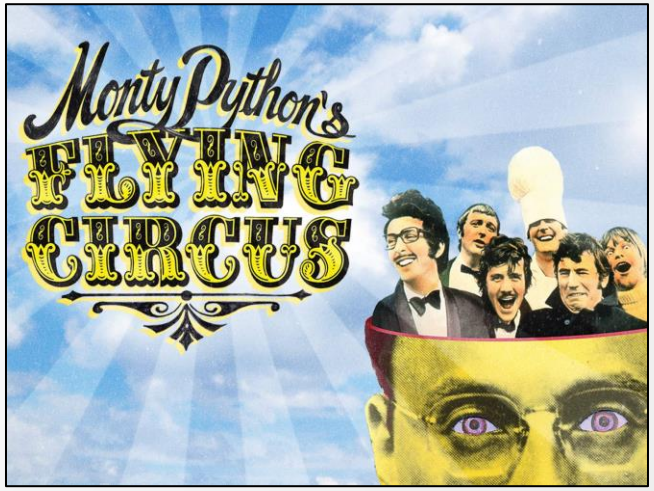
Introduction To Python

- What Is Python And History Of Python
- Unique Features Of Python
- Python-2 And Python-3 Differences
- Install Python And Environmental Setup
- First Python Program
- Python Identifiers, Keywords, Indentation
- Comments And Document Include In Python
- Command Line Arguments
- Getting User Input
- Python Basic Data Types
- What Are Variables

What Is Python and History Of Python

- Since 1991
- Found by Guido Van Rossum
- Named after a comedy series called Monty Python's Flying Circus
- The goal was to keep the programming language simple



Unique Features Of Python

- Free and Open source
- Interpreted language
- Object-Oriented
- High-level language
- Portable

Python-2 And Python-3 Differences

Python 2	Python 3
print “hello”	print(“hello”)
7/2 = 3	7/2 = 3.5
xrange(5)	range(5)
try: … except err, err: …	try: … except err ***as*** err: …
b’abcdef’ – type str	b’abcdef’ – type bytes

Install Python And Environmental Setup

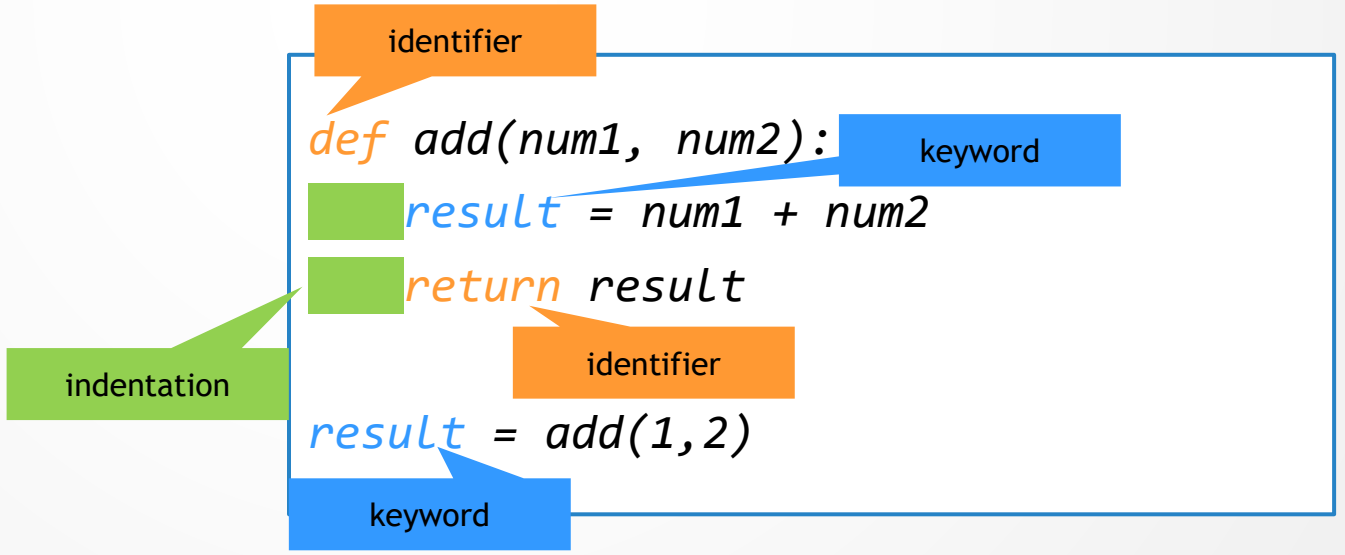
- https://www.python.org/downloads/

First Python Program

```
print('hello world')
```

Python Identifiers, Keywords, Indentation

```
def add(num1, num2):
    result = num1 + num2
    return result

result = add(1,2)
```



Comments And Document Include In Python

```
# this is a single line comment
```

```
'''
multi line comment
line 1
line 2
'''
```

```
"""
multi line comment
line 1
line 2
"""
```

Command Line Arguments

1 2 3 are passed in command line

Accessing them within python script

```
>python ex1.py 1 2 3
['ex1.py', '1', '2', '3']
```

ex1.py

```
import sys

print(sys.argv)
```

Getting User Input

```
>python slide10.py
Hi, What is your name? jessy
Hope you are doing good jessy

>python slide10.py
Hi, What is your name? karthik
Hope you are doing good karthik
```

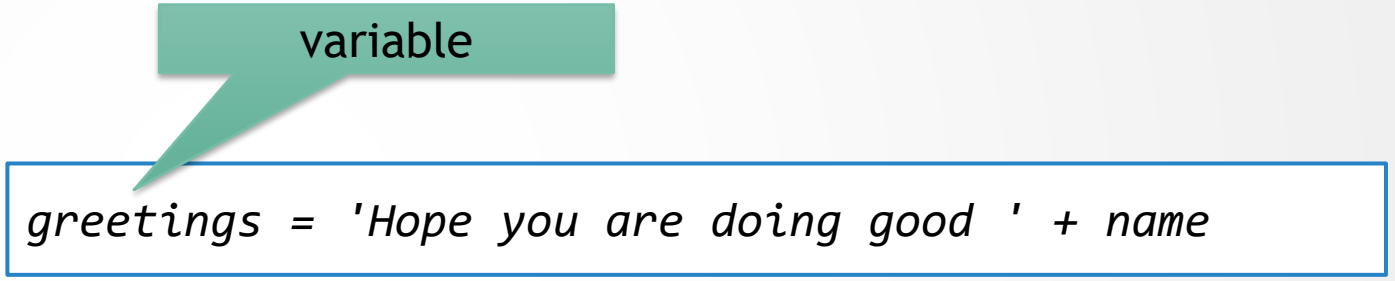
slide10.py

```
name = input('Hi, What is your name? ')

greetings = 'Hope you are doing good ' + name

print(greetings)
```

Python Basic Data Types

Category	Type
Text	Str
Numeric	Int, float, complex
Sequence	List, tuple, range
Key-pair	Dict
Set	Set, frozenset
Boolean	Bool
Binary	Bytes, bytearray, memoryview

What Are Variables



List, Ranges & Tuples In Python

- Introduction
- Lists In Python
- More About Lists
- Understanding Iterators
- Generators, Comprehensions, Lambda Expressions
- Understanding And Using Ranges

Lists In Python

```
Fruit_basket = [‘apple’, ‘orange’, ‘lemon’]

Numbers = [1, 2, 3, 4]

Mix = [‘lizzie’, 4081, ‘dutchess’]
```

More About Lists

- Refer slide15.py

Exercise

- Find the second largest number in a list

```
Nums = [3,4,6,2,7]
```

- Second largest number is 6 in this list

Exercise

Input:
[1, 2, 3, 4, 4, 9, 7, 4]
4

Output:
12

Explanation:
4 has occurred 3 times in the list, so 4*3 = 12

case 1:
Input = 4
Output = 12

Case 2:
Input = 9
Output = 9

Case 3:
Input = 5
Output = 0

Understanding Iterators

- an object that can itered (traversed through is called iterator
- Example: list, dict, set, tuple etc
- They have __iter__() & __next__() method
- *String are also iterable objects*

Understanding Iterators

```
nums = [1,2,3]
obj = iter(nums)
next(obj)
next(obj)

animal = ‘lion’
obj = iter(animal)
next(obj)
```

Generators, Comprehensions, Lambda Expressions

Generators

- To temporarily save the state of the function
- Uses yield statement

```
def my_gen():
        for num in range(3):
                yield num

gen = my_gen()
print(next(gen))
print(next(gen))
print(next(gen))
print(next(gen))
```

Generators, Comprehensions, Lambda Expressions

Generators - Exercise

- What's the difference between yield & return statement ?
- What will happen if you replace yield in above code snippet to return ?

Generators, Comprehensions, Lambda Expressions

Comprehensions

- From the list numbers, get only even numbers
- How would you do that ?

```
nums = [1,2,3,4,5,6,7,8,9,10]

even_nums = []

for num in nums:
    if num % 2 == 0:
        even_nums.append(num)

print(even_nums)
[2, 4, 6, 8, 10]
```

Generators, Comprehensions, Lambda Expressions

Comprehensions

- Using comprehensions

```
nums = [1,2,3,4,5,6,7,8,9,10]

evens = [ num for num in nums if num % 2 == 0 ]

print(evens)
```

Generators, Comprehensions, Lambda Expressions

Comprehensions - Exercise

#1 print even numbers for the given input range

- Get input number from the user
- Print the even numbers from that range of numbers

HINT use ***input()*** to get user input. Use **list comprehension** with ***range()***. Convert user input to **int**.

NOTE you can print in the format you like. Try this format too if you can find it how to do it.

Examples – next slide

Generators, Comprehensions, Lambda Expressions

Comprehensions - Exercise

#1 print even numbers for the given input range
Example

```
>python ex.py
Enter a number: 20
The even numbers in the range of 20 are
0  2  4  6  8  10  12  14  16  18

>python ex.py
Enter a number: 15
The even numbers in the range of 15 are
0  2  4  6  8  10  12  14

>python ex.py
Enter a number: 34
The even numbers in the range of 34 are
0  2  4  6  8  10  12  14  16  18  20  22  24  26
28  30  32
```

Generators, Comprehensions, Lambda Expressions

Comprehensions - Exercise

#2 from a given list, print only elements which starts with a given input letter

```
tamil_actors = [‘vijay’, ‘suriya’, ‘str’, ‘dhanush’, ‘kamal’, ‘vikram’, ‘madhavan’, ‘sivakumar’, ‘prabhu’, ‘siddharth’]
```

```
>python ex.py
enter a letter: s
['suriya', 'str', 'sivakumar', 'siddharth’]

>python ex.py
enter a letter: v
['vijay', 'vikram']
```

Generators, Comprehensions, Lambda Expressions

Comprehensions - Exercise

#3 print only the words which has vowel letters in it. Also print the words which don't have vowels

```
my_str='a fox jumps over the lazy dog sp'
```

```
>python ex.py

{'fox', 'over', 'dog', 'the', 'lazy', 'a',
'jumps'}

{'sp'}
```

Generators, Comprehensions, Lambda Expressions

Lambda Expressions

- Anonymous function
- Any number of arguments
- Only one expression

Generators, Comprehensions, Lambda Expressions

Lambda Expressions

slide28.py

```
x = lambda a : a + 10
print(x(10))

y = lambda a,b,c: a+b+c
print(y(1,1,1))
```

```
>python slide28.py
20
3
```

Understanding And Using Ranges

Ranges

- Generate numbers till 10

slide30.py

```
for num in range(11):
    print(num)
```

- Generate numbers from 2 to 10

slide30.py

```
for num in range(2, 11):
    print(num)
```

- Generate even numbers from 2 to 10

slide30.py

```
for num in range(2, 11, 2):
    print(num)
```

Mutable and Immutable

- Str is immutable. Why?

slide31.py

```
quote = 'a fox jumps over the lazy dog'
quote.replace(' ', '') # replaced spaces

print(quote)
```

- What will be the output here?

Mutable and Immutable

- list is immutable. Why?

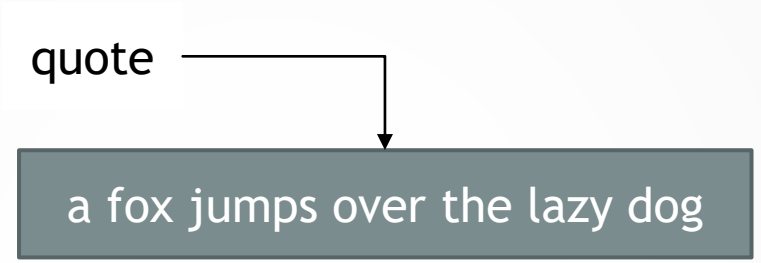
slide32.py

```
nums = [1,2,3,4,5]
nums.append(6)

print(nums)
```

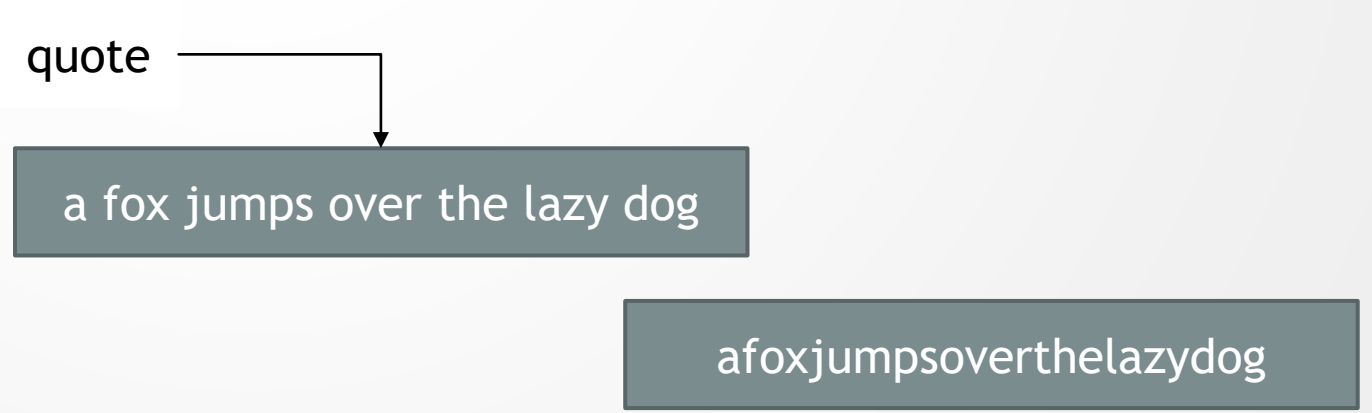
- What will be the output here?

Mutable and Immutable

- Why ?
- quote is a variable pointing to a memory location with the value

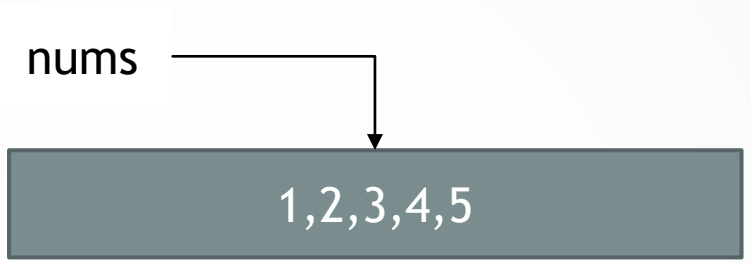


- On quote.replace(' ', ‘’) it creates the new value in different memory location, which quote is not pointing to

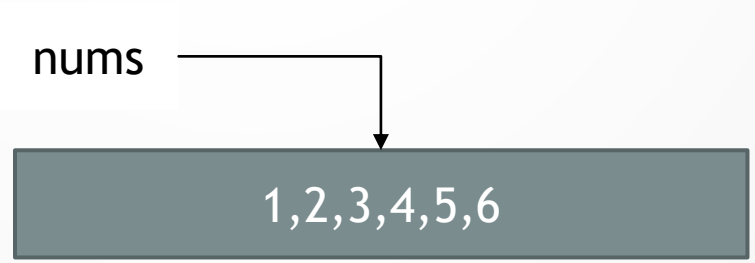


Mutable and Immutable

- Why ?
- nums is a variable pointing to a memory location with the value



- On nums.append(6), 6 is appended to the same memory location



Mutable and Immutable

Type	Immutable
Bool	Y
Int	Y
Float	Y
List	
Tuple	Y
Str	Y
Set	
Frozenset	Y
Dict	

Python Dictionaries And Sets

- Introduction
- Python Dictionaries
- More On Dictionaries
- Sets
- Python Sets Examples

Python Dictionaries

- Key-pair value

slide37.py

```
student1 = {'name': 'jessy', 'gender': 'female',
'age': 26}
student2 = {'name': 'karthik', 'gender': 'male',
'age': 25}


print(student1)
print(student1['name'])
print(student2['age'])
```

```
>python slide37.py
{'name': 'jessy', 'gender': 'female', 'age': 26}
jessy
25
```

More On Dictionaries

- Refer slide38.py

More On Dictionaries

Dictionaries - Exercise

- What is dictionary comprehension
- Check if a key exists in a Python dictionary
- Sort a dict

- Try out dictionary operations

Sets

- Sets are declared using { }

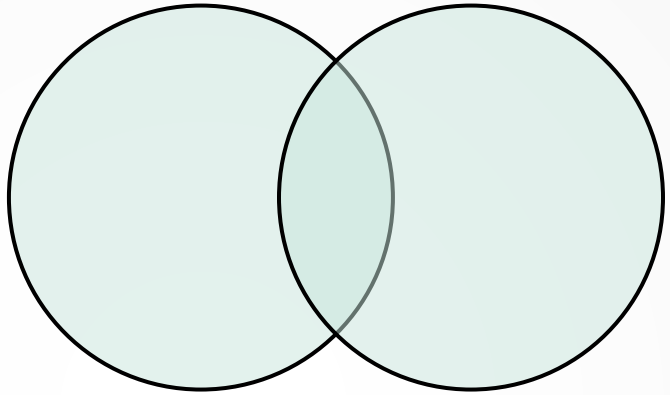
```
p = {1,2,3}

print(type(p))
```

- Set can't have duplicate values

```
nums = [1,1,1]

print(set(nums))
```

Python Sets Examples

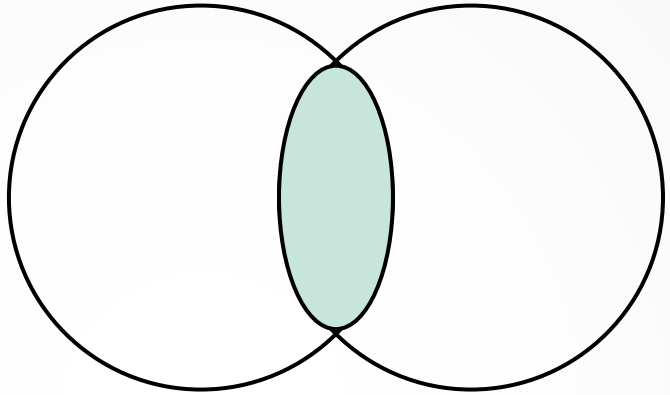
- Set union



```
Odd_nums = {0,1,3,5,7,9}
Even_nums = {0,2,4,6,8,10}

Nums = odd_nums | even_nums
```

Python Sets Examples

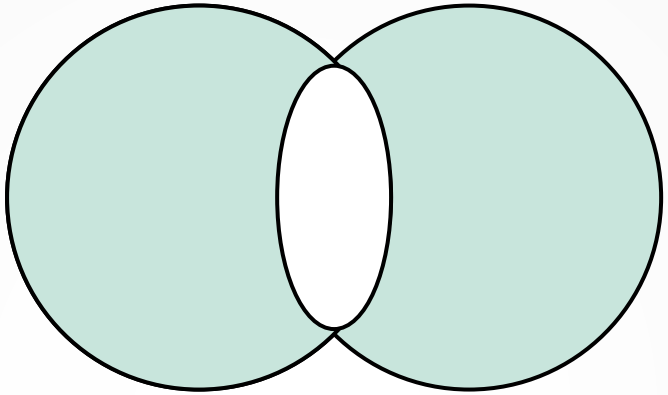
- Set Intersection



```
Odd_nums = {0,1,3,5,7,9}
Even_nums = {0,2,4,6,8,10}

Nums = odd_nums & even_nums
```

Python Sets Examples

- Set Symmetric Difference



```
odd_nums = {0,1,3,5,7,9}
even_nums = {0,2,4,6,8,10}

nums = odd_nums ^ even_nums
```

Python Sets Examples

- Other set operations

Refer slide43.py

Input And Output In Python

- Reading Text Files
- Writing Text Files
- Appending To Files And Change
- Writing Binary Files Manually
- Using Pickle To Write Binary Files

Reading Text Files

- Open a file in read mode

```
fp = open('mytxtfile.txt', 'r')
```

- Read all the lines in the file

```
content = fp.read()
print(content)
```

- Read single line

```
line = fp.readline()
print(line)
```

- To read line-by-line

```
for line in fp:
    print(line)
```

Reading Text Files

- Check if the file is closed

```
print(fp.closed)
```

- Close the file

```
fp.close()
```

- Automatically close the file by using ***with***

```
with open('mytxtfile.txt', 'r') as fp:
    for line in fp:
        print(line)
```

Reading Text Files

- Read n character

```
content = fp.read(3)
print(content)
```

- Tell where the file pointer is

```
fp.tell()
```

- Bring the file pointer to a position

```
fp.seek(1)  # seeks to position 1
```

Writing Text Files

- Open file with write mode

```
fp = open('mytxtfile.txt', 'w')
```

- Write a string value to it

```
value = 'abc'
fp.write(value)
```

- Mode ‘w’ truncates the file
- Use ‘a’ in that case
- Use ‘w+’ to open with both write and read

Appending To Files And Change

- Open file with write mode

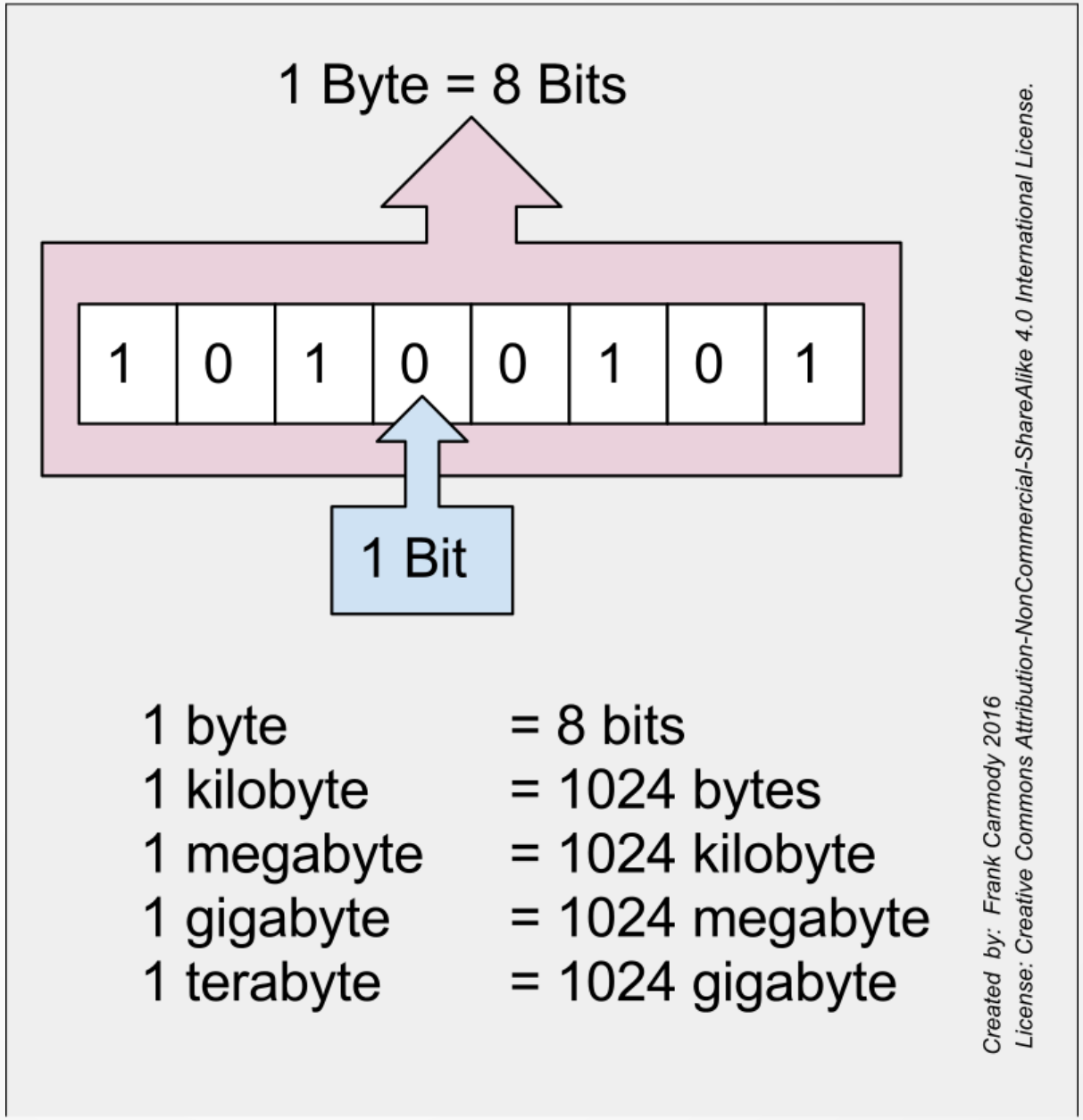
```
fp = open('mytxtfile.txt', 'a+')
```

- Write a string value to it

```
value = 'defgh'
fp.write(value)
```

- Mode ‘w’ truncates the file
- Use ‘a’ in that case
- Use ‘w+’ to open with both write and read

Writing Binary Files Manually



Writing Binary Files Manually

- Refer slide52.py

Using Pickle To Write Binary Files

- Refer slide53.py
- Use pickling when you have deal with large python objects

Python Functions

- Python User Defined Functions
- Python Package Functions
- Defining And Calling Functions
- The Anonymous Function
- Loops And Statement In Python
- Python Modules And Packages

User Defined Function

- Get a number from user, say ip=4
- Print in below formats
- **NOTE**: the code should use a function

```
1
11
111
1111
```

```
1
121
12321
1234321
```

Need to cover

Loop

Tuple

Nested list

Difference between 1 and ‘1’

Creating a iterator with class