "__main__":
    main()
```

```
harry from gryffindor
```

[22]:
```
class Student:
    def __init__(self, name, house):
        if not name:
            raise ValueError("missing name")
        if house not in ["gryffindor", "hufflepuff", "ravenclaw", "slytherin"]:
            raise ValueError("invalid house")
        self.name = name
        self.house = house


def main():
    student = get_student()
    print(f"{student.name} from {student.house}")


def get_student():
    name = input("Name: ")
    house = input("House: ")
    return Student(name, house)


if __name__ == "__main__":
    main()
```

```
harry from gryffindor
```

[23]:
```
class Student:
    def __init__(self, name, house):
        if not name:
            raise ValueError("missing name")
        if house not in ["gryffindor", "hufflepuff", "ravenclaw", "slytherin"]:
```

```
            raise ValueError("invalid house")
        self.name = name
        self.house = house


def main():
    student = get_student()
    print(student)


def get_student():
    name = input("Name: ")
    house = input("House: ")
    return Student(name, house)


if __name__ == "__main__":
    main()
```

```
<__main__.Student object at 0x0000021F82F48790>
```

[26]:
```
class Student:
    def __init__(self, name, house):
        if not name:
            raise ValueError("missing name")
        if house not in ["gryffindor", "hufflepuff", "ravenclaw", "slytherin"]:
            raise ValueError("invalid house")
        self.name = name
        self.house = house

    def __str__(self):
        return " a student"


def main():
    student = get_student()
    print(student)


def get_student():
    name = input("Name: ")
    house = input("House: ")
    return Student(name, house)


if __name__ == "__main__":
    main()
```

a student

[28]:
```
class Student:
    def __init__(self, name, house):
        if not name:
            raise ValueError("missing name")
        if house not in ["gryffindor", "hufflepuff", "ravenclaw", "slytherin"]:
            raise ValueError("invalid house")
        self.name = name
        self.house = house

    def __str__(self):
        return f"{self.name} from {self.house}"


def main():
    student = get_student()
    print(student)


def get_student():
    name = input("Name: ")
    house = input("House: ")
    return Student(name, house)


if __name__ == "__main__":
    main()
```

harry from gryffindor

[30]:
```
class Student:
    def __init__(self, name, house, patronus):
        if not name:
            raise ValueError("missing name")
        if house not in ["gryffindor", "hufflepuff", "ravenclaw", "slytherin"]:
            raise ValueError("invalid house")
        self.name = name
        self.house = house
        self.patronus = patronus

    def __str__(self):
        return f"{self.name} from {self.house}"


def main():
    student = get_student()
```

```
    print(student)


def get_student():
    name = input("Name: ")
    house = input("House: ")
    patronus = input("Patrnous: ")
    return Student(name, house, patronus)


if __name__ == "__main__":
    main()
```

```
harry from gryffindor
```

[2]:
```
class Student:
    def __init__(self, name, house, patronus):
        if not name:
            raise ValueError("missing name")
        if house not in ["gryffindor", "hufflepuff", "ravenclaw", "slytherin"]:
            raise ValueError("invalid house")
        self.name = name
        self.house = house
        self.patronus = patronus

    def __str__(self):
        return f"{self.name} from {self.house}"

    def charm(self):
        match self.patronus:
            case "stag":
                return " "
            case "otter":
                return " "
            case _:
                return" "


def main():
    student = get_student()
    print("Expecto patronum")
    print(student.charm())


def get_student():
    name = input("Name: ")
    house = input("House: ")
```

```
    patronus = input("Patrnous: ")
    return Student(name, house, patronus)


if __name__ == "__main__":
    main()
```

```
Expecto patronum
```

[2]:
```
class Student:
    def __init__(self, name, house):
        if not name:
            raise ValueError("missing name")
        if house not in ["gryffindor", "hufflepuff", "ravenclaw", "slytherin"]:
            raise ValueError("invalid house")
        self.name = name
        self.house = house

    def __str__(self):
        return f"{self.name} from {self.house}"



def main():
    student = get_student()
    student.house = "Number four" #problem
    print(student)


def get_student():
    name = input("Name: ")
    house = input("House: ")
    return Student(name, house)


if __name__ == "__main__":
    main()
```

```
harry from Number four
```

[3]:
```
class Student:
    def __init__(self, name, house):
        if not name:
            raise ValueError("missing name")
        self.name = name
        self.house = house
```

```
    def __str__(self):
        return f"{self.name} from {self.house}"

    #getter
    @property
    def house(self):
        return self._house

    #setter
    @house.setter
    def house(self,house):
        if house not in ["gryffindor", "hufflepuff", "ravenclaw", "slytherin"]:
            raise ValueError("invalid house")
        self._house = house


def main():
    student = get_student()
    student.house = "hufflepuff" #will call setter
    print(student)


def get_student():
    name = input("Name: ")
    house = input("House: ")
    return Student(name, house)


if __name__ == "__main__":
    main()
```

```
harry from hufflepuff
```

[5]:
```
class Student:
    def __init__(self, name, house):
        self.name = name
        self.house = house

    def __str__(self):
        return f"{self.name} from {self.house}"

    # getter
    @property
    def name(self):
        return self._name
```

```
    @property
    def house(self):
        return self._house

    # setter
    @name.setter
    def name(self, name):
        if not name:
            raise ValueError("missing name")
        self._name = name

    @house.setter
    def house(self, house):
        if house not in ["gryffindor", "hufflepuff", "ravenclaw", "slytherin"]:
            raise ValueError("invalid house")
        self._house = house


def main():
    student = get_student()
    print(student)


def get_student():
    name = input("Name: ")
    house = input("House: ")
    return Student(name, house)


if __name__ == "__main__":
    main()
```

```
harry  from gryffindor
```

[6]:
```
class Student:
    def __init__(self, name, house):
        self.name = name
        self.house = house

    def __str__(self):
        return f"{self.name} from {self.house}"

    # getter
    @property
    def name(self):
        return self._name
```

```
    @property
    def house(self):
        return self._house

    # setter
    @name.setter
    def name(self, name):
        if not name:
            raise ValueError("missing name")
        self._name = name

    @house.setter
    def house(self, house):
        if house not in ["gryffindor", "hufflepuff", "ravenclaw", "slytherin"]:
            raise ValueError("invalid house")
        self._house = house


def main():
    student = get_student()
    student._house = "number four" #plz dont do that
    print(student)


def get_student():
    name = input("Name: ")
    house = input("House: ")
    return Student(name, house)


if __name__ == "__main__":
    main()
```

```
harry from number four
```

2 type.py

[7]:
```
print(type(50))
```

```
<class 'int'>
```

[8]:
```
print(type("hello world"))
```

```
<class 'str'>
```

[9]:
```
print(type([]))
```

```
<class 'list'>
```

[10]:
```
print(type(list()))
```

```
<class 'list'>
```

[11]:
```
print(type({}))
```

```
<class 'dict'>
```

[12]:
```
print(type(dict()))
```

```
<class 'dict'>
```

3 hat.py

[13]:
```
class Hat:
    ...


hat = Hat()
```

[14]:
```
class Hat:
    def sort(self, name):
        print(name, "is in", "some house")


hat = Hat()
hat.sort("Harry")
```

```
Harry is in some house
```

[15]:
```
import random


class Hat:
    def __init__(self):
        self.houses = ["gryffindor", "hufflepuff", "ravenclaw", "slytherin"]

    def sort(self, name):
        print(name, "is in", random.choice(self.houses))


hat = Hat()
hat.sort("Harry")
```

```
Harry is in gryffindor
```

[23]:
```
import random


class Hat:
    houses = ["gryffindor", "hufflepuff", "ravenclaw", "slytherin"]

    @classmethod
    def sort(cls, name):
        print(name, "is in", random.choice(cls.houses))


Hat.sort("Harry")
```

```
Harry is in gryffindor
```

4 Student.py

[25]:
```
class Student:
    def __init__(self, name, house):
        self.name = name
        self.house = house

    def __str__(self):
        return f"{self.name} from {self.house}"

    @classmethod
    def get(cls):
        name = input("Name: ")
        house = input("House: ")
        return cls(name, house)


def main():
    student = Student.get()
    print(student)


if __name__ == "__main__":
    main()
```

```
harry from gryffindor
```

5 wizard.py

[1]:
```
class Student:
    def __init__(self, name, house):
        self.name = name
        self.house = house

class Professor:
    def __init__(self,name,subject):
        self.name = name
        self.subject = subject
```

[5]:
```
class Wizard:
    def __init__(self,name):
        if not name:
            raise ValueError()
        self.name = name


class Student(Wizard):
    def __init__(self, name, house):
        super().__init__(name)
        self.house = house


class Professor(Wizard):
    def __init__(self, name, subject):
        super().__init__(name)
        self.subject = subject

wizard = Wizard("ALbus")
student = Student("Harry","Gryffindor")
print(student.name)
```

```
Harry
```

6 valut.py

[9]:
```
class Vault:
    def __init__(self,galleons=0,sickles=0,knuts=0):
        self.galleons = galleons
        self.sickles = sickles
        self.knuts = knuts

    def __str__(self):
        return f" {self.galleons}, {self.sickles}, {self.knuts}"
```

```
potter = Vault(100,50,25)
print(potter)

weasley = Vault(25,50,100)
print(weasley)

galleons = potter.galleons + weasley.galleons
total = Vault(galleons)
print(total)
```

```
100, 50, 25
25, 50, 100
125, 0, 0
```

[10]:
```
class Vault:
    def __init__(self, galleons=0, sickles=0, knuts=0):
        self.galleons = galleons
        self.sickles = sickles
        self.knuts = knuts

    def __str__(self):
        return f"{self.galleons} Galleons, {self.sickles} Sickles, {self.knuts}␣
 ↪Knuts"

    def __add__(self, other):
        galleons = self.galleons + other.galleons
        sickles = self.sickles + other.sickles
        knuts = self.knuts + other.knuts
        return Vault(galleons, sickles, knuts)


potter = Vault(100, 50, 25)
print(potter)

weasley = Vault(25, 50, 100)
print(weasley)

total = potter + weasley
print(total)
```

```
100 Galleons, 50 Sickles, 25 Knuts
25 Galleons, 50 Sickles, 100 Knuts
125 Galleons, 100 Sickles, 125 Knuts
```

[ ]:
```

```